



Gentoo Linux

сборник статей

Версия 1.5 от 02.04.2008



2008 г.

Оглавление

Новое и измененное (changelog).....	1
Составители	4
Цели данного сборника.....	5
Обращение к русскому сообществу Gentoo.....	6
Вступительное слово.....	7
Вводная информация о Linux.....	8
UNIX.....	8
История.....	8
Стандарты.....	14
Канонические команды UNIX.....	15
GNU/Linux.....	16
Linux (ядро).....	19
История.....	19
Версии.....	20
Поддержка.....	21
Стабильные версии.....	22
Архитектура.....	22
Совместимость.....	23
Лицензия.....	24
Символ.....	24
BSD.....	25
История BSD.....	25
GNU General Public License.....	27
Свобода.....	27
История.....	27
GPL v3.....	28
Схема GNU GPL.....	29
Linux не для идиотов: сборник рассказов и рецептов.....	30
Предисловие.....	30
Ядро и модули.....	30
Загрузка операционной системы.....	33
Организация памяти.....	34
System V shared memory.....	35
Файловая система.....	37
Права доступа.....	40
Журналируемые файловые системы.....	41
Отображенные в память файлы.....	42
Специальные файловые системы.....	43
Сетевые файловые системы.....	43
Создание файловых систем.....	44
Интерфейс sysctl.....	44
Статически и динамически собранные программы.....	45
Системная библиотека GNU libc.....	46
LD, Shared Library, SO и много страшных слов.....	47
Информация о процессах и файловая систем /proc.....	49
Создание процессов.....	50
Секреты /dev.....	50

Статическая организация каталога /dev.....	52
Что такое DevFS.....	53
Немного о UDEV.....	54
Блочные устройства.....	54
Распределение мажоров и миноров IDE дисков.....	57
Распределение мажоров и миноров для SCSI-дисков.....	58
Устройства SATA и переход с IDE на PATA.....	59
Logical Volume Manager.....	60
Software RAID.....	64
Device mapper.....	67
Host-RAID, или дешевых RAID-контроллеров не бывает.....	67
Сетевая подсистема.....	68
Маршрутизация IP и форвардинг.....	70
Фильтры пакетов.....	72
Стандартные таблицы и цепочки.....	73
Порядок применения стандартных таблиц и цепочек.....	74
Стандартные действия.....	75
Условия отбора.....	75
Примеры конфигураций iptables.....	75
Управление пользователями, NSS и PAM.....	78
X11 и все-все-все.....	80
Клиент-серверная архитектура X11.....	80
Графическая среда пользователя.....	81
Внутри X11. Упрощенная схема.....	82
Внутри X11. Растровые шрифты и их отображение.....	83
Внутри X11. FreeType и XFT.....	84
Внутри X11. Расширения.....	85
Внутри X11. Полезные утилиты.....	85
Внутри X11. Менеджер окон.....	85
Локализация.....	85
Файловые системы Linux.....	88
Создание файловых систем.....	88
Необходимое введение.....	88
Устройство файловых систем Unix-семейства.....	90
Файловые системы Linux.....	91
Ext2fs.....	91
О журналируемых файловых системах.....	92
ReiserFS.....	92
Ext3fs.....	93
XFS.....	94
Критерии выбора.....	95
Практические следствия.....	96
Библиография вопроса.....	98
Linux и файловые системы: еще раз о проблеме выбора.....	99
Windows-эквиваленты.....	103
Прикладные и системные программы.....	103
Работа в сети.....	103
Офис/Бизнес.....	104
Chat.....	105

File managers.....	105
FTP.....	105
Graphics.....	105
Internet.....	105
Mail.....	106
Multimedia.....	106
Office.....	106
P2P.....	106
О Gentoo Linux.....	107
Что такое портежи?.....	107
Gentoo Linux.....	108
История Gentoo.....	108
Философия Gentoo.....	109
Советы пользователям Gentoo	112
Проект X.org.....	113
Основы Portage.....	113
Управление настройками Portage.....	114
Когда нужно использовать оверлеи?.....	114
Как часто нужно обновляться?.....	115
Деинсталляция пакетов.....	116
Удаление временных файлов.....	116
Gentoo Linux: установка и настройка.....	117
Получение дистрибутива.....	118
Установка - первый этап.....	123
Этап второй - подготовка к смене корня.....	125
Третий этап - внутри Gentoo.....	126
Этап четвертый: в недра ядра.....	129
Краткое руководство по установке Gentoo Linux x86.....	139
Установочный диск.....	139
Настройка сети.....	139
Подготовка дисков.....	139
Установка стадии.....	140
Установка базовой системы Gentoo.....	140
Настройка ядра.....	141
Настройка системы.....	141
Установка системных утилит.....	142
Настройка загрузчика.....	142
Grub.....	142
Lilo.....	143
Окончание установки.....	143
Описание альтернативных способов установки Gentoo Linux.....	144
1. Об этом документе.....	144
2. Загрузка с установочного диска при помощи Smart BootManager.....	144
3. Установка при помощи Knoppix.....	145
4. Установка со Stage 1 без доступа в сеть.....	146
5. Бездисковая установка с загрузкой через PXE.....	147
6. Установка Gentoo из существующей Linux-системы.....	151
Полезные советы по установке Gentoo/x86.....	155
1. Вступление.....	155

2. Расширенная установка.....	155
3. Упрощение установки.....	158
4. Решение ошибок/проблем.....	159
Восстановление сбойной установки.....	159
Шпаргалка по установке Gentoo.....	161
Описание	161
Подготовка к установке	161
Порядок основных действий и команд	161
Разбиение диска	161
Установка базовой системы	161
Файлы настройки	162
Оживление Gentoo	163
Сборка ядра	164
Прочие системные установки	164
Настройка загрузчика	165
Локализация в UNICODE	165
Загрузка Gentoo	166
Вспомогательная информация	167
Логика разметки диска	167
Логика настройки ядра	168
Настольная книга Gentoo Linux x86.....	170
А. Установка Gentoo.....	173
1. Об установке Gentoo Linux.....	173
1.a. Введение.....	173
2. Выбор подходящего источника для установки.....	175
2.a. Аппаратные требования.....	175
2.b. Установочные компакт-диски Gentoo.....	176
2.c. Скачивание, запись и загрузка с установочного диска Gentoo.....	177
3. Настройка сети.....	182
3.a. Автоматическое подключение к сети.....	182
3.b. Автоматизированная настройка сети.....	183
3.c. Ручная настройка сети.....	185
Использование DHCP.....	186
Подготовка беспроводного доступа.....	187
Освоение сетевой терминологии.....	188
Использование ifconfig и route.....	189
4. Подготовка дисков.....	190
4.a. Общие сведения о блочных устройствах.....	190
4.b. Разработка схемы разбиения диска.....	191
4.c. Использование fdisk для создания разделов.....	193
4.d. Создание файловых систем.....	196
Файловые системы (сравнение !).....	197
4.e. Монтирование.....	198
5. Установка установочных файлов Gentoo.....	199
5.a. Установка архива стадии.....	199
5.b. По умолчанию: использование файла стадии из интернета.....	200
5.c. Альтернатива: использование файла стадии с установочного диска	201
5.d. Установка дерева портежей.....	202

5.e. Настройка параметров компиляции.....	203
6. Установка базовой системы Gentoo.....	205
6.a. Изменение корневого каталога.....	205
6.b. Настройка Portage.....	207
Настройка переменной USE.....	209
7. Настройка ядра.....	211
7.a. Часовой пояс.....	211
7.b. Установка исходных кодов ядра.....	212
7.c. По умолчанию: ручная настройка.....	212
7.d. Альтернатива: использование genkernel.....	216
7.e. Модули ядра.....	218
8. Настройка параметров системы.....	218
8.a. Параметры файловых систем.....	218
Примечание (JohnBat26): пример моего /etc/fstab:.....	220
8.b. Параметры сети.....	220
8.c. Параметры системы.....	223
9. Установка нужных системных средств.....	224
9.a. Диспетчер устройств.....	224
9.b. Системные службы журналирования.....	224
9.c. Дополнительно: демон cron.....	225
9.d. Дополнительно: индексация файлов.....	226
9.e. Утилиты для обслуживания файловых систем.....	226
9.f. Сетевые средства.....	226
10. Настройка начального загрузчика.....	227
10.a. Выбор загрузчика.....	227
10.b. По умолчанию: использование GRUB.....	229
10.c. Альтернатива: использование LILO.....	233
10.d. Перезагрузка системы.....	235
11. Завершение установки Gentoo.....	235
11.a. Управление учетными записями.....	235
12. Чем заняться дальше?.....	236
12.a. Документация.....	236
12.b. Gentoo в интернете.....	237
B. Работа с Gentoo.....	238
1. Введение в Portage.....	238
1.a. Добро пожаловать в Portage.....	238
1.b. Дерево портежей.....	238
1.c. Обслуживание программного обеспечения.....	239
1.d. Когда Portage жалуется.....	243
2. USE-флаги.....	247
2.a. Что такое USE-флаги?.....	247
2.b. Использование USE-флагов.....	248
2.c. USE-флаги отдельных пакетов.....	251
3. Возможности Portage.....	252
3.a. Возможности Portage.....	252
3.b. Распределенная компиляция.....	252
3.c. Кэширование компиляции.....	253
3.d. Поддержка двоичных пакетов.....	255
4. Сценарии инициализации.....	256

4.a. Уровни запуска.....	256
4.b. Использование rc-update.....	260
4.c. Настройка служб.....	260
4.d. Написание сценариев инициализации.....	261
4.e. Изменение поведения уровней запуска.....	264
5. Переменные среды.....	265
5.a. Переменные среды.....	265
5.b. Глобальное определение переменных.....	267
5.c. Локальное определение переменных.....	269
C. Работа с Portage.....	271
1. Файлы и каталоги.....	271
1.a. Файлы Portage.....	271
1.b. Хранение файлов.....	272
1.c. Сборка программного обеспечения.....	273
1.d. Ведение журнала.....	274
2. Настройка с помощью переменных.....	274
2.a. Настройка Portage.....	274
2.b. Параметры сборки.....	274
2.c. Защита конфигурационных файлов.....	275
2.d. Параметры скачивания.....	276
2.e. Настройка Gentoo.....	277
2.f. Поведение Portage.....	277
3. Смещение ветвей программного обеспечения.....	278
3.a. Использование одной ветви.....	278
3.b. Одновременное использование стабильной и тестовой ветвей.....	279
3.c. Использование заблокированных пакетов.....	279
4. Дополнительные средства Portage.....	280
4.a. etc-update.....	280
4.b. dispatch-conf.....	281
4.c. quickpkg.....	282
5. Отступление от официального дерева.....	282
5.a. Использование собственного дерева Portage.....	282
5.b. Добавление неофициального сборочного файла ebuild.....	283
5.c. Программы, поддерживаемые не Portage.....	284
6. Использование ebuild.....	284
6.a. Emerge и Ebuild.....	284
6.b. Ручная установка программ.....	284
6.c. Дополнительные возможности Ebuild.....	287
6.d. Дополнительная информация.....	287
D. Настройка сети в Gentoo.....	289
1. Начальная настройка.....	289
1.a. Приступаем к настройке.....	289
2. Расширенная настройка.....	290
2.a. Расширенная настройка.....	290
2.b. Сетевые зависимости.....	291
2.c. Имена и значения переменных.....	292
3. Модульное построение сети.....	293
3.a. Сетевые модули.....	293
3.b. Обработчики интерфейса.....	294

3.c. DHCP.....	294
3.d. Модем ADSL.....	296
3.e. APIPA (автоматическая частная IP-адресация).....	296
3.f. Объединение интерфейсов.....	297
3.g. Образование моста (поддержка 802.1d).....	297
3.h. MAC-адрес.....	298
3.i. Образование тоннеля.....	298
3.j. Виртуальные сети (поддержка 802.1q).....	299
4. Беспроводная сеть.....	299
4.a. Введение.....	299
4.b. Запросчик WPA.....	300
4.c. Утилиты Wireless tools.....	302
4.d. Раздельная настройка сети по ESSID.....	305
5. Дополнительные возможности.....	306
5.a. Стандартные функции-обработчики.....	306
5.b. Функции-обработчики wireless tools.....	307
6. Управление сетью.....	308
6.a. Управление сетью.....	308
6.b. ifplugd.....	309
Настольная книга по безопасности Gentoo.....	310
Введение.....	310
А. Безопасность системы.....	310
1. Соображения перед началом установки.....	310
2. Закручивание гаек.....	313
3. Журналирование.....	315
4. Монтирование разделов.....	320
5. Ограничения пользователей/групп.....	321
6. Права доступа к файлам.....	323
7. RAM.....	326
8. Упаковщики TCP.....	327
9. Безопасность ядра.....	327
10. Безопасность служб.....	330
11. Изменение корневого каталога и виртуальные серверы.....	338
12. Межсетевые экраны.....	339
13. Обнаружение вторжения.....	353
14. Регулярные обновления.....	358
Hardened Gentoo.....	360
Описание.....	360
Установка.....	362
Настройка.....	364
Впечатления.....	367
Сложность.....	367
Стабильность.....	367
Функциональность.....	367
Производительность.....	367
Выводы.....	368
Обновление Gentoo Linux.....	369
Руководство по обновлению Gentoo.....	369
1. Gentoo и обновления.....	369

2. Гонка за новыми выпусками.....	370
3. Инструкции по обновлению профилей.....	371
HOWTO Полное обновление системы.....	373
Приведение в порядок /var/lib/portage/world	373
Обновление profile	373
USE-флаги	373
Запуск обновления системы (если не нужно обновлять toolchain)	374
Некоторые причины не использовать emerge -U вместо -u	374
Обновление одного из пакетов входящих в toolchain	374
Обновление безопасности.	375
Удаление неиспользуемых пакетов.	375
Обновление конфигов.	376
Руководство Gentoo Linux по обновлению GCC.....	377
1. Введение.....	377
2. Общие указания по обновлению.....	377
3. Переход с GCC-3.3 на 3.4 или более новый.....	378
4. Обновление GCC на новой установке.....	381
5. Обычные грабли.....	382
Gentoo и X.Org.....	384
Описание настройки МОНОЛИТНОГО X-сервера версий 6.X.....	384
1. Что такое X Window Server?.....	384
2. Установка Xorg.....	385
3. Настройка Xorg.....	386
4. Настройка xorg.conf.....	388
5. Ресурсы.....	390
Описание настройки МОДУЛЬНОГО X-сервера версий 7.X (Англ.).....	392
Introduction.....	392
Preparation	392
Emerge Modular X.org	394
Installation	394
GL Issues.....	396
Configuration file locations have changed	397
Other problems and solutions	397
XkbOptions.....	398
Параметры "XkbOptions"	398
Пример	398
HOWTO Xorg and Fonts (Рус.).....	400
Назначение этого HOWTO.....	400
Общие сведения.....	400
Замечание по поводу DPI.....	400
Замечание по поводу сравнения шрифтов на Windows/Mac и Linux.....	401
Получение необходимых пакетов.....	401
Информация о хинтинге (Hinting).....	402
Изменение конфигурационных файлов.....	403
/etc/X11/xorg.conf.....	406
Файл: /etc/X11/xorg.conf (Список путей к шрифтам).....	409
Перезагрузка служб.....	409
Mozilla Firefox and Thunderbird.....	410
Проблемы с размерами шрифтов.....	412

Что делать, если Linux шрифт в Konsole не работает.....	413
Что, если Ваши шрифты стали размытыми и не читаемыми.....	413
Ошибка при запуске startx.....	413
XEmacs.....	414
Полезные ресурсы.....	414
Смотрите также.....	414
Примечание (от JohnBat26), мой файл: ~/.fonts.conf:.....	415
Руководство по видеокартам nVidia в Gentoo Linux.....	417
1. Введение.....	417
2. Настройка карты.....	417
3. Проблемы.....	422
4. Настройка для опытных.....	423
Распространенные вопросы об ATI в Gentoo Linux.....	425
1. Поддержка устройств.....	425
2. Установка.....	426
3. Дополнительные источники.....	427
HOWTO fb splash.....	428
Введение	428
Что такое gensplash?.....	428
Что такое fb splash (framebuffer splash)?.....	428
Что такое устройство frame buffer?.....	428
Установка fb splash и splashutils.....	429
Установка заплат ядра.....	429
Конфигурация ядра.....	429
Компиляция ядра (Вручную).....	429
Конфигурация / компиляция ядра (Для пользователей genkernel).....	430
Установка splashutils.....	430
Конфигурация / компиляция ядра (ручная работа).....	430
Создание образа initramfs	430
Что такое initramfs?.....	431
Компиляция образа initramfs непосредственно в ядро.....	431
Загрузка образа initramfs динамически.....	431
Создание собственных тем для gensplash.....	432
Перевод тем bootsplash в темы gensplash.....	433
Использование собственных тем gensplash.....	433
Настройка загрузчика	434
Параметры ядра.....	434
Пример конфигурации Grub.....	434
Пример конфигурации Lilo.....	435
Добавление фона на все консоли	435
Gensplash для других дистрибутивов	435
Дополнительные шаги.....	436
Тут должно быть название пункта 3.....	436
Решение проблем	436
Если /usr на отдельном разделе.....	436
Установка фона во время загрузки.....	436
Ссылки.....	437
Hardware 3D Acceleration Guide (Англ.).....	438
1. Introduction.....	438

2. Install Xorg and configure your kernel.....	439
3. Install X11-DRM and configure direct rendering.....	440
4. Test 3D acceleration.....	441
5. Using the CVS sources.....	441
6. Tweak your performance.....	442
7. Troubleshooting.....	444
8. Acknowledgments.....	445
9. References.....	445
Gentoo и KDE.....	446
HOWTO по настройке KDE версии 3.x.x.....	446
1. Что же такое K Desktop Environment?.....	446
2. Установка KDE.....	446
3. Настройка KDE.....	449
Способы переключения раскладок клавиатуры в KDE (3в1).....	451
Gentoo KDE 4.0 Guide (Рус.).....	457
1. Установка.....	457
2. Изменения по сравнению с KDE 3.5.....	464
3. FAQ (Часто задаваемые вопросы).....	464
Руководство по настройке GNOME.....	467
1. Что такое GNOME?.....	467
2. Установка GNOME.....	467
Использование UTF-8 в Gentoo.....	469
1. Предупреждение или Напутствие переводчика.....	469
2. Кодировки.....	469
3. Включение UTF-8 в Gentoo Linux.....	471
4. Поддержка приложениями.....	473
Руководство по русской локализации Gentoo Linux.....	481
1. Введение.....	481
2. Базовая кириллизация консоли.....	481
3. Установка русской locale.....	485
4. Установка часового пояса.....	487
5. Локализация оконной системы X.....	487
Листинг 5.1: Варианты выбора переключателя латиница/кириллица.....	488
Руководство Gentoo Linux ALSA.....	492
1. Введение.....	492
2. Установка ALSA.....	492
3. Настройка/тестирование ALSA.....	497
ALSA и USE.....	500
4. Другие возможности ALSA.....	503
Советы пользователям Gentoo (вариант 2).....	507
Если вы только собираетесь начать работу с Gentoo.....	507
Основы Portage.....	507
Настройка Portage.....	508
Когда нужно использовать оверлеи.....	508
Как часто нужно обновляться?.....	509
Держите «мир» в чистоте.....	510
Об удалении пакетов.....	510
Логи Portage.....	510
Удаление временных файлов.....	511

Быстрый поиск пакетов.....	511
Уменьшение использования трафика.....	511
Ускорение компиляции.....	512
Обновление конфигурационных файлов.....	512
Ещё раз о главном - жизнь после установки.....	514
Полезные пакеты и команды:.....	514
Обновление конфигов.....	515
USE-флаги.....	516
Поддержание системы в чистоте и порядке =).....	518
Проверка целостности системы:.....	518
Установка пакетов. X'сы, DE.....	519
Gentoo и Java.....	520
Руководство по настройке Java в Gentoo.....	520
1. Что такое Java ?.....	520
2. Перед тем как начать.....	521
3. Установка виртуальной машины.....	522
4. Настройка Вашей виртуальной машины.....	523
5. Компиляторы.....	524
6. Установка переменной CLASSPATH по умолчанию.....	525
7. Java-плагины для браузера.....	526
8. USE флаги, используемые с Java.....	527
9. Дополнительные ресурсы.....	528
Руководство по обновлению Java в Gentoo.....	529
1. Введение.....	529
2. Обновление java-config.....	530
3. Проверка вашего окружения.....	530
4. Обновление ... завершено!.....	531
5. Общие проблемы и вопросы.....	531
Gentoo и GPRS.....	532
GPRS через USB (Руководство).....	532
Настройка GPRS в Linux.....	535
Введение.....	535
Подготовка.....	535
Сборка пакетов.....	535
Настройка.....	537
Установка соединения.....	538
Автоматическая установка соединения.....	538
Заключение.....	539
Экономим трафик, с помощью Toonel. (Linux).....	539
Сжимаем трафик электронной почты.....	540
GPRS через KPPP.....	541
GPRS через Bluetooth.....	543
HOWTO Соединение с Internet.....	544
Настройка VPN.....	544
Проверка параметров ядра	544
Установка пакетов	544
Настройка клиента.....	544
Настройка с помощью pptpconfig.....	545
Проверяем соединение	546

Через pppoe.....	547
Настройка выхода через DIAL-UP.....	548
Подключение модема.....	548
Настройка программного обеспечения.....	549
Настройка Call back.....	550
Настройка выхода через GPRS.....	551
Требуемая аппаратура.....	551
Подключение с помощью USB-кабеля.....	551
Подключение с помощью COM-кабеля.....	552
Подключение через Bluetooth.....	552
Подключение через IrDA.....	553
Настройка выхода через CDMA (SKYLINK).....	554
Подключение аппаратуры.....	554
Настройка программного обеспечения.....	554
Настройка спутникового интернета в Gentoo.....	555
Введение.....	555
Как работает спутниковый Интернет?	555
Оборудование.....	555
Подготовка Gentoo к использованию спутникового Интернета.....	555
Настройка ядра.....	555
Установка пакета утилит.....	556
Настройка параметров системы.....	556
dvb_core.....	556
/etc/channels.conf.....	556
Сетевой интерфейс.....	557
Сетевой фильтр.....	558
Проверка работоспособности.....	558
Использование спутникового канала.....	560
Соединение нескольких офисов в одну сеть с помощью OpenVPN.....	561
Управление питанием компьютера.....	564
1. Введение.....	564
Краткий обзор.....	564
Покомпонентное распределение энергии.....	565
2. Подготовка.....	565
BIOS.....	565
Настройка USE-флагов.....	565
Конфигурация ядра.....	566
Создание уровня запуска (runlevel) "battery".....	567
Реагирование на ACPI-события.....	567
3. Управление питанием ЦПУ.....	569
Техническая часть.....	570
Установка частоты вручную.....	570
Автоматическая настройка частоты.....	571
Проверка результата.....	574
4. LCD Power Management.....	574
Настройки засыпания (standby).....	575
Уменьшение подсветки.....	575
Управление питанием диска.....	576
Увеличение времени покоя (idle time) - laptop-mode.....	577

Ограничение доступа на запись.....	578
hdparm.....	578
Другие "типы и трюки".....	578
6. Управление питанием других устройств.....	579
Графические карты.....	579
Управление питанием беспроводных соединений.....	579
Управление питанием USB.....	579
7. Состояния спячки: sleep, standby, suspend to disk.....	580
Sleep (S3).....	580
Hibernate (S4).....	581
8. Устранение неполадок.....	583
Gentoo и СУБД.....	586
Руководство по MySQL для начинающих.....	586
1. Начало работы с MySQL.....	586
2. Настройка базы данных.....	588
3. Работа с таблицами в MySQL.....	589
4. Привилегии в MySQL.....	593
PostgreSQL Guide.....	599
1. Introduction.....	599
2. PostgreSQL configuration.....	600
3. Using PostgreSQL.....	603
Gentoo и системы контроля версий.....	608
Gentoo и Subversion.....	608
HOWTO Subversion сервер при помощи Apache2 и WebDav.....	608
Руководство по udev в Gentoo.....	617
1. Что такое udev?.....	617
2. Как использовать udev в Gentoo.....	619
3. Известные проблемы.....	620
4. Ссылки на ресурсы и благодарности.....	623
Gentoo LDAP-DNS Guide (Англ.).....	624
1. Introduction.....	624
2. Configuring LDAP-DNS.....	624
3. Configuring OpenLDAP.....	625
4. Testing the Installation.....	626
Руководство по настройке Samba в режиме PDC с использованием LDAP.....	627
Устанавливаемые пакеты	627
OpenLDAP.....	627
Конфигурирование OpenLDAP.....	627
Запуск сервера OpenLDAP.....	628
SAMBA.....	630
Миграция OpenLDAP.....	630
Конфигурация SAMBA.....	632
Настройка системы на авторизацию в LDAP.....	634
Управление пользователями.....	636
Создание пользователя.....	636
Изменение пароля.....	637
Модификация пользователя.....	637
Удаление пользователя.....	638
Управление пользователями в оффтопике.....	639

Источники.....	639
Создание виртуальной почтовой системы.....	640
1. Введение.....	640
2. Начальная настройка postfix.....	641
3. Courier-imap.....	642
4. Cyrus-sasl.....	643
5. SSL-сертификаты для Postfix и Apache.....	643
6. Добавим SSL и SASL поддержку в Postfix.....	644
7. MySQL.....	646
8. Apache и phpMyAdmin.....	647
9. Vmail-пользователь.....	648
10. Настройка MySQL авторизации и виртуальных доменов.....	649
11. Squirrelmail.....	652
12. Mailman.....	652
13. Фильтрация содержимого и Anti-Virus.....	654
14. Окончание.....	654
15. Troubleshooting.....	655
Виртуальная почтовая система на основе qmail/vpopmail.....	658
1. Введение.....	658
2. qmail ("Разговаривая с собой").....	658
3. vpopmail.....	660
4. Courier POP/IMAP.....	662
5. qmail (разговаривая с миром).....	662
6. Веб клиент Horde / IMP.....	663
7. Дополнительные пакеты.....	665
8. Заключительные примечания.....	667
Руководство по электронной почте с использованием Mutt.....	668
1. Введение в E-Mail.....	668
2. Fetchmail.....	668
3. Procmal.....	669
4. Почтовый клиент Mutt.....	671
5. SMTP.....	674
Руководство по файловой системе для устройств.....	676
1. Что такое devfs?.....	676
Проблемы.....	677
Решения.....	677
2. Навигация через дерево устройств.....	678
3. Администрирование дерева устройств.....	679
4. Вопросы, относящиеся к правам доступа.....	680
Ebuild HOWTO (Англ.).....	683
1.a. The Portage tree.....	683
1.b. Ebuild scripts.....	685
1.c. File system Locations.....	695
1.d. The Portage scripts and utilities.....	696
1.e. Package Dependencies.....	700
1.f. Testing and deploying.....	701
Использование rpm-пакетов в Gentoo.....	704
Изучаем параметры gcc.....	707
Параметры, относящиеся к оптимизации кода.....	708

Параметры, относящиеся к вызову функций.....	712
Опции, относящиеся к отладке.....	716
Опции, управляющие стадиями компиляции.....	718
Заключение.....	721
Благодарности.....	721
Ссылки.....	721
Создание LiveCD дистрибутива на базе Gentoo Linux.....	723
Введение.....	723
Подготовительные действия.....	723
Собираем базовую систему.....	724
Установка KDE.....	727
Что еще можно сделать.....	727
Тюнинг Gentoo.....	729
Ядро 2.6: Тонкости настройки.....	729
Параметры виртуальной памяти	729
Руководство по предварительному связыванию в Gentoo Linux.....	730
1. Введение.....	730
2. Инициализация предварительного связывания.....	731
3. Предварительное связывание.....	732
4. Известные проблемы и методы их решения.....	733
5. Заключение.....	735
Тюнинг системы и запуска некоторых программ.....	736
Введение.....	736
mplayer.....	736
XMMS.....	736
Тюнинг настроек программ из семейства Mozilla.....	737
Navigator(Seamonkey/Firefox).....	737
Управление Кешем в Firefox и Seamonkey.....	737
Управление логотипами сайтов.....	738
Для перешедших с Firefox на Seamonkey.....	738
Thumbnail'ы просматриваемых картинок в табках.....	738
Тонкая настройка IDE дисков с помощью hdparm.....	739
Ускорение загрузки системы.....	742
Что делать, если система «зависла».....	745
1. «Завис» X сервер.....	745
2. Приложение не воспринимает команды.....	745
Русское WIKI.....	747
HOWTO hal и устройства ввода.....	747
hal-0.5.10	747
Обновляем portage	747
Обновляем установку xorg	747
Клавиатура	747
Перезапускаем X-ы	749
Мышка.....	749
Touchpad.....	750
HOWTO lvman.....	750
Цель.....	750
Требуемые программы.....	750
Использование.....	750

Проблемы	752
Разное.....	754
Другие проблемы.....	754
Настройка дисковой подсистемы.....	754
Всеобщий ликбез.....	754
Работа с CD/DVD.....	765
Основные программы	765
Создание DVD.....	769
Введение.....	769
Система Portage.....	775
Portage через NFS.....	775
Настройка Portage.....	778
Portage Setup Howto	778
Portage Overlay.....	782
Введение.....	782
Установка Portage Overlay.....	782
Месторасположение ebuild.....	782
Создание digest-файлов.....	782
Экономия трафика с помощью deltap.....	784
Что это такое?.....	784
Установка и настройка.....	784
Обновление портежей без доступа в интернет непосредственно из системы	785
Обновление пакетов без доступа в интернет непосредственно из системы	786
Полное обновление системы.....	787
Сборка на другом компьютере.....	791
Прочее.....	792
Разделы MS Windows по-русски.....	792
Udev.....	795
Использование CFLAGS для оптимизации собранных программ.....	797
Редактор nano.....	800
Udev и автомонтирование носителей.....	801
Оптимизация glibc.....	804
Подключение USB-flash.....	808
Чтение-запись NTFS с использованием драйвера ntfs-3g.....	811
Сетевые сервисы.....	814
Установка Apache2.....	814
Настройка iptables для начинающих.....	818
Подробная настройка iptables.....	823
Установка почтовой системы.....	840
Настройка vsftpd.....	847
Jabber Server.....	850
Выделенный сервер Counter-Strike.....	851
Файл сервер Samba в домене Win2k.....	852
Руководство по настройке Samba в режиме PDC с использованием LDAP	856
Мониторинг работы системы.....	869
Настройка X.....	889

Настройка русских шрифтов в X.....	889
Настройка размера шрифтов в X.....	893
Xorg X11 и прозрачность.....	895
Xorg. Переключение раскладок.....	899
Установка Xgl.....	899
XGL.....	902
Автоматическое переключение раскладок.....	924
Cedega 5.1.....	925
Секреты командной строки.....	926
Тонкость работы ссacne с emerge.....	934
Keuymap.....	934
Часто задаваемые вопросы.....	936
2. Введение.....	936
Как произносится слово Gentoo, и что оно означает?.....	936
Что делает Gentoo особенным?.....	936
3. Установка.....	936
Все работает очень нестабильно, мои флаги оптимизации "-O9 -ffast-math -fomit-frame-pointer". В чем проблема?	937
Как изменить пароль root (или любого другого пользователя)?.....	937
Как добавить обычного пользователя?.....	937
Почему пользователь не может получить права root командой su?.....	937
Как отключить devfs?.....	938
Могу ли я обновить Gentoo без переустановки?	938
Ядро не загружается (правильно), что мне теперь делать?.....	938
Мой прокси-сервер требует аутентификации, что делать?.....	939
Как записать ISO файл на компакт-диск?.....	939
Какой диск/стадию надо использовать с моим процессором?.....	940
Я не могу выйти в интернет после перезагрузки. В чем может быть дело?	940
Я хочу загрузить Windows из GRUB или LILO, но вижу только черный экран. Что делать?	941
Как установить Gentoo, используя файл Stage1 или Stage2?.....	941
4. Управление пакетами.....	942
В каком формате хранятся пакеты?.....	942
Я хочу выполнить сценарий ./configure сам. Возможно ли это?.....	943
Как использовать emerge, если я за межсетевым экраном?.....	943
Что делать, если rsync мне не подходит?.....	943
У меня дома лишь слабое модемное подключение. Могу ли я скачать исходные коды в другом месте, а затем добавить их в систему?	943
Исходные коды хранятся в /usr/portage/distfiles. Безопасно ли удалить их?	943
Что находится в /var/tmp/portage? Можно ли удалить файлы и каталоги из /var/tmp/portage?	944
5. Использование.....	944
Как установить международную раскладку клавиатуры?.....	944
Почему мой пользователь не может использовать свой файл crontab?..	944
Как включить numlock при загрузке?.....	945
Как сделать очистку терминала при выходе?.....	945
Не удается запускать X приложения из-под root'a (после su).....	945

6. Сопровождение.....	945
ReiserFS и порча файловых систем - как исправить, и т.п.....	946
7. Разработка.....	946
Где оставлять сообщения об ошибках?.....	946
Как часто появляются новые стабильные выпуски?.....	946
Мой динамик орет как сумасшедший. Как выключить динамик в консоли?.....	946
8. Ресурсы.....	947
Где найти информацию о Gentoo Linux?.....	947
Могу ли я купить CD с Gentoo Linux?.....	947
В этом списке нет ответа на мой вопрос. Что делать?.....	947
Наивные вопросы юных линуксоидов.....	948
Вступление.....	948
Как ставить программы?	948
Что еще за тарболл?	948
А кто такой root и как им стать?.....	950
Основные команды.....	951
Команда man.....	951
Команда cd.....	951
Команда ls.....	951
Команда cp.....	952
Команда rm.....	952
Команда df.....	952
А как мне быть если путь или имя очень длинные?.....	953
Как сделать чтобы программа запущенная из терминала работала после его закрытия?.....	953
А как работать с буфером?.....	953
Как заставить иксы стартовать при загрузке?.....	954
Термины.....	955
Запрещённые термины, или КАК НЕ НАДО.....	968
MAN (справки по командам).....	970
Emerge.....	970
Примеры конфигурационных файлов.....	986
1. Мой make.conf.....	986
2. Файл xorg.conf для NVIDIA Geforce.....	988
3. Файл xorg.conf для ATI radeon:.....	991
4. Файл /etc/fstab.....	997
Важные команды Gentoo Linux.....	998
Unix Toolbox.....	999
1. SYSTEM.....	999
1.1 Hardware Informations.....	999
1.2 Load, statistics and messages.....	1000
1.3 Users.....	1000
1.4 Limits.....	1000
1.5 Runlevels.....	1001
1.6 Reset root password.....	1002
1.7 Kernel modules.....	1003
1.8 Compile Kernel.....	1003
2. PROCESSES.....	1004

2.1 Listing and PIDs.....	1004
2.2 Priority.....	1004
2.3 Background/Foreground.....	1004
2.4 Top.....	1004
2.5 Signals/Kill.....	1005
3. FILE SYSTEM.....	1005
3.1 Permissions.....	1005
3.2 Disk information.....	1005
3.3 Boot.....	1005
3.4 System mount points/Disk usage.....	1006
3.5 Who has which files opened.....	1006
3.6 Mount/remount a file system.....	1007
3.7 Mount an SMB share.....	1007
3.8 Mount an image.....	1008
3.9 Create and burn an ISO image.....	1008
3.10 Create a file based image.....	1009
3.11 Create a memory file system.....	1009
3.12 Disk performance.....	1010
4. NETWORK.....	1010
4.1 Debugging (See also Traffic analysis) (page 18).....	1010
4.2 Routing.....	1010
4.3 Configure additional IP addresses.....	1010
4.4 Change MAC address.....	1011
4.5 Ports in use.....	1011
4.6 Firewall.....	1011
4.7 IP Forward for routing.....	1011
4.8 NAT Network Address Translation.....	1012
4.9 DNS.....	1012
4.10 DHCP.....	1013
4.11 Traffic analysis.....	1013
4.12 Traffic control (QoS).....	1014
4.13 NIS Debugging.....	1015
5. SSH SCP.....	1016
5.1 Public key authentication.....	1016
5.2 Check fingerprint.....	1017
5.3 Secure file transfer.....	1017
5.4 Tunneling.....	1017
6. VPN WITH SSH.....	1019
6.1 Single P2P connection.....	1019
6.2 Connect two networks.....	1020
7. RSYNC.....	1020
7.1 Rsync on Windows.....	1021
8. SUDO.....	1022
8.1 Configuration.....	1022
9. ENCRYPT FILES.....	1023
9.1 A single file.....	1023
9.2 tar and encrypt a whole directory.....	1023
9.3 tar zip and encrypt a whole directory.....	1023
10. ENCRYPT PARTITIONS.....	1023

10.1 Linux.....	1023
10.2 FreeBSD.....	1024
11. SSL CERTIFICATES.....	1025
11.1 Procedure.....	1026
11.2 Configure OpenSSL.....	1026
11.3 Create a certificate authority.....	1026
11.4 Create a certificate signing request.....	1026
11.5 Sign the certificate.....	1027
11.6 Create united certificate.....	1027
11.7 View certificate information.....	1027
12. CVS.....	1027
12.1 Server setup.....	1027
12.2 Test it.....	1029
12.3 SSH tunneling for CVS.....	1029
12.4 CVS commands and usage.....	1030
13. SVN.....	1030
13.1 Server setup.....	1030
13.2 SVN commands and usage.....	1032
14. USEFUL COMMANDS.....	1032
14.1 less.....	1032
14.2 vi.....	1033
14.3 mail.....	1033
14.4 tar.....	1034
14.5 dd.....	1034
14.6 screen.....	1035
14.7 Find.....	1036
14.8 Miscellaneous.....	1036
15. INSTALL SOFTWARE.....	1037
15.1 List installed packages.....	1037
15.2 Add/remove software.....	1037
15.3 Library path.....	1037
16. CONVERT MEDIA.....	1037
16.1 Text encoding.....	1037
16.2 Unix - DOS newlines.....	1038
16.3 PDF to Jpeg and concatenate PDF files.....	1038
16.4 Convert video.....	1038
16.5 Copy an audio cd.....	1038
17. PRINTING.....	1038
17.1 Print with lpr.....	1038
18. DATABASES.....	1039
18.1 PostgreSQL.....	1039
18.2 MySQL.....	1039
18.1 SQLite.....	1040
19. DISK QUOTA.....	1041
19.1 Linux setup.....	1041
19.2 FreeBSD setup.....	1041
19.3 Assign quota limits.....	1041
20. SHELLS.....	1042
20.1 bash.....	1042

20.2 tcsh.....	1043
21. SCRIPTING.....	1044
21.1 Basics.....	1044
21.2 Bourne script example.....	1045
21.3 Some sed commands.....	1045
21.4 Some useful commands.....	1045
22. PROGRAMMING.....	1046
22.1 C basics.....	1046
22.2 C example.....	1046
22.3 C++ basics.....	1046
22.4 C++ example.....	1046
22.5 Simple Makefile.....	1047
23. ONLINE HELP.....	1048
23.1 Documentation.....	1048
23.2 Other Unix/Linux references.....	1048
Полезные утилиты Gentoo Linux.....	1049
Ресурсы с документацией.....	1050
Ресурсы с программным обеспечением.....	1050
Каналы IRC (freenode.net).....	1050

Новое и измененное (changelog)

Версия 1.1 от 15.07.2007

1. По просьбе Mati_maniak (linuxforum.ru):
 - a) emerge --depclean world - исправлено на emerge --depclean;
 - b) --oneshot (-i) – исправлено на --oneshot.
2. Дополнительное форматирование текста.
3. Добавлен сборник статей: [Linux не для идиотов](#), автор: Артем Капитула (podashi, dalth & viking).
4. Добавлен: [список IRC каналов](#).
5. Добавил: [примеры моих конфигурационных файлов](#).

Версия 1.2 от 01.09.2007

1. Полностью переработан раздел по [обновлению Gentoo](#).
2. Произведено значительное форматирование текста.
3. Обновился раздел: [Полезные утилиты Gentoo Linux](#)
4. Полностью оформлен и дополнен [Handbook](#) !
5. По просьбе **inhab** (linuxforum.ru):
добавлена статья: [Настройка Samba в режиме PDC с использованием LDAP](#).
6. Добавлено: [Руководство по настройке GNOME](#).
7. Добавлено: [HOWTO fbsplash](#)

Версия 1.3 от 01.11.2007

1. Изменены цели написания данного сборника.
2. Появилось вступительное слово. :)
3. По просьбе **МАЗ** (<http://linuxforum.ru>) включена информация о Linux для новичков.
4. Сборник "Linux не для идиотов" перенесён в начало сборника !
5. Добавлен собственный перевод Wiki-статьи: [HOWTO Xorg and Fonts](#).
6. Оформлен новый раздел: "[Gentoo и GPRS](#)". Спасибо следующим авторам:
 - Nikoli (Linuxforum.ru).
 - Дмитрию а.к.а. doc (<http://doc.3dn.ru>).
 - Валентину Синицину.
7. Перенесены [XkbOptions](#).
8. Оформлен раздел: "[Gentoo и X.Org](#)"
9. Создан новый раздел: "[Файловые системы Linux](#)", состоящий из двух статей:
 - [Создание файловых систем](#)
 - [Linux и файловые системы: еще раз о проблеме выбора](#)Отдельное **СПАСИБО** Алексею Федорчуку (alv@linux-online.ru)
10. Заново импортирована: "[Настольная книга по безопасности Gentoo](#)" т.к. безопасность это очень важный вопрос. Теперь на русском языке.
11. Обновлены: "[Примеры конфигурационных файлов](#)"
12. Обновлен и дополнен: "[HOWTO Соединение с Internet](#)"

13. Продолжено форматирование текста.

Версия 1.4 от 01.02.2008

1. Добавлено: [Обращение к русскому сообществу Gentoo](#).
2. Дополнены цели написания данного сборника. ;)
3. В связи с выходом KDE 4.0.0, оформил раздел: [Gentoo и KDE](#)
 - добавлен собственный перевод статьи: [Gentoo and KDE 4.0](#)
 - добавлен материал: [Способы переключения раскладок клавиатуры в KDE \(3в1\)](#).
4. Так как мы используем Java, то решено оформить раздел: [Gentoo и Java](#)
 - заново переведена и дополнена статья: [Gentoo Java Guide](#);
 - переведена статья: [Gentoo Java Upgrade Guide](#);
5. Добавлена статья (автор: [MooSe](#)): [Соединение нескольких офисов в одну сеть с помощью OpenVPN](#).
6. Добавлена статья из русского wiki: [Шпаргалка по установке Gentoo](#).
7. Добавлено очень актуальное HOWTO: [HOWTO hal и устройства ввода](#).
8. Добавлено: [Ebuild HOWTO \(Англ\)](#).
9. Изменена статья о [XkbLayouts](#) в связи с переходом на hal версии 0.5.10 и возникшими проблемами с переключением раскладки клавиатуры.
10. Обновлено руководство: [Power Management Guide \(Англ.\)](#)
11. Обновлено и перенесена статья: [Windows-эквиваленты](#).
12. Исправлены многие орфографические ошибки.
13. Продолжено форматирование текста.
14. Обновлено: "[Примеры конфигурационных файлов](#)"

Версия 1.5 от 02.04.2008

1. Добавлены новые участники проекта:
 - Калинин Вадим а.к.а. **MooSe**;
 - Шевцов Константин а.к.а **integer || night_fever**.
2. Добавлен перевод [Power Management Guide: Управление питанием компьютера](#) (автор перевода: vah).
3. Добавлена статья: [Использование rpm-пакетов в Gentoo](#) (автор: [MooSe](#)).
4. Добавлена статья: [Настройка спутникового интернета в Gentoo](#) (автор: [tuxmaster](#)).
5. Добавлена статья: [Unix Toolbox](#).
6. Добавлена статья: [Советы пользователям Gentoo \(вариант 2\)](#) (автор Алексеев А.);
7. Добавлена статья: [Ещё раз о главном - жизнь после установки](#) (автор: gentoo).
8. Добавлена статья: [HOWTO ivman](#).
9. Добавлена статья: [Изучаем параметры gcc](#).
10. Создан новый раздел: [Gentoo и СУБД](#), а в нем добавлены:
 - [Руководство по MySQL для начинающих](#);
 - [PostgreSQL Guide](#).
11. Создан новый раздел: [Тюнинг Gentoo](#), а в нем добавлены:
 - [Ядро 2.6: Тонкости настройки](#);
 - [Тюнинг системы и запуска некоторых программ](#);

- [Тюнинг настроек программ из семейства Mozilla](#);
- перенесены:
- [Руководство по предварительному связыванию в Gentoo Linux](#);
 - [Тонкая настройка IDE дисков с помощью hdparm](#);
 - [Ускорение загрузки системы](#).
12. Создан новый раздел: [Gentoo и системы контроля версий](#), а в нем:
 - [HOWTO Subversion сервер при помощи Apache2 и WebDav](#).
 13. Добавлена статья: [Создание LiveCD дистрибутива на базе Gentoo Linux](#) (автор: [Pizgin](#)).
 14. Добавлена статья: [Что делать, если система «зависла»](#) (автор: [Perlovka](#)).
 15. [Добавлена статья: Hardened Gentoo](#) (автор: [powerman](#)).
 16. Обновлен перевод статьи: [Gentoo and KDE 4.0](#).
 17. Исправлены многие орфографические ошибки.
 18. Практически завершено форматирование текста.
 19. Обновлено: "[Примеры конфигурационных файлов](#)"

Составители

- **Батогов Е.В. а.к.а. JohnBat26** (ICQ:198710313; Jabber.ru/Yahoo/AIM/Google/MSN/Skype: JohnBat26)
e-mail: JohnBat26@gmail.com, JohnBat26@yandex.ru
occupation: JEE developer
о себе:
 - <http://picasaweb.google.ru/JohnBat26>
 - <http://www.odnoklassniki.ru/user/182870623>
 - <http://vkontakte.ru/id4283007>
 - <http://evgeniy-batogov.moikrug.ru/>
- **Шевцов Константин а.к.а integer || night_fever** (jabber: night_fever@jabber.ru; ICQ: 496480428)
e-mail: gentoo.integer@gmail.com
- **Калинников Вадим Валерьевич а.к.а. MooSE** (jabber: moose1@jabber.ru)
e-mail: bulvinkl@gmail.com
- **Конахин Э.В. а.к.а. BadEd** (ICQ: 757727756)
e-mail: baded@mail.ru
о себе:
 - <http://www.odnoklassniki.ru/user/265150462>
- **Киселев М.А. а.к.а. Kismih05** (ICQ:77833996); Yahoo/AIM: kismih05)
e-mail: kismih05@yahoo.com
- **Dmeshr**
e-mail: dmeshr@gmail.com

По материалам сайтов:

- <http://gentoo.ru>
- <http://gentoo.org>
- <http://ru.gentoo-wiki.com>
- <http://gentoo.blog.ru>
- <http://www.rugentoo.org>
- <http://linuxforum.ru>
- <http://myfotomx.com/dalth/linuxbook.odt>
- <http://linuxshop.ru>
- <http://citkit.ru>
- <http://rus-linux.net>
- <http://lxj.endofinternet.net>
- <http://www.habrahabr.ru/blog/linux>
- <http://perlovka.homelinux.net>

Версия: 1.5

Последняя редакция: 02.04.2008

Лицензия: GPL v2

Создано с помощью: OpenOffice Writer 2.4.0

Цели данного сборника

- *Избавить от необходимости серфинга по различным ресурсам в поисках информации о Linux и Gentoo в частности.*
- *Увеличить количество пользователей Gentoo Linux в России.*
- *Упростить освоение Gentoo Linux, тем самым увеличить его популярность.*
- *Создать лучшую документацию по Gentoo Linux на русском языке.*

Пожелания, участие в написании и критика приветствуются. :)

Просьба при копировании или дополнении данного сборника, согласовывать с:

1. Авторами статей.
2. Составителями данного сборника (см. вверху страницы).

Всем авторам статей: ОГРОМНОЕ СПАСИБО ЗА ВАШ ТРУД !!!

Обращение к русскому сообществу Gentoo

Дорогие и уважаемые пользователи Gentoo Linux, вся наша команда по созданию данного сборника желает Вам дальнейшего совершенствования в области Linux. Для дальнейшего улучшения книги, мы предлагаем всех Вам написать на наши e-mail:

- johnbat26@gmail.com;
- gentoo.integer@gmail.com;
- bulvinkl@gmail.com;
- baded@mail.ru;
- kismih05@yahoo.com;
- dmeshr@gmail.com.

ОТЗЫВЫ О ТОМ ЧТО:

- понравилось;
- не понравилось;
- чем вы могли бы помочь.

Заранее спасибо.

Нам необходимы люди:

- по сбору интересной информации, которую можно и нужно опубликовать в книге;
- переводчики;
- знающие толк в форматировании и стилистике документов.

Просьба к администраторам <http://gentoo.ru> опубликовать наши переводы статей на <http://gentoo.ru> и <http://gentoo.org>.

Чем больше отзывов, тем быстрее будут выходить новые версии книг ! ;)

Еще раз всем удачи на пути освоения, а затем и наслаждения Gentoo Linux !

Вступительное слово

Автор: Батогов Е.В. а.к.а JohnBat26

Добро пожаловать всем, кто решил вступить на путь освоения операционной системы Linux (и Gentoo в особенности). Немного забегаая вперед, хочется заметить, что путь предстоит не легкий, но очень интересный. Главное помните одну известную истину: "Лучше день подождать, а потом за 5 минут долететь!"

Расскажу немного из своего опыта использования Linux.

Когда 3 года назад мне надоело переустанавливать Windows по два и более раза в год, я стал искать альтернативу. Кто-то скажет сейчас, что мол я не умею использовать Windows, сразу отвечу, что работал в ней около 5 лет. Так как я являюсь человеком, который не может просто так сидеть и не проводить эксперименты с системой, то устанавливал более 200 программ, и это делало Windows очень медленной и нежизнеспособной (Сейчас у меня в Gentoo установлено > 1000 пакетов :)). Далее, в Windows, автоматически возникали проблемы обновления всего этого ПО. И это еще не говоря о вирусах...

Сначала я поставил OpenSuSE. В принципе все нравилось, НО это до необходимости совершения таких действий как, например, смена ядра, установка драйверов, обновление системы (хотя Smart в принципе, справлялся с этим).

Установка программ из RPM вызывала напряжение нервной системы, особенно когда не было хорошего интернета. Ох уж эти зависимости.... :-(

Так как я являюсь человеком, который не остановится, пока не получит систему, близкую к идеальной, то мой взгляд, наконец-то, пал на Gentoo. Спустя год я уже установил Gentoo на 10 машин, из которых половина ноутбуки (Dell, Asus, Acer, Sony VAIO)! Процесс установки действительно захватывает тем, что ты полностью понимаешь, что полностью управляешь системой.

Gentoo поразил меня следующим:

- очень быстрая скорость работы;
- ИМНО, лучшая системы управления ПО — portage;
- очень много документации, в том числе и на русском языке;
- никаких проблем с обновлением всей системы, включая ядро, gcc, glibc и т.п.;
- отличное сообщество пользователей;
- абсолютная контролируемость системы;
- никаких лишних пакетов.

Еще ни одна ОС не вдохновляла меня на написание книги, пусть она и составлена из трудов многих хороших людей. В нашей Великой России такие сборники должны быть очень актуальны, т.к. не у всех есть постоянный доступ к сети (сам работал через GPRS в течение двух лет).

В заключении хочется пожелать всем удачи на пути освоения Gentoo Linux. Как говорится: "Дорогу осилит идущий !" Вперед !!!

Вводная информация о Linux

UNIX

Ссылка на оригинал: <http://ru.wikipedia.org/wiki/UNIX>

С версии: 1.3

UNIX (читается юникс) — группа переносимых, многозадачных и многопользовательских операционных систем.

Первая система UNIX была разработана в конце 1960-х — начале 1970-х годов в подразделении Bell Labs компании AT&T. С тех пор было создано большое количество различных UNIX-систем. Юридически лишь некоторые из них имеют полное право называться «UNIX»; остальные же, хотя и используют сходные концепции и технологии, объединяются термином «UNIX-подобные» (англ. Unix-like). Для краткости в данной статье под UNIX-системами подразумеваются как истинные UNIX, так и UNIX-подобные ОС.

Некоторые отличительные признаки UNIX-систем включают в себя:

- использование простых текстовых файлов для настройки и управления системой;
- широкое применение командной строки;
- представление устройств и некоторых средств межпроцессного взаимодействия как файлов;
- использование конвейеров из нескольких программ, каждая из которых выполняет одну задачу.

В настоящее время UNIX используются в основном на серверах, а также как встроенные системы для различного оборудования. На рынке ОС для рабочих станций и домашнего применения UNIX уступили другим операционным системам, в первую очередь Microsoft Windows. UNIX-системы имеют большую историческую важность, поскольку благодаря им распространились некоторые популярные сегодня концепции и подходы в области ОС и программного обеспечения.

История

Предшественники

В 1957 году в Bell Labs была начата работа по созданию операционной системы для собственных нужд. Под руководством Виктора Высотского (русского по происхождению) была создана система BESYS. Впоследствии он возглавил проект Multics, а затем стал главой информационного подразделения Bell Labs.

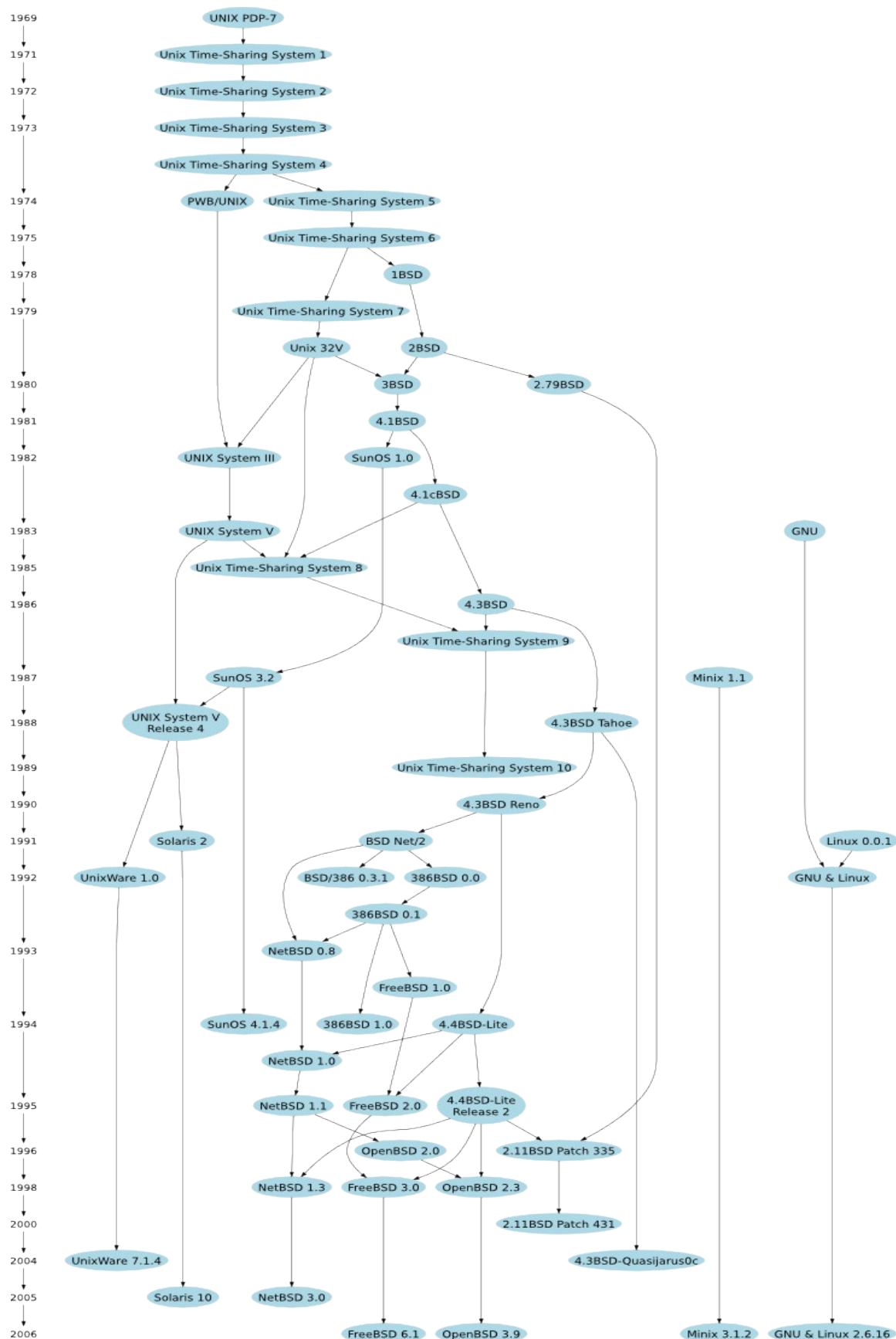


Рис 1. Генеалогическое древо UNIX-систем.

В 1964 году появились компьютеры третьего поколения, для которых возможности BESYS уже не подходили. Высотский и его коллеги приняли решение не разрабатывать новую собственную операционную систему, а подключиться к совместному проекту General Electric и Массачусетского технологического института Multics. Телекоммуникационный гигант AT&T, в состав которого входили Bell Labs, оказал проекту существенную поддержку, но в 1969 году вышел из проекта, поскольку он не приносил никаких финансовых выгод.

Первые UNIX

Первоначально UNIX была разработана в конце 1960-х годов сотрудниками Bell Labs, в 1969 году Кен Томпсон, стремясь реализовать идеи, что были положены в основу MULTICS, но на более скромном аппаратном обеспечении (DEC PDP-7), написал первую версию новой операционной системы, а Брайан Керниган придумал для неё название — UNICS (UNIpIplexed Information and Computing System) — в противовес MULTICS (MULTIpIplexed Information and Computing Service). Позже это название сократилось до UNIX. В 1971 году вышла версия для PDP-11, наиболее успешного семейства миникомпьютеров 1970-х (в СССР оно известно как СМ ЭВМ). Эта версия получила название «первая редакция» (Edition 1) и была первой официальной версией. Системное время все реализации UNIX отсчитывают с 1 января 1970.

Первые версии UNIX были написаны на ассемблере и не имели встроенного компилятора с языка высокого уровня. Примерно в 1969 году Кен Томпсон при содействии Дениса Ритчи разработал и реализовал язык Би (B), представлявший собой упрощённый (для реализации на миникомпьютерах) вариант разработанного в 1966 языка BCPL. Би, как и BCPL, был интерпретируемым языком. В 1969—1973 годах на основе Би был разработан компилируемый язык, получивший название Си (C).

В 1973 году вышла третья редакция UNIX, со встроенным компилятором с Си. 15 октября того же года появилась четвёртая редакция, с переписанным на Си системным ядром (в духе системы Multics, также написанной на языке высокого уровня (ПЛ/1)), а в 1975 — пятая редакция, полностью переписанная на Си. С 1974 года UNIX стал бесплатно распространяться среди университетов и академических учреждений. С 1975 года началось появление новых версий, разработанных за пределами Bell Labs, и рост популярности системы.

В том же 1975 году вышла шестая редакция, известная по широко разошедшимся комментариям Джона Лайонса (Lions' Commentary on UNIX 6th Edition, with Source Code, [1], [2]). К 1978 г. система была установлена более чем на 600 машинах, прежде всего, в университетах. Седьмая редакция была последней единой версией UNIX. Именно в ней появился близкий к современному интерпретатор командной строки Bourne shell.ую очередь Кеном Томпсоном, Денисом Ритчи и Дугласом МакИлроем.

Раскол

С 1978 года начинает свою историю BSD UNIX, созданный в университете Беркли. Его первая версия, была основана на шестой редакции. В 1979 выпущена новая

версия, названная 3BSD, основанная на седьмой редакции. BSD поддерживал такие полезные свойства, как виртуальную память и замещение страниц по требованию. Автором BSD был Билл Джой.

В начале 1980-х компания AT&T, которой принадлежали Bell Labs, осознала ценность UNIX и начала создание коммерческой версии UNIX. Эта версия, поступившая в продажу в 1982 году, носила название UNIX System III и была основана на седьмой версии системы.

Важной причиной раскола UNIX стала реализация в 1980 г. стека протоколов TCP/IP. До этого межмашинное взаимодействие в UNIX пребывало в зачаточном состоянии — наиболее существенным способом связи был UUCP (средство копирования файлов из одной UNIX-системы в другую, изначально работавшее по телефонным сетям с помощью модемов).

Было предложено два интерфейса программирования сетевых приложений: Berkley sockets и интерфейс транспортного уровня TLI (англ. Transport Layer Interface). Интерфейс Berkley sockets был разработан в университете Беркли и использовал стек протоколов TCP/IP, разработанный там же. TLI был создан AT&T в соответствии с определением транспортного уровня модели OSI и впервые появился в системе System V версии 3. Хотя эта версия содержала TLI и потоки, первоначально в ней не было реализации TCP/IP или других сетевых протоколов, но подобные реализации предоставлялись сторонними фирмами. Реализация TCP/IP официально и окончательно была включена в базовую поставку System V версии 4. Это, как и другие соображения (по большей части, рыночные), вызвало окончательное размежевание между двумя ветвями UNIX — BSD (университета Беркли) и System V (коммерческая версия от AT&T). Впоследствии, многие компании, лицензировав System V у AT&T, разработали собственные коммерческие разновидности UNIX, такие, как AIX, HP-UX, IRIX, Solaris.

В середине 1983 года была выпущена версия 4.2BSD, поддерживающая работу в сетях Ethernet и Agranet. Система стала весьма популярной. Между 1983 и 1990 в BSD были добавлено много новых возможностей, таких как отладчик ядра, сетевая файловая система NFS, виртуальная файловая система VFS, и существенно улучшены возможности работы с файловыми сетями.

Тем временем AT&T выпускала новые версии своей системы, названной System V. В 1983 была выпущена версия 1 (SVR1 — System V Release 1), включавшая полноэкранный текстовый редактор vi, библиотеку curses, буферизацию ввода-вывода, кеширование inode. Версия 2 (SVR2), выпущенная в 1984, реализовывала монополярный доступ к файлам (file locking), доступ к страницам по требованию (demand paging), копирование при записи (copy-on-write). Версия 3 вышла в 1987 году и включала, среди прочего, TLI, а также систему поддержки удалённых файловых систем RFS. Версия 4 (SVR4), разработанная в сотрудничестве с фирмой Sun и вышедшая 18 октября 1988, поддерживала многие возможности BSD, в частности TCP/IP, сокеты, новый командный интерпретатор csh. Кроме того, там было много других добавлений, таких как символические ссылки, командный интерпретатор ksh, сетевая файловая система NFS (заимствованная у SunOS) и т. д.

Современные реализации UNIX, как правило, не являются системами V или BSD в чистом виде. Они реализуют возможности как System V, так и BSD.

Свободные UNIX-подобные операционные системы

В 1983 году Ричард Столлман объявил о создании проекта GNU — попытки создания свободной UNIX-подобной операционной системы с нуля, без использования оригинального исходного кода. Большая часть программного обеспечения, разработанного в рамках данного проекта, — такого, как GNU toolchain, Glibc (стандартная библиотека языка Си) и Coreutils — играет ключевую роль в других свободных операционных системах. Однако работы по созданию замены для ядра UNIX, необходимые для полного выполнения задач GNU, продвигались крайне медленно. В настоящее время GNU Hurd — попытка создать современное ядро на основе микроядерной архитектуры Mach — всё ещё далека от завершения.

В 1991 году, когда Линус Торвалдс опубликовал ядро Linux и привлёк помощников, использование инструментов, разработанных в рамках проекта GNU, было очевидным выбором. Операционная система GNU и ядро Linux вместе составляют ОС, известную, как GNU/Linux. Дистрибутивы этой системы (такие как Red Hat и Debian), включающие ядро, утилиты GNU и дополнительное программное обеспечение стали популярными как среди любителей, так и среди представителей бизнеса.

В результате урегулирования юридического дела, возбуждённого UNIX Systems Laboratories против университета Беркли и Berkeley Software Design Inc., было установлено, что университет может распространять BSD UNIX, в том числе и бесплатно. После этого были возобновлены эксперименты, связанные с BSD-версией UNIX. Вскоре разработка дистрибутива BSD была продолжена в нескольких направлениях одновременно, что привело к появлению проектов, известных как FreeBSD, NetBSD, OpenBSD, TrustedBSD и DragonFlyBSD.

В настоящий момент GNU/Linux и представители семейства BSD быстро отвоевывают рынок у коммерческих UNIX-систем и одновременно проникают как на настольные компьютеры конечных пользователей, так и на мобильные и встраиваемые системы. Одним из свидетельств данного успеха служит тот факт, что, когда фирма Apple искала основу для своей новой операционной системы, она выбрала NEXTSTEP — операционную систему со свободно распространяемым ядром, разработанную фирмой NeXT и переименованную в Darwin после приобретения фирмой Apple. Данная система относится к семейству BSD и основана на ядре Mach. Применение Darwin BSD UNIX в Mac OS X делает его одной из наиболее широко используемых версий UNIX.

Современность

После разделения компании AT&T, товарный знак UNIX и права на оригинальный исходный код неоднократно меняли владельцев, в частности, длительное время принадлежали компании Novell.

В 1993 году Novell передала права на товарный знак и на сертификацию программного обеспечения на соответствие этому знаку консорциуму X/Open, который затем объединился с Open Software Foundation, образовав консорциум The Open Group. Он объединяет ведущие компьютерные корпорации и государственные организации, в том числе IBM, Hewlett-Packard, Sun, NASA и многие другие. Консорциум занимается разработкой открытых стандартов в области операционных систем, самым важным из которых является Single UNIX Specification, ранее известный как POSIX. С точки зрения The Open Group,

название UNIX могут носить только системы, прошедшие сертификацию на соответствие Single UNIX Specification.

В 1995 году Novell продала права на существующие лицензии и дальнейшую разработку System V компании Santa Cruz Operation. В 2000 году Santa Cruz Operation продала свой UNIX-бизнес компании Caldera, которая затем была переименована в SCO Group. Хотя это название похоже на аббревиатуру SCO, используемую Santa Cruz Operation, это две разные компании.

SCO Group заявила, что она также обладает правами на исходный код UNIX и развернула кампанию против различных пользователей и поставщиков UNIX-подобных систем, требуя выплаты лицензионных отчислений. Однако Novell утверждает, что права на исходный код не были переданы Santa Cruz Operation и, таким образом, не перешли к SCO Group, а остаются у Novell, что и подтвердил вердикт суда.

Влияние UNIX

Идеи, заложенные в основу UNIX, оказали огромное влияние на развитие компьютерных операционных систем. В настоящее время UNIX-системы признаны одними из самых исторически важных ОС.

Как и Multics, UNIX была написана на языке высокого уровня, а не на ассемблере (доминировавшем в то время).

Она содержала значительно упрощённую, по сравнению с современными ей операционными системами, файловую модель. Файловая система включала как службы, так и устройства (такие как принтеры, терминалы и жёсткие диски) и предоставляла внешне единообразный интерфейс к ним, но дополнительные механизмы работы с устройствами (такие как IOCTL и биты доступа) не вписывались в простую модель «поток байтов».

UNIX популяризовала предложенную в Multics идею иерархической файловой системы с произвольной глубиной вложенности. Другие операционные системы того времени позволяли разбивать дисковое пространство на каталоги или разделы, но число уровней вложенности было фиксировано и, зачастую, уровень вложенности был только один. Позднее все основные фирменные операционные системы обрели возможность создания рекурсивных подкаталогов, также заимствованную из Multics.

То, что интерпретатор команд стал просто одной из пользовательских программ, а в качестве дополнительных команд выступают отдельные программы, является ещё одной инновацией Multics, популяризированной UNIX. Язык командной оболочки UNIX используется пользователем как для интерактивной работы, так и для написания скриптов, то есть не существует отдельного языка описания заданий, как, например, в системе JCL фирмы IBM. Так как оболочка и команды операционной системы являются обычными программами, пользователь может выбирать их в соответствии со своими предпочтениями, или даже написать собственную оболочку. Наконец, новые команды можно добавлять к системе без перекомпиляции ядра. Новый, предложенный в командной строке UNIX, способ создания цепочек программ, последовательно обрабатывающих данные, способствовал использованию параллельной обработки данных.

Существенными особенностями UNIX были полная ориентация на текстовый ввод-вывод и предположение, что размер машинного слова кратен восьми битам. Первоначально в UNIX не было даже редакторов двоичных файлов — система полностью конфигурировалась с помощью текстовых команд. Наибольшей и

наименьшей единицей ввода-вывода служил текстовый байт, что полностью отличало ввод-вывод UNIX от ввода-вывода других операционных систем, ориентированного на работу с записями. Ориентация на использование текста для представления всего, что только можно, сделала полезными т. н. конвейеры (англ. pipelines). Ориентация на текстовый восьмибитный байт сделала UNIX более масштабируемой и переносимой, чем другие операционные системы. Со временем текстовые приложения одержали победу и в других областях, например, на уровне сетевых протоколов, таких как Telnet, FTP, SMTP, HTTP и других.

UNIX способствовала широкому распространению регулярных выражений, которые были впервые реализованы в текстовом редакторе ed для UNIX. Возможности, предоставляемые UNIX-программам, стали основой стандартных интерфейсов операционных систем (POSIX).

Широко используемый в системном программировании язык Си, созданный изначально для разработки UNIX, превзошёл UNIX по популярности. Си был первым «веротерпимым» языком, который не пытался навязать программисту тот или иной стиль программирования. Си был первым высокоуровневым языком, предоставляющим доступ ко всем возможностям процессора, таким как ссылки, таблицы, битовые сдвиги, приращения и т. п. С другой стороны, свобода Си приводила к ошибкам переполнения буфера в таких функциях стандартной библиотеки Си, как gets и scanf. Результатом стали многие печально известные уязвимости, например, та, что эксплуатировалась в знаменитом черве Морриса.

Первые разработчики UNIX способствовали внедрению принципов модульного программирования и повторного использования в инженерную практику.

UNIX предоставлял возможность использования протоколов TCP/IP на сравнительно недорогих компьютерах, что привело к быстрому росту Интернета. Это, в свою очередь, способствовало быстрому обнаружению нескольких крупных уязвимостей в системе безопасности, архитектуре и системных утилитах UNIX.

Со временем ведущие разработчики UNIX разработали культурные нормы разработки программного обеспечения, которые стали столь же важны, как и сам UNIX (см. Философия UNIX).

Стандарты

Большое количество разных вариантов системы UNIX привело к необходимости стандартизовать её, чтобы упростить переносимость приложений и избавить пользователя от необходимости изучать особенности каждой разновидности UNIX. С этой целью ещё в 1980 была создана пользовательская группа /usr/group. Самые первые стандарты были разработаны в 1984—1985 гг. В настоящее время наиболее важными являются следующие стандарты:

- **POSIX 1003.1-1988**, определяющий программный интерфейс приложений (API, Application Programming Interface). Он используется не только в UNIX, но и в других операционных системах. В 1990 он был принят институтом IEEE как IEEE 1003.1-1990.
- **POSIX 1003.2-1992**, определяющий поведение утилит, в том числе командного интерпретатора.
- **POSIX 1003.1b-1993**, дополняющий POSIX 1003.1-1988. Определяет поддержку систем реального времени.
- **POSIX 1003.1c-1995**, дополняющий POSIX 1003.1-1988. Определяет нити

(threads), известные также как pthreads.

Все стандарты POSIX собраны в документе IEEE 1003.

В начале 1990-х годов The Open Group предложила другой, похожий на POSIX стандарт — Common API Specification, или Spec 1170. Стандарт приобрёл большую популярность, чем POSIX, поскольку был доступен бесплатно, в то время как IEEE требовало немалую плату за доступ к своему стандарту.

В 1998 году были начаты работы по объединению данных стандартов. Благодаря этому в настоящее время данные стандарты почти идентичны. Совместный стандарт называется Single UNIX Specification Version 3 и доступен бесплатно в интернете [3].

В целях совместимости несколько создателей UNIX-систем предложили использовать ELF-формат систем SVR4 для двоичных и объектных файлов. Единый формат полностью обеспечивает соответствие двоичных файлов в рамках одной компьютерной архитектуры.

Структура каталогов некоторых систем, в частности, GNU/Linux, определена в стандарте Filesystem Hierarchy Standard. Однако во многих отношениях этот тип стандарта является спорным, и он, даже внутри сообщества GNU/Linux, далеко не универсален.

Канонические команды UNIX

- Создание и навигация по файлам и каталогам: ls, mv, rm, cp, ln, CD, pwd, mkdir, rmdir, find, du, df.
- Просмотр и редактирование файлов: touch, more, less, ed, ex, vi, emacs.
- Обработка текста: echo, cat, grep, sort, uniq, sed, awk, tee, head, tail, cut, tr, split, printf.
- Сравнение файлов: comm, cmp, diff, patch.
- Разнообразные утилиты командного интерпретатора: yes, test, xargs, expr.
- Системное администрирование: chmod, chown, ps, su, w, who, df, mount, umount.
- Коммуникации: mail, telnet, ftp, finger, ssh.
- Командные оболочки: sh, bash, csh, ksh, tcsh, zsh.
- Работа с исходным кодом и объектным кодом: cc, gcc, ld, nm, yacc, bison, lex, flex, ar, ranlib, make.
- Сжатие и архивация: compress, uncompress, gzip, gunzip, tar.
- Работа с двоичными файлами: od, strings.

Ниже приведён список 60 команд из раздела 1 первой версии UNIX:

ar, as, b, bas, bcd, boot, cat, chdir, check, chmod, chown, cmp, cp, date, db, dbppt, dc, df, dsw, dtt, du, ed, find, for, form, hup, lbppt, ld, ln, ls, mail, mesg, mkdir, mkfs, mount, mv, nm, od, pr, rew, rkd, rkf, rkl, rm, rmdir, roff, sdate, sh, stat, strip, su, sum, tap, tm, tty, type, un, wc, who, write

GNU/Linux

Ссылка на оригинал: <http://ru.wikipedia.org/wiki/Линукс>

С версии: 1.3

GNU/Linux (произносится «гну слэш линукс») — свободная UNIX-подобная операционная система. Она основана на системных программах, разработанных в рамках проекта GNU, и на ядре Linux. Зачастую, по историческим причинам и для краткости, эту систему называют просто «Linux».

GNU/Linux работает на PC-совместимых системах семейства Intel x86, а также на IA-64, AMD64, PowerPC, ARM и многих других.

К операционной системе GNU/Linux также часто относят программы, дополняющие эту операционную систему, и прикладные программы, делающие её полноценной многофункциональной операционной средой.

В отличие от большинства других операционных систем, GNU/Linux не имеет единой «официальной» комплектации. Вместо этого GNU/Linux поставляется в большом количестве так называемых дистрибутивов, в которых программы GNU соединяются с ядром Linux и другими программами. Наиболее известными дистрибутивами GNU/Linux являются Slackware, Red Hat, Fedora, Mandriva, SuSE, Debian, Gentoo, Ubuntu. Из дистрибутивов российских разработчиков наиболее известны ALT Linux и ASPLinux.

В отличие от Microsoft Windows, Mac OS (Mac OS X) и коммерческих UNIX-подобных систем, GNU/Linux не имеет географического центра разработки. Нет и организации, которая владела бы этой системой; нет даже единого координационного центра. Программы для GNU/Linux — результат работы тысяч проектов. Некоторые из этих проектов централизованы, некоторые сосредоточены в фирмах, но большинство объединяют программистов со всего света, которые знакомы только по переписке. Создать свой проект или присоединиться к уже существующему может любой и, в случае успеха, результаты работы станут известны миллионам пользователей. Пользователи принимают участие в тестировании свободных программ, общаются с разработчиками напрямую, что позволяет быстро находить и исправлять ошибки и реализовывать новые возможности.

Именно такая гибкая и динамичная система разработки, невозможная для проектов с закрытым кодом, определяет исключительную экономическую эффективность GNU/Linux. Низкая стоимость свободных разработок, отлаженные механизмы тестирования и распространения, привлечение людей из разных стран, обладающих разным видением проблем, защита кода лицензией GPL — всё это стало причиной успеха свободных программ. [1]

Конечно, такая высокая эффективность разработки не могла не заинтересовать крупные фирмы, которые стали открывать свои проекты. Так появились Mozilla (Netscape, AOL), OpenOffice.org (Sun), свободный клон Interbase (Borland) — Firebird, SAP DB (SAP). IBM способствовала переносу GNU/Linux на свои мейнфреймы.

С другой стороны, открытый код значительно снижает себестоимость разработки закрытых систем для GNU/Linux и позволяет снизить цену решения для пользователя. Вот почему GNU/Linux стала платформой, часто рекомендуемой для таких продуктов, как Oracle Database, DB2, Informix, SyBase, SAP R3, Domino. Сообщество GNU/Linux поддерживает связь посредством групп пользователей Linux.

Именно такая гибкая и динамичная система разработки, невозможная для проектов с закрытым кодом, определяет исключительную экономическую эффективность GNU/Linux. Низкая стоимость свободных разработок, отлаженные механизмы тестирования и распространения, привлечение людей из разных стран, обладающих разным видением проблем, защита кода лицензией GPL — всё это стало причиной успеха свободных программ. [1]

Конечно, такая высокая эффективность разработки не могла не заинтересовать крупные фирмы, которые стали открывать свои проекты. Так появились Mozilla (Netscape, AOL), OpenOffice.org (Sun), свободный клон Interbase (Borland) — Firebird, SAP DB (SAP). IBM способствовала переносу GNU/Linux на свои мейнфреймы.

С другой стороны, открытый код значительно снижает себестоимость разработки закрытых систем для GNU/Linux и позволяет снизить цену решения для пользователя. Вот почему GNU/Linux стала платформой, часто рекомендуемой для таких продуктов, как Oracle Database, DB2, Informix, SyBase, SAP R3, Domino. Сообщество GNU/Linux поддерживает связь посредством групп пользователей Linux.

18

Linux (ядро)

Ссылка на оригинал: [http://ru.wikipedia.org/wiki/Linux_\(ядро\)](http://ru.wikipedia.org/wiki/Linux_(ядро))

С версии: 1.3

Linux (Линукс) — ядро операционной системы, разработка которого была начата финским студентом Линусом Торвальдсом в 1991 году.

В большинстве своём код написан на Си с некоторыми расширениями GNU C и на ассемблере (с использованием синтаксиса GNU Assembler AT&T).

Распространяется свободно на условиях GNU General Public License.

Торговая марка Linux зарегистрирована на Линуса Торвальдса.

История

Начало проекту было положено в 1991 году с публикации сообщения в новостной группе Usenet comp.os.minix] следующего содержания[1]:

Привет всем, кто использует миникс — Я делаю (бесплатную) операционную систему (всего лишь хобби, не будет большой и профессиональной как gnu) для клонов 386 (486) AT...

К тому времени проект GNU уже создал множество составляющих для свободной операционной системы, но их ядро GNU Hurd ещё не было готово. BSD-системы в то время не могли быть использованы по юридическим причинам, связанным с использовавшимися лицензиями. Поэтому пустующее место ядра для свободной ОС занял Linux и, несмотря на ограниченную функциональность ранних версий, привлёк к себе множество разработчиков и пользователей.

Linux — это название только ядра, а не операционной системы. Часто системы, основанные на ядре Linux, называют просто Linux, но большинство из них на самом деле называются GNU/Linux, так как состоят из ядра Linux и множества системных библиотек и программ, написанных в рамках проекта GNU.

Хронология

* Апрель 1991 — 21-летний Линус Торвальдс начал работу над некоторыми механизмами операционной системы. Он начал с эмулятора терминала и планировщика задач.

* 25 августа 1991 — Линус поместил следующее сообщение (русский перевод оригинального сообщения[1] на английском):

*From: torvalds@klaava.Helsinki.Fi (Линус Бенедикт Торвальдс)
Newsgroups: comp.os.minix
Subject: Маленький опрос о моей новой операционной системе
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.Fi>
Date: 25 Aug 91 20:57:08 GMT
Organization: Хельсинский Университет*

Привет всем, кто использует миникс - Я делаю (бесплатную) операционную систему

(всега лишь хобби, не будет большой и профессиональной как gnu) для клонов 386 (486) AT. Она ваялась с апреля, и скоро будет готова. Я хочу отзывов о том, что людям нравится/не нравится в миниксе, ибо моя система на неё похожа(такое же устройство файловой системы(по практическим соображениям) среди всего прочего).

Я уже включил bash (1.08) и gcc (1.40), и похоже всё работает. Это значит, что что-то полезное появится через несколько месяцев, и я хотел бы узнать, чего люди хотят. Любые советы принимаются, но я не обещаю, что всё исполню :-)

Линус (torvalds@kruuna.helsinki.fi)

PS. Да, у неё никакого миниксового кода, и многозадачная фс. Она НЕ переносима (применяет переключение задач 386-го, итп.), и скорее всего будет поддерживать только AT-винчестеры, так как это всё, что у меня есть :-)

- 17 сентября 1991 — Linux версии 0.01. (10 239 строк кода);
- 5 октября 1991 — Linux версии 0.02 [1](англ.);
- Декабрь 1991 — Linux версии 0.11. Это была первая версия Linux, на которой можно было собрать Linux из исходных кодов;
- 19 января 1992 — Первое сообщение в группе новостей alt.os.linux(англ.);
- 31 марта 1992 — Создана группа новостей comp.os.linux(англ.);
- Апрель 1992 — Linux версии 0.96, на котором стало возможно запустить графический сервер X Window System;
- Весь 1993 и начало 1994 — 15 тестовых релизов версии 0.99.* (в июле 1993 введено понятие BogoMips);
- 14 марта 1994 — Linux версии 1.0.0 (176 250 строк кода);
- Март 1995 — Linux версии 1.2.0 (310 950 строк кода);
- 9 мая 1996 — Выбран символ Linux — пингвин Tux;
- 9 июня 1996 — Linux версии 2.0.0 (777 956 строк кода);
- 25 января 1999 — Linux версии 2.2.0, изначально довольно недоработанный (1 800 847 строк кода);
- 4 января 2001 — Linux версии 2.4.0 (3 377 902 строк кода);
- 18 декабря 2003 — Linux версии 2.6.0 (5 929 913 строк кода).

Версии

Торвальдс продолжает выпускать новые версии ядра, объединяя изменения, вносимые другими программистами, и внося свои. В дополнение к официальным версиям ядра существуют альтернативные ветки, которые могут быть взяты из различных источников. Как правило, разработчики дистрибутивов GNU/Linux поддерживают свои собственные версии ядра Linux, например, включая в них драйвера устройств, которые ещё не включены в официальную версию.

Нумерация версий

Номер версии ядра Linux в настоящее время содержит четыре числа, следуя недавнему изменению в долго используемой до этого политике схемы версий,

основанной на трёх числах. Для иллюстрации допустим, что номер версии составлен таким образом: A.B.C[D] (например 2.2.1, 2.4.13 или 2.6.12.3).

* Число A обозначает версию ядра. Оно изменяется наименее часто и только тогда, когда вносятся значительные изменения в код и концепцию ядра. Оно изменялось дважды в истории ядра: в 1994 (версия 1.0) и в 1996 (версия 2.0).

* Число B обозначает старшую версию ревизии ядра. Чётные числа обозначают стабильные ревизии, то есть те, которые предназначены для промышленного использования, такие как 1.2, 2.4 или 2.6. Нечётные числа обозначают ревизии для разработчиков, такие как 1.1 или 2.5. Они предназначены для тестирования новых улучшений и драйверов до тех пор, пока они не станут достаточно стабильными для того, чтобы быть включёнными в стабильный выпуск.

* Число C обозначает младшую версию ревизии ядра. В старой трёхчисловой схеме нумерации, оно изменялось тогда, когда в ядро включались заплатки связанные с безопасностью, исправления ошибок, новые улучшения или драйвера. С новой политикой нумерации, однако, оно изменяется только тогда, когда вносятся новые драйвера или улучшения; небольшие исправления поддерживаются числом D.

* Число D впервые появилось в случае, когда смертельная ошибка, которая требовала незамедлительного исправления, была обнаружена в коде ядра 2.6.8, связанного с NFS. Однако, было недостаточно других изменений, для того чтобы это послужило причиной для выпуска новой младшей ревизии (которой должна была стать 2.6.9). Поэтому была выпущена версия 2.6.8.1 с единственным исправлением в виде исправления для этой ошибки. С ядра 2.6.11, эта нумерация была адаптирована в качестве новой официальной политики версий. Исправления ошибок и заплатки безопасности теперь управляются с помощью четвёртого числа, тогда как большие изменения выполняются в изменениях младшей версии ревизии ядра (число C).

Поддержка

В то время как Торвальдс продолжает выпускать новые экспериментальные версии, руководство «старыми» стабильными версиями передаётся другим лицам:

Версии	Сопроводители
2.0	Дэвид Виенхал
2.2	Марк-Кристиан Петерсон (раньше Элан Кокс)
2.4	Марчело Тозатти
2.6	Эндрю Мортон / Линус Торвальдс

Другими программистами ядра Linux являются Роберт Лав и Инго Молнар. (См. Список сопроводителей Linux (англ.)).

Стабильные версии

- Версия 1.0 в марте 1994 — поддерживала только однопроцессорные i386-машины.
- Версия 1.2 в марте 1995 — добавлена поддержка процессоров Alpha, SPARC и MIPS.
- Версия 2.0 в июне 1996 — добавлена поддержка других процессоров, а также многопроцессорных систем.
- Версия 2.2 в январе 1999 — [2](англ.).
- Версия 2.4 в январе 2001 — добавлена поддержка ISA Plug and Play, процессоров PA-RISC, шин USB и PC-Card (PCMCIA). Поддержка для процессоров Axis Communications ETRAX CRIS и файловой системы InterMezzo были добавлены чуть позже. [3](англ.)
- Версия 2.6 от 17 декабря 2003:
 - встроен uClinux (для микроконтроллеров);
 - добавлена поддержка для процессоров Hitachi серии H8/300, NEC v850, процессоры для встроенных систем Motorola m68k, новая архитектура доступа к памяти NUMA, поддержка NCR Voyager, технологии Intel hyperthreading и PAE;
 - добавлено:
 - + поддержка файловой системы XFS SGI;
 - + улучшена поддержка APIC;
 - + увеличено максимальное количество пользователей и групп с 65 000 до более 4 млрд;
 - + увеличено максимальное количество процессов с 32 000 до 1 млрд;
 - + увеличено максимальное количество типов устройств (major device) с 255 до 4095 и максимальное количество устройств каждого типа (minor device) с 255 до более миллиона;
 - + улучшена поддержка 64-битных систем и поддержка файловых систем размером более 16 Тбайт;
 - + улучшено время реакции для процессов реального времени;
 - + переписана реализация потоков под использование Native POSIX Thread Library (NPTL);
 - + улучшен загрузчик модулей;
 - + добавлена новая служебная файловая система sysfs;
 - + интеграция User-mode Linux;
 - + и др.

Архитектура

Ядро Linux поддерживает многозадачность, виртуальную память, динамические библиотеки, отложенную загрузку, производительную систему управления памятью и многие сетевые протоколы.

На сегодняшний день Linux — монолитное ядро с поддержкой загружаемых модулей. Драйверы устройств и расширения ядра обычно запускаются на «кольце 0», с полным доступом к оборудованию. В отличие от обычных монолитных ядер, драйвера устройств легко собираются в виде модулей и загружаются или выгружаются во время работы системы.

То, что архитектура Linux не является микроядерной, вызвало обширнейшие прения между Линусом Торвальдсом и Эндрю Таненбаумом в конференции comp.os.minix(англ.) в 1992 г.

Совместимость

Не задуманный изначально как многоплатформенное ядро, Linux на данный момент портирован на очень широкий круг архитектур, запускается на широком спектре оборудования от iPAQ (карманный компьютер) до IBM S/390 (высокопроизводительный мейнфрейм). Linux используется как ядро операционной системы на суперкомпьютере Blue Gene (англ.) фирмы IBM.

Изначально Linux разрабатывался для 32-битных x86-совместимых ПК; на сегодняшний день Linux запускается на следующих процессорных архитектурах:

- ARM
 - Acorn: Archimedes, A5000, RiscPC
 - StrongARM, Intel XScale и т. п.
 - HP iPAQ
- Axis Communications CRIS
- DEC Alpha
- HP PA-RISC
- Hitachi: SuperH (SEGA Dreamcast), H8/300
- IBM S/390
- IBM zSeries-мэйнфреймы
- Intel 80386 и выше: IBM PC и совместимые с процессорами:
 - 80386, 80486, а также AMD, Cyrix, TI и IBM-варианты;
 - серия Pentium;
 - Core, Core2 Duo в 32 и 64-х битных версиях.
 - AMD Am5x86, K5, K6, Athlon (все 32-битные версии), Duron;
 - AMD64: 64-битная технология AMD (также известная как x86-64);
 - Cyrix 5x86, 6x86 (M1), 6x86MX и MediaGX (National/AMD Geode) серия;
 - VIA C3 (англ.) и последующие процессоры;
 - поддержка Intel 8086, 8088, 80186, 80188 и 80286 процессоров находится в разработке (см. проект ELKS(англ.));
 - Microsoft Xbox (Pentium III).
- Intel IA-64
- MIPS
 - Silicon Graphics, Inc.;
 - Cobalt Qube, Cobalt Raq;
 - Sony PlayStation 2, PlayStation 3;
 - DECstation;
 - некоторые другие.
- Motorola 68020 и выше:
 - более новые Amiga: A1200, A2500, A3000, A4000
 - Apple Macintosh II, LC, Quadra, Centris и ранняя серия Performa
 - рабочие станции Sun Microsystems серии 3 (экспериментальная, с использованием Sun-3 MMU)
- NEC v850e
- Renesas M32R
- PowerPC и IBM POWER:

- о все новые компьютеры Apple (все оснащённые PCI Power Macintoshes, ограниченная поддержка NuBus Power Macs)
- о клоны PCI Power Mac, разработанные Power Computing, UMAX и Motorola
- о IBM RS/6000, iSeries- и pSeries-системы
- о Pegasos I и II системы
- о некоторые встроенные системы PowerPC
- SPARC и UltraSparc: Sun 4-series, SPARCstation/SPARCserver, Ultra-, Blade- и Fire-серии рабочих станций и серверов

Лицензия

Linux распространяется на условиях лицензии GNU General Public License, то есть свободно. Эту лицензию выбрал Линус Торвальдс практически сразу после того, как стало понятно, что его хобби начало получать распространение по всему миру. Обладателем торговой марки Linux™ является Линус, а помогает следить за соблюдением его прав и условий GPL Фонд свободного программного обеспечения.

Символ

Официальным символом Linux является пингвин по имени Tux, отличающийся от «обычных» пингвинов жёлтым цветом клюва и лап.

BSD

Ссылка на оригинал: <http://ru.wikipedia.org/wiki/BSD>

С версии: 1.3

BSD (англ. Berkeley Software Distribution) — система распространения программного обеспечения в исходных кодах, созданная для обмена опытом между учебными заведениями. Особенностью пакетов ПО BSD была специальная лицензия BSD, которую кратко можно охарактеризовать так: весь исходный код — собственность BSD, все правки — собственность их авторов.

В данный момент термин BSD чаще всего употребляется как синоним BSD-UNIX общего названия вариантов UNIX, восходящих к дистрибутивам университета Беркли.

К семейству BSD относятся: NetBSD, FreeBSD, OpenBSD, DragonFly BSD, PC-BSD, DesktopBSD, Darwin (ядро Mac OS X).

Отличия от «классической» системы UNIX состоят в отсутствии переключения уровней выполнения, системе печати, файловой системе и командного процессора. Пользовательские команды практически идентичны.

История BSD

В начале 1978 года аспирант UCB Билл Джой (Bill Joy) начал создание дистрибутива программ Беркли (Berkeley Software Distribution — BSD). Первая лента (360м, ценой около \$50) содержала систему Pascal для UNIX и редактор ех; распространено около 30 копий. До конца 1978 года распространялся пакет 2BSD, продано около 75 копий. Приблизительно тогда же компания Interactive Systems выпускает первую коммерческую версию UNIX, а фирма Whitesmiths — первый клон системы UNIX — Idris.

Версия UNIX 7 была первой переносимой операционной системой. Однако её производительность была хуже, чем в версии шесть. После многих доработок в январе 1982 года вышла в свет модель 2.8.1BSD.

Впоследствии вышли версии:

- * 4BSD — октябрь 1980
- * 4.1BSD — июнь 1981
- * 4.1a, 4.1b и 4.1c (1982—1983)
- * 4.2 — сентябрь 1983 — действительно сильно переработанная система, включавшая поддержку сетей (протокол TCP/IP, новую файловую систему и возможность использования сигналов.

Последней университетской версией была 4.4BSD(июнь 1993). После суда права на BSD были переданы компании BSDI (Berkeley Software Design, Inc).

На базе BSD-UNIX существуют несколько операционных систем с открытыми исходными текстами:

- * NetBSD
- * FreeBSD
- * OpenBSD

Вводная информация о Linux

- * DragonFlyBSD
- * PC-BSD
- * DesktopBSD

GNU General Public License

Ссылка на оригинал: http://ru.wikipedia.org/wiki/GNU_General_Public_License

С версии: 1.3

GNU General Public License (Универсальная общедоступная лицензия GNU или Открытое лицензионное соглашение GNU) — возможно, наиболее популярная лицензия на свободное программное обеспечение, созданная в рамках проекта GNU в 1988 г. Её также сокращённо называют GNU GPL или даже просто GPL, если из контекста понятно, что речь идёт именно о данной лицензии (существует довольно много других лицензий, содержащих слова «general public license» в названии). Вторая версия этой лицензии была выпущена в 1991 году, третья версия, после многолетней работы и длительной дискуссии — в 2007 году. GNU Lesser General Public License (LGPL) — это модифицированная версия GPL, предназначенная для некоторых библиотек ПО.

Цель GNU GPL — предоставить пользователю права копировать, модифицировать и распространять программы (что по умолчанию запрещено законом об авторских правах), а также гарантировать, что и пользователи всех производных программ получают вышеперечисленные права. Принцип «наследования» прав называется «копилефт» (транслитерация английского «copyleft») и был придуман Ричардом Столлмэном. По контрасту с GPL, лицензии собственного ПО очень редко дают пользователю такие права и обычно, наоборот, стремятся их ограничить, например, запрещая восстановление исходного кода.

Свобода

GPL предоставляет получателям компьютерных программ следующие права, или «свободы»:

- свободу запуска программы, с любой целью;
- свободу изучения того, как программа работает, и её модификации (предварительным условием для этого является доступ к исходному коду);
- свободу распространения копий;
- свободу улучшения программы, и выпуска улучшений в публичный доступ (предварительным условием для этого является доступ к исходному коду).

История

GPL была написана Ричардом Столлмэном для использования с программами как часть проекта GNU. Она базируется на сходных лицензиях, использовавшихся для ранних версий GNU Emacs, GDB (отладчика GNU) и Коллекции компиляторов GNU (GCC), унифицирует и обобщает их.

Лицензии-прототипы содержали части, подобные частям GPL, но были специфичными для каждой программы. Целью Столлмэна являлось создание единой лицензии, которая могла бы использоваться для любого проекта, делая таким образом возможным совместное использование кода различными программами. Такой лицензией и стала первая версия GNU GPL, выпущенная в январе 1989 года.

В 1990 году стало очевидным, что требуется менее ограничивающая лицензия, которая могла бы использоваться для некоторых библиотек ПО; когда версия 2 GPL была выпущена в июне 1991 года, вместе с ней была введена в обращение GNU Library General Public License, также получившая номер 2, для обозначения того, что эти две лицензии являются взаимодополняющими. Номера версий разошлись в 1999 году, когда была выпущена LGPL версии 2.1, которая была переименована в Lesser General Public License для уточнения её местоположения в философии GNU.

GPL v3

В 2005 году Эбен Моуглен и Ричард Столлмэн написали черновик GPL версии 3. В разгоревшейся затем 7 апреля 2005 года дискуссии в Филадельфии, Столлмэн сделал несколько заявлений, касающихся патентов на ПО и DRM.

В 2006 году Free Software Foundation начал двенадцати месячную консультацию о возможных изменениях в GPL. Этот процесс координируется Фондом свободного программного обеспечения, Правовым центром свободы программного обеспечения и Европейским Фондом свободного программного обеспечения. Целью консультаций является создание новой версии лицензии с учётом рекомендаций и опыта всех заинтересованных сторон, но с сохранением приверженности принципам свободного ПО.

Первый черновик был опубликован 16 января 2006 года.

Тем не менее, 25 января 2006 года Линус Торвальдс публично заявил, что ядро Linux, скорее всего, будет по-прежнему распространяться по лицензии GPL версии 2. (В отличие от многих других GPL-программ, Linux распространяется на условиях только второй версии GPL, а не «версии 2 или более поздней»).

В своём сообщении в почтовую рассылку для Linux-разработчиков Линус Торвальдс, автор ядра Linux, говорит о том, что ОС Solaris может инициировать переход Linux-ядра на новую готовящуюся версию лицензии на свободное программное обеспечение — GNU GPLv3.

«Если Sun действительно собирается выпустить OpenSolaris под GPLv3, это может стать хорошей причиной» для перехода Linux на новую лицензию, заявил Торвальдс.[1]

«Не думаю, что GPLv3 — такая же хорошая лицензия, как и GPLv2, но с другой стороны, я прагматик, и, если мы можем избежать ситуации с существованием двух ядер, распространяющихся под разными лицензиями, и с вызванными этим разногласиями, я по крайней мере вижу причину для перехода на GPLv3», — пояснил свою позицию главный разработчик Linux-ядра.

До этого Торвальдс уже выражал своё недовольство новой версией лицензии GNU GPL, однако после появления последнего чернового варианта GPLv3 стал

лучше относиться к этому проекту. Несмотря на это, сам он до сих пор отдаёт предпочтение GPLv2.

Компании, распространяющие GPLv3-ПО, не могут предъявлять судебные претензии к пользователям GPLv3-продуктов.

Окончательная версия GPLv3 была опубликована 29 июня 2007. Черновой вариант перевода можно прочитать тут.

Схема GNU GPL

Текст GNU GPL состоит из нескольких пронумерованных разделов. Ниже приведена схема версии 2.0 лицензии. Эта схема не имеет никакой юридической силы и служит только для краткого ознакомления.

0. Определения
 - * (первый абзац) Определение термина «программа»
 - * (второй абзац) Область действия лицензии
1. Право на копирование и распространение
2. Изменение программы
 - * (первый абзац) Право на изменения при соблюдении следующих условий:
 - о а) добавление информации об изменении в модифицированных файлах;
 - о б) лицензирование модифицированных версий на условиях GNU GPL;
 - о с) условное требование интерактивного вывода информации об авторских правах и отсутствии гарантии.
 - * (абзацы 2—4) Уточнение термина «производная работа»
3. Требование предоставления исходного кода
 - * (первый абзац) Возможные варианты распространения исполнимого кода:
 - о а) распространение вместе с исходным кодом, или
 - о б) распространение с гарантией предоставления исходного кода, или
 - о с) (для некоммерческого использования) распространение вместе с такой гарантией, полученной от третьего лица.
 - * (второй абзац) Определение термина «исходный код»
 - * (третий абзац) Достаточность одинакового доступа для копирования исполнимого и исходного кодов
4. Прекращение действия лицензии при нарушении её условий
5. Акты, означающие принятие лицензии
6. Запрещение дополнительных ограничений при дальнейшем распространении
7. Внешние ограничения не снимают обязательства выполнять условия лицензии
8. Возможность географических ограничений
9. Будущие версии GNU GPL
10. Запросы на исключения из правил
11. Отказ от предоставления гарантий
12. Отказ от ответственности

Linux не для идиотов: сборник рассказов и рецептов

Ссылка на оригинал: <http://myfotomx.com/dalth/linuxbook.odt>

Автор: Артем Капитула (no-dashi, dalth & viking).

Скачано с <http://myfotomx.com/dalth>: 15.07.07

С версии: 1.1

Предисловие

Я много лет работал с Linux, и, общаясь со многими единомышленниками, сделал один странный вывод: нам катастрофически не хватает документации. Причем не инструкций вида «сделайте так и вот так» и не справочных руководств, а некоторого «мостика» между новичком, который видел только графическую оболочку подобную GNOME или KDE, и профессионалом, который может скомпилировать необходимый ему драйвер, даже если этот драйвер упорно сопротивляется.

Соответственно, я попытался сделать попытку написать книжку (хотя на книгу этот материал не тянет, скорее на методичку), которая позволила бы сравнительно просто перейти с пользовательского уровня знакомства с Linux на более высокий уровень, не проходя по типичным ошибкам, и за сравнительно короткое время.

Некоторое время я использовал фрагменты этой книги также как часть учебного курса в Челябинском Государственном университете (ЧелГУ) для того, чтобы мои студенты могли ориентироваться в системе несколько лучше, чем на уровне команд ls/ps/exit. Конечно, если честно – как учебное пособие эта маленькая книга непригодна, но, как мне кажется, она неплохо подходит как дополнительная литература.

Если у вас есть пожелания и дополнения – пишите мне почтой на dalt74@gmail.com, я постараюсь учесть ваши замечания в следующей редакции. Большое спасибо всем тем кто участвовал в рецензировании и помогал советами и замечаниями.

Искренне ваш, **Артем Капитула** (no-dashi, dalth & viking)

P.S. если вы будете распечатывать или выкладывать это пособие – пожалуйста, указывайте ссылку на автора и оригинальный источник, договорились?

P.P.S. пока что есть следующий to-do list: основы DNS, базовые настройки серверов и рабочих станций под типичные нужды. Ориентировочный срок следующей редакции – через три месяца.

Ядро и модули

Ядро Linux является единственным процессом, имеющим непосредственный доступ к аппаратуре – все остальные процессы обращаются к устройствам только через ядро. В ядре Linux можно выделить несколько важных подсистем: подсистему управления памятью; планировщик задач; подсистему VFS –

виртуальную файловую систему и драйверы.

Подсистема управления памятью управляет распределением оперативной памяти между задачами, а также обслуживает файл подкачки, планировщик задач управляет разделением процессорного времени между задачами (процессами и нитями), подсистема VFS предназначена для обслуживания файловых операций.

Драйверы предназначены для управления устройствами и поддержки различных протоколов. Существует две разновидности драйверов – статически подключенные в ядро драйверы и загружаемые модули; первые всегда загружены, вторые могут быть загружены при необходимости и выгружены, когда необходимость в них отпала. Каждый модуль и само ядро содержат сигнатуру версии – специальную метку, которая описывает версию ядра и некоторые опции, использованные при компиляции ядра. Кроме того, ядра версии 2.6 могут поддерживать цифровую подпись модулей. Это было сделано для повышения надежности системы – по умолчанию ядро не будет загружать и использовать драйверы, предназначенные для другой версии ядра, или собранные с другими опциями, поскольку это может привести к возникновению проблем. Версию ядра можно узнать с помощью команды `uname`:

```
[dalth@inferno dalth]$ uname -r
2.6.8.1
```

В принципе, утилиты для работы с модулями поддерживают возможность загрузки модулей, собранных для другого ядра – но пользоваться этой возможностью следует с крайней осторожностью, поскольку это может привести к непредсказуемым последствиям – от ошибок ядра (`kernel panic`) и вплоть до странных потерь данных и непонятных ошибок, взявшихся на пустом месте.

В большинстве дистрибутивов образ ядра располагается в каталоге `/boot`, а загружаемые модули ядра располагаются в `/lib/modules/<версия_ядра>`, там же располагается таблица зависимостей модулей, поскольку некоторые модули могут нуждаться для своей работы в других модулях (например, драйвер поддержки SCSI-дисков нуждается в драйвере поддержки SCSI – как следствие этого, если объекты какого-либо модуля используются другим драйвером, такой модуль невозможно выгрузить). Следующий листинг демонстрирует достаточно типичное содержание каталога модулей для ядер линейки 2.6:

```
[root@viking dev]# ls -l /lib/modules/2.6.8.1/
total 616
lrwxrwxrwx      1 root root          18 Apr 27 15:36 build ->
/usr/src/linux
drwxr-xr-x    10 root root      4096 Окт 1 13:55 kernel
-rw-r--r--     1 root root 108680 Окт 1 13:56 modules.alias
-rw-r--r--     1 root root     69 Окт 1 13:56 modules.ccwmap
-rw-r--r--     1 root root 153967 Окт 1 13:56 modules.dep
-rw-r--r--     1 root root     73 Окт 1 13:56 modules.ieee1394map
-rw-r--r--     1 root root    357 Окт 1 13:56 modules.inputmap
-rw-r--r--     1 root root 16658 Окт 1 13:56 modules.isapnpmap
http://www.eclipse.org/downloads/download.php?
file=/eclipse/downloads/drops/R-3.3.1.1-200710231652/eclipse-
SDK-3.3.1.1-linux-gtk-x86_64.tar.gz&url=http://ftp-stud.fht-
esslingen.de/pub/Mirrors/eclipse/eclipse/downloads/drops/R-3.3.1.1-
200710231652/eclipse-SDK-3.3.1.1-linux-gtk-
x86_64.tar.gz&mirror_id=17-rw-r--r--     1 root root 85093 Окт 1
13:56 modules.pcimap
-rw-r--r--     1 root root 68078 Окт 1 13:56 modules.symbols
-rw-r--r--     1 root root 150781 Окт 1 13:56 modules.usbmap
lrwxrwxrwx      1 root root          18 Окт 1 13:22 source ->
/usr/src/linux
[root@viking dev]# find /lib/modules/2.6.8.1/kernel -type f | head
-20
/lib/modules/2.6.8.1/kernel/arch/i386/kernel/cpuid.ko
/lib/modules/2.6.8.1/kernel/arch/i386/kernel/microcode.ko
/lib/modules/2.6.8.1/kernel/arch/i386/kernel/msr.ko
/lib/modules/2.6.8.1/kernel/crypto/blowfish.ko
/lib/modules/2.6.8.1/kernel/crypto/deflate.ko
/lib/modules/2.6.8.1/kernel/crypto/md5.ko
/lib/modules/2.6.8.1/kernel/crypto/twofish.ko
/lib/modules/2.6.8.1/kernel/drivers/acpi/fan.ko
/lib/modules/2.6.8.1/kernel/drivers/acpi/processor.ko
/lib/modules/2.6.8.1/kernel/drivers/acpi/thermal.ko
/lib/modules/2.6.8.1/kernel/drivers/base/firmware_class.ko
/lib/modules/2.6.8.1/kernel/drivers/block/cryptoloop.ko
/lib/modules/2.6.8.1/kernel/drivers/block/loop.ko
/lib/modules/2.6.8.1/kernel/drivers/block/nbd.ko
/lib/modules/2.6.8.1/kernel/drivers/block/paride/epat.ko
/lib/modules/2.6.8.1/kernel/drivers/block/paride/paride.ko
/lib/modules/2.6.8.1/kernel/drivers/block/paride/pd.ko
/lib/modules/2.6.8.1/kernel/drivers/block/paride/pg.ko
/lib/modules/2.6.8.1/kernel/drivers/bluetooth/bcm203x.ko
/lib/modules/2.6.8.1/kernel/drivers/bluetooth/bfusb.ko
[root@viking dev]#
```

Бинарные файлы модулей содержатся в подкаталоге *kernel*, и имеют расширение “.o” для ядер линейки 2.4 и расширение “.ko” для ядер линейки 2.6. В файлах *modules.***map* перечисляются символы (функции и переменные), экспортируемые модулями.

Часто в одном каталоге с модулями содержатся ссылки на каталоги, в которых хранились исходные тексты ядра и в котором производилась сборка ядра (это ссылки *source* и *build*, соответственно). Эти ссылки, как правило, используются для того, чтобы скомпилировать модули или программы, которые зависят от версии ядра (например, эти ссылки используются при инсталляции модуля поддержки видеокарт *nvidia*).

Учет взаимосвязей между загруженными модулями производится с помощью

счетчика ссылок – модуль увеличивает свой счетчик ссылок как только какой-либо его объект начинает использоваться другими драйверами. Когда объекты модуля освобождаются, счетчик ссылок уменьшается. Модуль может быть выгружен, если число ссылок на него станет равно 0. В ядрах версии 2.6 существует возможность произвести принудительную выгрузку модуля даже если он используется, но этим пользоваться без крайней необходимости не рекомендуется, поскольку очень возможно возникновение ошибок.

Ядро содержит множество переменных и функций, которые используются различными драйверами, и соответственно, если какой-либо драйвер должен обратиться к такому объекту, он должен знать его адрес. Некоторые драйверы также содержат переменные и функции, которые должны быть доступны другим драйверам, и адреса таких объектов тоже размещаются в специальной таблице. При загрузке модуля ядро и программа загрузки модулей устанавливает адреса всех объектов, в которых нуждается загружаемый модуль, и только после этого модуль может начать инициализацию.

Загрузка модулей и их выгрузка осуществляются утилитами *modprobe*, *insmod* и *rmmod*. Программы *modinfo* и *depmod* предназначены для получения служебной информации о загружаемых модулях. В процессе своей работы эти утилиты опираются на конфигурационные файлы */etc/modprobe.conf* (для ядер 2.6.X) или *modules.conf* (для ядер 2.4.X).

Загрузка операционной системы

Для компьютеров архитектуры x86 последовательность загрузки хорошо описана в специализированной литературе, но мы все-таки кратко ее повторим. После включения компьютера первым загружается BIOS. Он тестирует аппаратуру и инициализирует устройства. После этого BIOS прочитывает начальный сектор загрузочного жесткого диска (MBR), убеждается что он содержит код первичного загрузчика, и передает управление прочитанному коду. Кроме кода первичного загрузчика, начальный сектор также может содержать таблицу разделов жесткого диска.

В задачи первичного загрузчика входит чтение основного кода загрузчика операционной системы и передача управления ему, после чего основная часть загрузчика может считать конфигурационный файл, загрузить ядро операционной системы, установить параметры для ядра и передать ядру управление. Ядро инициализирует драйверы, проверяет параметры и, опираясь на параметры, пытается смонтировать корневую файловую систему, после чего (если не было проинструктировано об ином) запускает программу */sbin/init*. Дальнейшая работа *init* подробно описана во множестве книг и статей.

В настоящий момент в мире Linux наиболее распространен загрузчик GRUB. Этот загрузчик состоит из нескольких частей – первичного загрузчика, собственно основного кода который организует интерфейс пользователя, и набора мини-драйверов различных файловых систем, позволяющих прочесть необходимые файлы с файловой системы в момент когда операционная система еще недоступна. Каждая из этих компонент работает на одном из двух этапов загрузки. Рассмотрим эти этапы:

Этап 0 – здесь срабатывает первичный загрузчик GRUB. Он компактен и умещается в один блок жесткого диска, что позволяет при желании разместить его в MBR. В задачи кода *stage_0* входит прочтение кода необходимого на следующем этапе (собственно кода загрузчика и мини-драйвера файловой системы где расположены основные файлы загрузчика), и передача управления прочитанному коду.

Этап 1 – это на этом этапе первичным загрузчиком в память уже загружен основной код загрузчика, а также мини-драйвер файловой системы, на которой расположены

конфигурационные файлы загрузчика, ядро и необходимые драйверы. Основной код, используя функции мини-драйвера, прочитывает конфигурационный файл и организует диалог с пользователем. В зависимости от выбора пользователя, используя мини-драйвер файловой системы, с диска прочитываются файлы ядра и необходимых драйверов, после чего управление передается ядру. Как вариант, пользователь может отказаться от загрузки Linux и инструктировать GRUB прочесть загрузочный сектор некоторого раздела жесткого диска и передать управление ему.

Первичный загрузчик из состава GRUB может быть расположен как в MBR, так и в загрузочном секторе какого-либо раздела жесткого диска – или даже храниться в файле и быть вызван из другого загрузчика (например NTLOADER).

Нередко случаются ситуации, когда корневая файловая система располагается на устройстве, чей драйвер скомпилирован в виде модуля, или драйвер корневой файловой системы скомпилирован в виде модуля. Получается замкнутый круг – чтобы смонтировать корневую файловую систему, необходимо прочесть драйвер, а чтобы прочесть драйвер – нужно смонтировать корневую файловую систему. Чтобы разорвать этот порочный круг, в Linux была введена поддержка *initrd* – INITial RamDisk.

Initial ramdisk – это файл, который прочитывается загрузчиком ОС и загружается в память вместе с ядром. Ядро интерпретирует фрагмент памяти, куда загружен этот файл, как блочное устройство с помощью специального драйвера, статически вкомпилированного в ядро. После инициализации статически скомпилированных драйверов ядро монтирует файловую систему, хранящуюся в *initrd* и загружает с нее драйверы и запускает программы, необходимые для монтирования корневой файловой системы.

Обычно файл с образом ядра хранится в каталоге */boot* и называется *vmlinuz-
<версия>*, там же располагается файл *initrd-
<версия>.img*, содержащий образ файловой системы *initrd*. Для каждой версии ядра необходим свой образ *initrd*, в который включены модули для этой версии ядра. В большинстве случаев образ *initrd* поставляется в бинарном пакете вместе с ядром, или автоматически создается в процессе построения ядра из исходных текстов в момент выполнения команды *make install*, если же возникает ситуация, когда необходимо повторно собрать образ *initrd* (например, в сервере сменили SCSI-контроллер), можно воспользоваться специальной командой *mkinitrd*, позволяющей произвести повторную генерацию *initrd*:

```
[root@viking dalth]# mkinitrd /tmp/initrd-2.4.8.1.img 2.6.8.1
[root@viking dalth]# cp /tmp/initrd-2.4.8.1.img /boot
[root@viking dalth]# reboot
```

Для Linux существует два основных загрузчика – LILO и GRUB. Второй является более поздней разработкой и немного удобней в использовании, а LILO используется по историческим или личным причинам (например, он нравится системному администратору), либо в некоторых случаях, когда требуются специфичные для LILO функции. Для более подробной справки лучше обратиться к справочному руководству (*man grub*, *man lilo*).

Из интересных особенностей GRUB и LILO следует отметить то, что и оба этих загрузчика, и ядро оперируют термином корневой файловой системы – но если с точки зрения ядра эта та файловая система, которая содержит программу */sbin/init*, то с точки зрения обоих загрузчиков корневой файловой системой является та, которая содержит образ ядра и файл *initrd*.

Организация памяти

Подсистема виртуальной памяти управляет распределением оперативной памяти

между задачами (процессами). Каждая задача считает, что ей выделен непрерывный участок памяти максимального размера, поддерживаемого на соответствующей архитектуре (для архитектуры x86 это 4GB). Из них последний гигабайт резервируется для себя ядром, часть отдается под код программы и разделяемые библиотеки (оба этих фрагмента ядром защищаются), а оставшееся пространство отдается собственно программе под ее данные – но это только то, как видит это все программа.

На самом же деле программа занимает только тот объем памяти, с которым она реально работает. Большинство памяти существует только “на бумаге”, т.е. будет предоставлена программе в тот момент, когда она обратится в эту область. Ядро распределяет память страницами фиксированного размера. Процедура, когда страница оперативной памяти объявляется частью адресного пространства процесса, называется *отображением* этой страницы в адресное пространство процесса.

Соответственно, ядро отображает реально используемые страницы в виртуальное адресное пространство процесса. Когда процесс обращается к некоторой странице своего адресного пространства, ядро проверяет, имеет ли он право на доступ к этой странице, и если проверка пройдена и доступ получен, то ядро переадресовывает обращение на реальный адрес этой страницы. Если это первое обращение к странице, ядро попытается найти свободную страницу и в случае успеха отобразит ее в адресное пространство соответствующего процесса. Размер страницы фиксирован архитектурой процессора, и для x86 ее размер составляет 4096 байт.

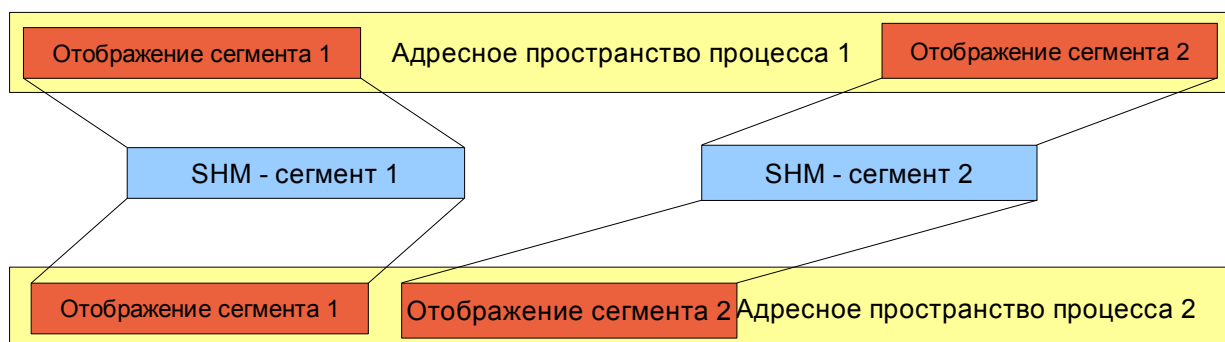
Если случается ситуация, когда свободных страниц больше нет, но существует файл подкачки, ядро может убрать одну из наиболее долго не использовавшихся страниц в файл подкачки, и освободившуюся физическую страницу отдать запросившему память процессу. Если же нет ни незанятого пространства в файле подкачки, ни свободных страниц RAM, то развитие событий может быть следующим: либо запросивший память процесс прерван и “убит” системой, либо какой-то другой из процессов (это определяется специфическими алгоритмами) будет “убит” ядром, и освободившаяся память будет передана запросившему память процессу.

На самом деле большинством действий занимается одна из подсистем процессора, называемая MMU – Memory Management Unit, и в действительности ядро просто полагается на его работу и вмешивается в нее только для проведения операций пейджинга (подгрузки/выгрузки страниц в SWAP-файл), или когда возникает ошибка доступа к странице.

Ограничение адресного пространства в 4GB не означает, что система не сможет адресовать более этого объема памяти. На платформе x86 ядро Linux может использовать до 64GB, а ограничение в 4GB накладывается лишь на размер адресного пространства процесса.

System V shared memory

Linux поддерживает стандартную для всех UNIX-подобных операционных систем организацию разделяемой памяти. Пользовательские приложения могут создавать *сегменты* разделяемой памяти, которые могут быть присоединены к некоторому фрагменту адресного пространства процесса. Любой процесс, имеющий достаточные права доступа, может присоединиться к сегменту разделяемой памяти, и отобразить его в свое адресное пространство, начиная с некоторого адреса.



Если в приведенной схеме любой из процессов изменит содержимое памяти в области, занимаемой отображением одного из сегментов, то же самое изменение произойдет в адресном пространстве другого процесса, поскольку соответствующий сегмент существует в одном экземпляре и отображен в адресное пространство обоих процессов.

Кроме System V IPC ядро Linux также поддерживает другие объекты IPC, в частности семафоры и очереди сообщений. Каждый объект System V IPC идентифицируется уникальным ключом. Просмотреть список всех объектов IPC можно командой *ipcs*. Команда *ipcrm* позволяет удалять объекты IPC, которые по каким-либо причинам остались не освобожденными после завершения создавшего их процесса – например, такая ситуация может возникнуть после аварийного завершения работы СУБД Oracle, Informix или DB2.

Соответственно, перед перезапуском процесса системный администратор с помощью команды *ipcrm* должен освободить неиспользуемые объекты IPC, поскольку стартующее приложение не сможет их повторно создать и не будет корректно работать.

Для каждого объекта IPC система устанавливает права доступа, как если бы это был файл (т.е. для каждого объекта IPC можно устанавливать набор прав *ugo/rwx*, но в отличие от обычных файлов сменить права доступа для IPC-объектов можно только вызывая специализированные функции, предназначенные для работы с такими объектами.

```
[dalth@viking dalth]$ ipcs

----- Shared Memory Segments -----
key          shmid          owner          perms          bytes          nattch
status
0x00000000  0                oracle        640            4194304        10
0x00000000  32769           oracle        640            20971520       10
0x00000000  65538           oracle        640            29360128       10
0x0d3c24a0  98307           oracle        640            29360128       50
0x00000000  13697028        root          777            49152          1
0x00000000  13729797        root          777            16384          1
0x000004d2  13795334        dalth         666            1008           2
0x00000000  14286866        root          644            790528         2
dest
0x00000000  21823507        dalth         600            393216         2
dest
0x00000000  21921814        root          644            122880         2
dest
0x00000000  14516249        root          644            151552         1
dest

----- Semaphore Arrays -----
key          semid          owner          perms          nsems
0x0b4f657c  262147         oracle        640            154
0x000004d2  458756         dalth         666            1

----- Message Queues -----
key          msqid          owner          perms          used-bytes      messages
[dalth@viking dalth]$
```

Поддержка System V IPC позволяет сравнительно легко переносить на Linux приложения, написанные для других UNIX-систем.

Файловая система

VFS и драйверы файловых систем являются одной из важнейших составляющих ядра. Для того, чтобы получить доступ к файлам, хранящимся на каком-либо устройстве хранения данных, необходимо, чтобы в ядро был загружен драйвер соответствующей файловой системы, и файловая система была смонтирована. Драйвера всех файловых систем поддерживают набор стандартных функций: открыть файл по имени, записать данные в файл, прочитать данные из файла, закрыть файл, удалить файл и т.д.

Уточним, что драйвера файловых систем не занимаются кэшированием, этим занимается VFS.

При первоначальной загрузке драйвер файловой системы регистрирует в VFS имя файловой системы и те функции, которые предназначены для выполнения стандартных файловых операций. Впоследствии при обращении к файлу на какой-либо файловой системе VFS будет переадресовывать обращение на соответствующую функцию, если таковая была зарегистрирована драйвером. Посмотреть список обслуживаемых ядром файловых систем можно в файле `/proc/filesystems`:

```
[dalth@viking proc]$ cat /proc/filesystems
nodev    sysfs
nodev    rootfs
nodev    bdev
nodev    proc
nodev    sockfs
nodev    usbfs
nodev    usbdevfs
nodev    futexfs
nodev    tmpfs
nodev    pipefs
nodev    eventpollfs
nodev    devpts
nodev    ext2
nodev    ramfs
nodev    hugetlbfs
nodev    iso9660
nodev    devfs
nodev    mqueue
nodev    ext3
nodev    rpc_pipefs
nodev    nfsd
nodev    smbfs
```

Операция монтирования предназначена для того, чтобы сделать доступной файловую систему, расположенную на каком-либо блочном устройстве. Суть операции монтирования заключается в том, что ядро ассоциирует некоторый каталог (называемый точкой монтирования) с блочным устройством и драйвером файловой системы. Для этого оно передает ссылку на блочное устройство драйверу файловой системы, и в случае, если драйвер успешно проидентифицировал эту файловую систему, ядро заносит в специальную таблицу монтирования информацию о том, что все файлы и каталоги, чей полный путь начинается с указанной точки монтирования, обслуживаются соответствующим драйвером файловой системы и расположены на указанном блочном устройстве.

Некоторые файловые системы не нуждаются в блочном устройстве, поскольку хранят свои данные исключительно в памяти, например файловая система *procfs*, через файлы которой можно получить доступ к различным системным параметрам и таблицам.

Очень часто при монтировании файловой системы системный администратор имеет возможность задать *опции монтирования*. Опции монтирования – это специальные параметры, которые влияют на работу драйвера файловой системы, когда он работает с файловой системой на соответствующем блочном устройстве – например, с помощью опций монтирования можно управлять режимом кэширования данных, преобразованиями имен файлов и данных, включать и отключать поддержку ACL и т.д.

Посмотреть таблицу примонтированных файловых систем можно через файл */proc/mounts*:

```
[dalth@vikings proc]$ cat /proc/mounts
rootfs / rootfs rw 0 0
/dev/root / ext3 rw 0 0
none /dev devfs rw 0 0
/proc /proc proc rw,nodiratime 0 0
/sys /sys sysfs rw 0 0
none /dev/pts devpts rw 0 0
usbdevfs /proc/bus/usb usbdevfs rw 0 0
/dev/chimera/var /var ext3 rw 0 0
/dev/chimera/temp /tmp ext3 rw 0 0
/dev/chimera/usr /usr ext3 rw 0 0
/dev/chimera/home /home ext3 rw 0 0
/dev/chimera/opt /opt ext3 rw 0 0
none /dev/shm tmpfs rw 0 0
```

Виртуальная файловая система Linux различает несколько типов файлов: каталоги, обычные файлы, именованные каналы, символьные ссылки, сокеты и специальные файлы. Каждая из этих разновидностей обрабатывается своим собственным образом:

- Обычные файлы могут быть прочитаны, записаны, удалены или отображены в память.
- Каталог можно представить как список имен файлов, и для этого списка определены операции добавления элемента в список, удаление элемента из списка, переименование элемента списка.
- Именованные каналы являются просто буферами – в него можно записать некоторый объем данных, и прочесть их в том же порядке, в каком они были записаны.
- Сокеты являются вариантом именованных каналов, но если у именованного канала буфер только один, то есть нельзя определить какой процесс записал данные в канал, то у сокетов может быть несколько клиентов, один из которых осуществляет управление сокетом и может обмениваться данными с любым из остальных клиентов, а те в свою очередь могут обмениваться данными с диспетчером канала – то есть сокет поддерживает множество независимых буферов, по одному на каждую пару сервер+клиент
- Специальные файлы предназначены для того, чтобы осуществлять прямой доступ к устройствам. Подробнее о специальных файлах будет говориться в главе **Секреты /dev**
- Символьные ссылки являются “ярлыками”, которые могут содержать имя другого файла – и тогда операции чтения, записи и открытия/закрытия файла автоматически переадресовываются к файлу, на который указывает символьная ссылка, но в отличие от “ярлыков” Windows (shortcuts), символьные ссылки (symbolic links) не требуют специальных функций при работе – все программы (кроме тех, которые специально предназначены для работы с ними) видят их как обычные файлы, и также их открывают, читают и записывают данные и т.д.

В некоторых файловых системах, которые изначально проектировались для UNIX-подобных систем, есть возможность создавать кроме символьных ссылок еще и жесткие ссылки. Фактически, жесткая ссылка – это второе имя для файла. Жесткие ссылки возможно создавать только в пределах одной файловой системы.

Из-за того, что в VFS присутствует понятие кэширования, перед отключением системы необходимо делать обязательный сброс изменений на диск. Сброс кэша на диск осуществляется в момент размонтирования файловой системы. Кроме того, с помощью команды `sync` можно в любой момент принудительно сбросить на диск все закэшированные изменения в файловой системе (например, системные администраторы часто делают `sync` перед загрузкой нового драйвера). Размонтировать файловую систему можно только тогда, когда ни один процесс не удерживает в открытом состоянии файлов с этой файловой системы, а также не находится ни один процесс не имеет рабочим каталогом каталога с размонтируемой файловой системы. При невыполнении этого условия размонтировать файловую систему не удастся:

```
[root@viking dalth]# umount /home/ftp/pub/linux/fedora/cd1
umount: /home/ftp/pub/linux/fedora/cd1: device is busy
umount: /home/ftp/pub/linux/fedora/cd1: device is busy
[root@viking dalth]#
```

Некоторые файловые системы поддерживают специальные опции, позволяющие принудительно синхронизировать файловую систему при каждой операции чтения или записи. Обычно опции, влияющие на синхронизацию файловой системы, содержат в своем названии слово **sync**, например приведенная ниже команда инструктирует операционную систему примонтировать некоторый раздел в режиме принудительной синхронизации:

```
[root@viking dalth]# mount -t ext3 -o sync,dirsync /dev/hda9 /home
```

Следует учесть, что принудительная синхронизация – это удар по производительности операций записи для файловой системы, смонтированной в таком режиме, поэтому использовать такой его следует осторожно.

Права доступа

Кроме стандартных наборов прав доступа к файлам некоторые файловые системы Linux поддерживают т.н. *POSIX ACL* – списки контроля доступа POSIX. Эта возможность позволяет гибко управлять доступом к файлу, не ограничиваясь “классическим” набором `ugo/rwx`. Для того, чтобы использовать на файловой системе POSIX ACL, необходимо смонтировать файловую систему с опцией `acl`:

```
[root@inferno root]# mount -t ext3 -o acl /dev/inferno/opt /opt
```

Возможно также настроить соответствующий параметр для файловой системы, чтобы она поддерживала POSIX ACL по умолчанию:

```
[root@inferno root]# tune2fs -o acl /dev/inferno/opt
```

После установки соответствующей опции можно приступать к работе с POSIX ACL. Для работы с ними существует две базовых утилиты: *getfacl* для получения списка дополнительных атрибутов доступа, и *setfacl* для установки расширенных атрибутов контроля доступа. Если в выводе команды `ls` вы видите символ “+” рядом со списком стандартных прав доступа, это означает, что для файла также установлены расширенные атрибуты контроля доступа:

```
[root@inferno root]# ls -l /home/dalth/.bash_??????
-rw-r-----+ 1 dalth dalth 20034 Окт 11 22:48
/home/dalth/.bash_history
-rw-r--r-- 1 dalth dalth 191 Авг 23 21:51
/home/dalth/.bash_profile
[root@inferno root]#
```

Для просмотра значений расширенных атрибутов можно воспользоваться

утилитой `getfacl`. Жирным шрифтом выделен дополнительный атрибут контроля доступа, позволяющий пользователю `kiki` получить доступ на чтение к файлу `.bash_history`:

```
[root@inferno dalth]# getfacl .bash_history
# file: .bash_history
# owner: dalth
# group: dalth
user::rw-
user:kiki:r--
group::---
mask::r--
other::---
```

Добавим пользователю `oracle` права на чтение и запись файла `.bash_history` с помощью команды `setfacl`, и затем отберем дополнительные права на доступ к указанному файлу у пользователя `kiki`:

```
[root@inferno dalth]# setfacl -m u:oracle:rw .bash_history
[root@inferno dalth]# setfacl -x u:kiki .bash_history
[root@inferno dalth]# getfacl .bash_history
# file: .bash_history
# owner: dalth
# group: dalth
user::rw-
user:oracle:rw-
group::---
mask::rw-
other::---
```

Последним шагом сбросим все расширенные атрибуты с файла `.bash_history`:

```
[root@inferno dalth]# setfacl -b .bash_history
[root@inferno dalth]# ls -l .bash_history
-rw----- 1 dalth dalth 20034 Окт 11 22:48 .bash_history
```

Расширенные атрибуты позволяют гибко контролировать доступ к файловым объектам, обходя стратегию `ugo/rwx` пришедшую из “классического UNIX”. Права доступ на файловые объекты могут быть выданы не только пользователю, но и группе.

К сожалению, далеко не все утилиты и файловые системы поддерживают ACL, поэтому при резервном копировании или восстановлении файлов необходимо проверять корректность установки расширенных атрибутов и правильность их переноса.

Журналируемые файловые системы

Для обеспечения сохранности данных и обеспечения целостности файловых систем при неожиданных сбоях были разработаны *журналируемые файловые системы*. Как правило, у этих файловых систем существует специальная область данных, называемая *журналом*. Все изменения, которые необходимо произвести с файловой системой, сначала записываются в журнал, и уже из журнала попадают в основную часть файловой системы.

В большинстве случаев журналируются только *метаданные* файловых систем (служебная информация, обеспечивающая целостность структуры файловой системы — например, изменения в каталогах или служебных таблицах размещения файлов). Некоторые файловые системы позволяют журналировать

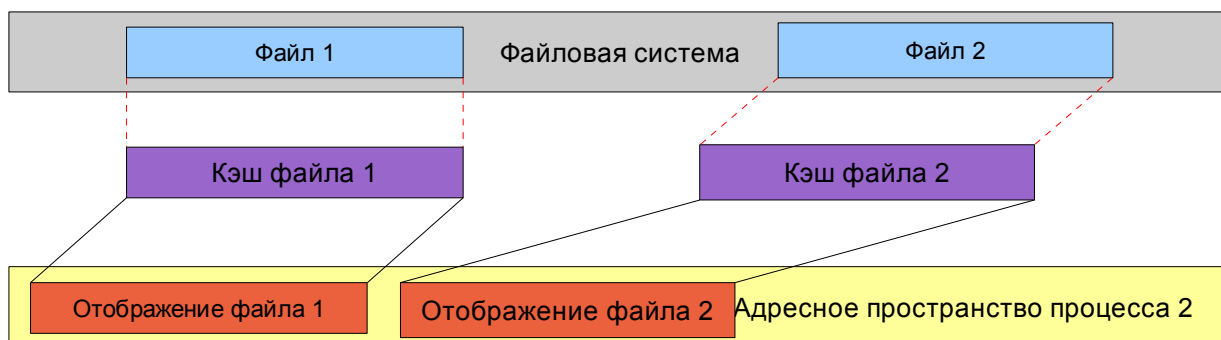
не только метаданные, но и данные файлов – такой шаг позволяет повысить надежность, но уменьшает скорость записи данных на файловую систему, поскольку каждый блок данных записывается на диск дважды – сначала в журнал, и затем из журнала в основную область файловой системы..

В большинстве случаев, журналируемые файловые системы способны решить проблемы с надежностью при неожиданных сбоях без тех потерь производительности, к которым может привести использование опции **sync** при монтировании.

В частности, к журналируемым файловым системам, например, относятся EXT3, ReiserFS, XFS, JFS и некоторые другие.

Отображенные в память файлы

Объединение кэширования файлов и разделяемой памяти позволяет реализовать такое действие, как отображение файла в память. Для простоты можно представить, что файл загружается в кэш, и страницы кэша отображаются в адресное пространство процесса, и в результате любое изменение в том фрагменте адресного пространства, которое занято отображением файла, автоматически попадает в закэшированные данные файла. Когда файл закрывается, закэшированные изменения сбрасываются на диск, изменяя сам файл. Кроме того, в свободное время ядро также постепенно сбрасывает изменившиеся кэшированные данные на диск.



На самом деле, механизм отображения файлов в память куда “хитрее” - при обращении по записи к странице, которая является отображением некоторого файла, ядро перехватывает обращение, производит запись в файл (в подавляющем большинстве случаев эта операция попадает в кэш). При обращении по чтению к такой странице ядро опять же перехватывает обращение и производит чтение из файла – в большинстве случаев это чтение производится из кэша. Для наших же целей проще будет считать, что страницы кэша отображены в память процесса.

Такая методика часто используется для загрузки разделяемых библиотек, когда выполняемый код библиотек и исполняемого кода программы хранится в кэше и через отображение файла в память становится виден в адресном пространстве процесса:

```
[dalth@viking dalth]$ cat /proc/1/maps
08048000-08050000 r-xp 00000000 03:01 75813 /sbin/init
08050000-08051000 rw-p 00008000 03:01 75813 /sbin/init
08051000-08072000 rw-p 08051000 00:00 0
40015000-40016000 rw-p 40015000 00:00 0
4c8ee000-4c903000 r-xp 00000000 03:01 92869 /lib/ld-2.3.3.so
4c903000-4c904000 r--p 00014000 03:01 92869 /lib/ld-2.3.3.so
4c904000-4c905000 rw-p 00015000 03:01 92869 /lib/ld-2.3.3.so
4c907000-4c908000 r-xp 00000000 03:01 92857 /lib/tls/libc-2.3.3.so
4c908000-4c909000 r--p 00115000 03:01 92857 /lib/tls/libc-2.3.3.so
4c909000-4c90a000 rw-p 00117000 03:01 92857 /lib/tls/libc-2.3.3.so
4c90a000-4c90b000 rw-p 4c90a000 00:00 0
4d201000-4d202000 r-xp 00000000 03:01 92965 /lib/libselinux.so.1
4d202000-4d203000 rw-p 0000d000 03:01 92965 /lib/libselinux.so.1
bffffd000-c0000000 rw-p bffffd000 00:00 0
fffffe000-fffff000 ---p 00000000 00:00 0
```

На самом деле, драйверы любого устройства и любой файловой системы могут по-своему реализовывать операцию mmap, но большинство драйверов файловых систем полагаются в этом на VFS.

Специальные файловые системы

Некоторые типы файловых систем являются специальными и предназначены для выполнения или реализации специфических задач операционной системы. К таким файловым системам относятся файловые системы *procfs* и *sysfs*, предоставляющие доступ к различным параметрам системы и системным объектам, “живущим” в ядре.

Файловая система *sysfs* в основном предоставляет доступ к объектам ядра и отображает взаимосвязи между ними. Файловая система *procfs* предоставляет доступ к различным параметрам ядра и драйверов и к пользовательским процессам, позволяя тем самым реализовать такие команды как *ps* или *sysctl*. Большинство файлов в *sysfs* двоичные, в *procfs* – текстовые.

Драйверы файловых систем *ramfs*, *tmpfs* и *shmfs* очень похожи, и после монтирования файловой системы такого типа на ней можно создавать файлы, хранящиеся в памяти и отличаются в основном небольшими особенностями работы (например, страницы, используемые *ramfs* под данные файлов, не вытесняются в *swap*-файл в отличие от *shmfs* и *tmpfs*). В ядре 2.6 *shmfs* была заменена на *tmpfs*.

Сетевые файловые системы

Сетевые файловые системы предназначены для получения доступа к файловым системам других компьютеров с использованием сетевых протоколов.

Наиболее часто используются сетевые файловые системы *NCPFS* (для доступа к серверам Novell NetWare), *SMBFS* (для доступа к серверам Windows) и *NFS* (для доступа к файловым системам других UNIX-систем).

Как правило, процедура монтирования сетевых файловых систем схожа с процедурой монтирования обычных файловых систем на блочных устройствах с

тем отличием, что вместо блочного устройства указывается адрес сервера, чья файловая система монтируется, и имя монтируемого ресурса. Для примера рассмотрим процедуры монтирования ресурсов, доступных по SMB и по NFS:

```
# mount -t smbfs -o username=usr,workgroup=tst //server/share_name  
/mnt/smb_target  
Password: *****  
# mount -t nfs -o timeout=4 server:/export/home /mnt/nfs_target
```

В данном примере опция *-t* команды *mount* указывает тип файловой системы, опция *-o* позволяет задать дополнительные параметры для монтирования – для SMB мы задаем, например, имя пользователя, с правами которого производится подключение к серверу и имя рабочей группы или домена, для NFS мы указываем таймаут, по истечении которого операция ввода/вывода считается неудавшейся. Вместо блочного устройства мы указываем адрес сервера, ресурс которого хотим использовать, и имя ресурса на сервере. Последним параметром идет точка монтирования.

Создание файловых систем

Для создания файловых систем в Linux используется команда *mkfs*:

```
[root@inefrno root]# mkfs -t ext3 /dev/hda6
```

На самом деле, *mkfs* является просто “оберткой” к реальным программам создания файловых систем, которые обычно именуются как *mkfs.<имя_ФС>*, например *mksf.ext2* или *mkfs.reiserfs*.

В большинстве случаев программы группы *mkfs* просто инициализируют специальную область раздела, называемую *суперблоком файловой системы*. Суперблок содержит ссылки на все значимые элементы файловой системы (например, ссылку на оглавление корневого каталога, ссылку список свободных блоков, ссылку на список файлов и т.д.)

Наличие суперблока (который практически всегда содержит в своем теле некоторое характерное значение) позволяет производить монтирование файловых систем без указания их типа. К сожалению, некоторые файловые системы вследствие своей архитектуры не содержат суперблока (в частности, FAT и большинство ее разновидностей) и для блочных устройств, содержащих такие файловые системы, автоматическое определение типа файловой системы затруднено.

Интерфейс sysctl

Ядро содержит очень много параметров, от которых зависит его производительность и которые могут изменять алгоритмы его работы. Для того, чтобы иметь возможность узнавать и изменять эти параметры, в UNIX был разработан интерфейс *sysctl*.

Виртуальная файловая система *procfs* содержит каталог *sys* с деревом подкаталогов и файлов. Содержимое каждого из этих файлов можно прочесть, например, командой *cat*, или записать в такой файл новое значение командой *echo*:

```
[root@inferno root]# cat /proc/sys/kernel/shmmax
33554432
[root@inferno root]# echo 67108864 >/proc/sys/kernel/shmmax
[root@inferno root]# cat /proc/sys/kernel/shmmax
67108864
[root@inferno root]#
```

Команда `sysctl` предназначена для того, чтобы избежать необходимости использовать прямой доступ к этим файлам, и предоставить возможность автоматизации установки таких параметров при загрузке системы. На самом же деле, команда `sysctl` просто читает или записывает значения в файлы из каталога `/proc/sys`, т.е. если системный администратор устанавливает с помощью команды `sysctl` значение некоторого параметра, фактически он просто записывает это значение в соответствующий файл. Существует однозначное соответствие между именем параметра и именем файла, через который его можно изменить: если посмотреть вывод `sysctl -a`, можно увидеть, что параметры в большинстве своем именуются несколькими мнемоническими аббревиатурами, разделенными точками:

```
[root@inferno root]# sysctl -a | grep mem
net.ipv4.tcp_rmem = 4096      87380    174760
net.ipv4.tcp_wmem = 4096      16384    131072
net.ipv4.tcp_mem = 24576      32768    49152
net.ipv4.igmp_max_memberships = 20
net.core.optmem_max = 10240
net.core.rmem_default = 108544
net.core.wmem_default = 108544
```

Если в имени параметра заменить точки на символ разделителя пути (символ `/`), и к началу получившейся строки добавить `/proc/sys/` - то мы получим имя файла, через который можно изменить или прочесть значение соответствующего параметра.

Если системному администратору необходимо при каждой загрузке изменять некоторые параметры через интерфейс `sysctl`, то список параметров и их значений можно записать в конфигурационный файл `/etc/sysctl.conf`, который прочитывается при каждой загрузке системы.

Статически и динамически собранные программы

В Linux исполняемые файлы можно условно поделить на две группы – те, которые содержат в себе весь код, необходимые для работы, и те, которым необходимы разделяемые библиотеки. Первые называют статически собранными бинарными файлами, вторые называют динамически собранными исполняемыми файлами.

Статически собранные программы характеризуются тем, что могут корректно функционировать в любых условиях, и не зависят от наличия или отсутствия разделяемых библиотек, что может оказаться полезным в ситуациях, когда возникают конфликты версий разделяемых библиотек, или когда системные библиотеки повреждены или недоступны (например во время восстановления операционной системы после серьезного сбоя). К недостаткам таких исполняемых файлов следует отнести то, что они имеют значительный размер и для обновления программы необходимо полностью заменить ее исполняемый файл – например, если несколько статически собранных программ, которые работают с архивами ZIP, содержат ошибку, то для исправления ошибки необходимо

заменить все эти программы, что может быть затруднено (например, будет трудно точно установить, какие именно программы содержат ошибочный код и нуждаются в обновлении). Кроме того, статически собранные программы не умеют совместно использовать совпадающие участки кода, что ведет к излишнему расходу системных ресурсов.

Динамически собранные исполняемые файлы для корректной работы требуют наличия файлов разделяемых библиотек, и соответственно при их отсутствии/повреждении не могут корректно функционировать, но зато для обновления программы и исправления ошибки часто оказывается достаточным просто заменить соответствующую разделяемую библиотеку, после чего ошибка исчезает во всех программах, которые эту библиотеку используют динамически. Динамически связанные программы также значительно меньше по объему, чем статически связанные, и код разделяемых библиотек может использоваться одновременно многими программами – что позволяет экономить системные ресурсы.

Подавляющее большинство программ в современных дистрибутивах Linux являются динамически собранными. Определить тип исполняемого файла (статический ли он либо с динамическим связыванием) можно, например, с помощью команды *ldd*:

```
[dalth@viking dalth]$ ldd /bin/su
linux-gate.so.1 => (0xffffe000)
libpam.so.0 => /lib/libpam.so.0 (0x4ce08000)
libpam_misc.so.0 => /lib/libpam_misc.so.0 (0x4cb3c000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4e3a2000)
libdl.so.2 => /lib/libdl.so.2 (0x4ca49000)
libc.so.6 => /lib/tls/libc.so.6 (0x4c907000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x4c8ee000)
[dalth@viking dalth]$ ldd /sbin/devlabel
not a dynamic executable
[dalth@viking dalth]$
```

В данном случае мы видим, что исполняемый файл */bin/su* использует динамическое связывание, а исполняемый файл */sbin/devlabel* собран статическим образом.

Системная библиотека GNU libc

Основная системная библиотека, которая так или иначе используется практически всеми программами, называется *glibc* (GNU libc). Основными задачами *glibc* являются обеспечение взаимодействия между ядром и пользовательскими процессами, поддержка локализации и многие другие распространенные действия.

На нижнем уровне прикладные процессы могут обратиться к функциям ядра посредством системных вызовов (*syscall*). Фактически *syscall* – это вызов прерывания 80h с установленными параметрами, описывающими параметры для этого системного вызова. Те функции *glibc*, которые должны обратиться к ядру, в большинстве случаев просто устанавливают параметры для соответствующего системного вызова и вызывают *int80h*.

Большинство программ используют динамически загружаемую библиотеку *glibc*, но некоторые приложения, которые должны работать вне зависимости от наличия файловой системы, где расположена динамически загружаемая реализация *libc*, используют статическое связывание, когда весь программный код, необходимый

для их работы, содержится в исполняемом файле программы. В основном к таким программам относятся утилиты, используемые при загрузке системы совместно с `initrd` – например, статические варианты утилит `insmod`, `lvm` или `devlabel`, а также командные оболочки для первичной загрузки или восстановления системы – например `sash` – `standalone shell`, часто используемый при восстановлении системы после серьезного сбоя или `nash`, используемый при выполнении сценариев загрузки после монтирования `initrd`, но до монтирования корневой файловой системы, когда разделяемая версия `glibc` еще недоступна.

LD, Shared Library, SO и много страшных слов

Существует набор базовых действий, которые практически любая программа выполняет одинаково – открытие файла, чтение и запись данных и тому подобное. Разделяемые библиотеки предназначены для того, чтобы предоставить прикладным программам готовые интерфейсы функций для выполнения каких-либо более-менее стандартных действий. Разделяемая библиотека, как понятно из названия, может использоваться множеством программ. В настоящий момент стандартным форматом для разделяемых библиотек в Linux является ELF (Executable Linked Format).

Каждый файл ELF имеет заголовок, в котором описывается, какие секции содержит этот файл. Секции объединяют однотипные данные, и их детальное описание можно прочесть в справочном руководстве [`man elf`]. Мы же выделим следующую информацию: каждая библиотека содержит список имен переменных и функций, которые она содержит и предоставляет другим (экспортирует) и список переменных и функций, которые необходимо взять в других библиотеках, а также секции инициализации и деинициализации. Экспортируемые и импортируемые объекты (переменные и функции) называют *символами* библиотеки.

Большинство исполняемых файлов программ также имеют формат ELF, и на самом деле отличаются от библиотек в основном тем, что не имеют экспортируемых функций. Загрузчик ELF (он же `dl`, `dynamic linker` и `dynamic loader`) умеет загружать в память код ELF-файла, анализировать его структуру для определения списков экспортируемых и импортируемых символов и загружать необходимые для работы программы библиотеки.

Когда пользователь пытается запустить какую-либо программу, первым начинает работу загрузчик ELF. Он загружает в память процесса бинарный файл и выделяет, какие символы и из каких библиотек необходимо догрузить в память. После дозагрузки каждой библиотеки загрузчик связывает символы (проставляет реальные адреса) из загруженной библиотеки и повторяет цикл анализа на предмет того, какую библиотеку нужно загрузить. Когда все нужные библиотеки загружены, загрузчик передает управление коду инициализации каждой из загруженных библиотек в порядке, обратном загрузке, после чего передает управление коду программы. По завершении программы загрузчик снова “проходится” по всем библиотекам и вызывает их функции деинициализации. Если на этапе загрузки какой-либо библиотеке возникает ошибка, загрузчик сообщит об этом пользователю. Наиболее типичные ошибки `dl` – это не найденный файл библиотеки или неразрешимый символ (символ не был найден в библиотеке, в которой ожидался).

Вполне естественно, что загрузчик ищет библиотеки не по всей файловой системе, а только в определенных каталогах. Это каталоги `/lib`, `/usr/lib` и те, которые были перечислены системным администратором в файле `/etc/ld.so.conf`.

Уточним, что этот файл на самом деле используется только системной утилитой `ldconfig`, сам же загрузчик использует кэш-файл `/etc/ld.so.cache`. Обновить этот кэш-файл можно путем простого запуска `ldconfig` без параметров. Следствием этого является то, что если вы установили в систему новые библиотеки, не мешает вызвать `ldconfig`.

В некоторых дистрибутивах есть возможность включать в `ld.so.conf` дополнительные файлы без его изменения. Для этого в `ld.so.conf` включается специальная строка вида:

```
include ld.so.conf.d/*.conf
```

Это приводит к тому, что каталоги, перечисленные в файлах с расширением `conf`, расположенных в каталоге `/etc/ld.so.conf.d` будут использованы для поиска разделяемых библиотек:

```
[root@vikings dalth]# cat /etc/ld.so.conf
include ld.so.conf.d/*.conf
/usr/lib/mysql
/usr/X11R6/lib
/usr/lib/qt-3.3/lib
[root@vikings dalth]# ls /etc/ld.so.conf.d/
oracle
[root@vikings dalth]# cat /etc/ld.so.conf.d/oracle
/opt/oracle/9i/lib
[root@vikings dalth]#
```

Нередко возникает ситуация, когда пользователю необходимо запустить какую-либо программу, которая не находится в каталогах, описанных в `/etc/ld.so.conf`. В таких ситуациях можно воспользоваться специальным “люком”, оставленным разработчиками `ld` специально для таких случаев: дело в том, что кроме загрузки библиотек с использованием данных из `ld.so.cache` загрузчик проверяет факт наличия библиотеки с указанным именем в каталогах, перечисленных в переменной среды `LD_LIBRARY_PATH`.

Разработчики часто используют еще одну возможность `ld`: если файл некоторой разделяемой библиотеки указан в переменной `LD_PRELOAD`, эта библиотека принудительно загружается и ее символы считаются более “приоритетными” и перекрывают одноименные символы, если таковые существуют в других библиотеках, загружаемых `ld` при запуске на выполнение бинарного файла ELF.

Попробуем рассмотреть примеры использования указанных возможностей `ld`: пусть есть некоторый программный продукт, в состав которого кроме собственно исполняемых программ входят разделяемые библиотеки (например, таковы практически все продукты, разработанные с помощью Borland Kylix). Если мы установим такой пакет, например, в `/opt/program`, его исполняемые файлы в `/opt/program/bin` а разделяемые библиотеки в `/opt/program/lib`, то программа, скорее всего, не будет запускаться, поскольку не сможет загрузить необходимых библиотек. Для того, чтобы программы пакета начали запускаться, мы должны “объяснить” `ld` где именно искать библиотеки. Рассмотрим возможные способы, которыми мы можем воздействовать на `ld` чтобы добиться нужного нам результата.

Первый способ – указать каталог с библиотеками перед запуском программы и уже затем запустить программу (`ld` воспользуется значением переменной для того, чтобы попытаться найти библиотеки по указанному пути):

```
$ export LD_LIBRARY_PATH=/opt/program/lib
$ /opt/program/bin/filename
```


Второй способ – добавить каталог `/opt/program/lib` в файл `/etc/ld.so.conf` и запустить `ldconfig`, решив проблему с невозможностью нахождения этих библиотек для всех программ сразу:

```
$ su -
# echo /opt/program/lib >>/etc/ld.so.conf
# ldconfig
# exit
$ /opr/program/bin/filename
```

Можно также воспользоваться возможностью принудительной загрузки тех библиотек, которые необходимы программе для запуска:

```
$ export LD_PRELOAD=/opt/program/lib/*
$ /opr/program/bin/filename
```

Большая часть кода разделяемых библиотек находится в кэше и становится доступна процессам через отображение файла в память. Это отображение делается с правами доступа “только чтение”, что защищает код библиотек от переписывания его неправильно работающими или просто злонамеренными программами.

Информация о процессах и файловая систем /proc

Ядро и его подсистемы очень важны, но большинство пользы приносят прикладные задачи, поэтому мониторинг состояния задач (процессов) – очень важная часть работы системного администратора. В Linux получить информацию о процессах можно через файлы и каталоги файловой системы `procfs`, как правило монтируемой к каталогу `/proc`.

Каждому процессу сопоставляется в `/proc` отдельный каталог, имя которого совпадает со значением PID процесса. Файлы в этом каталоге предоставляют информацию о соответствующем процессе. Таблица приводит список файлов и их назначение:

Имя файла	Формат	Назначение
<code>cmdline</code>	строка, разделенная символами <code>\0</code>	Представляет командную строку, которой был запущен процесс. параметры командной строки отделяются друг от друга символами <code>\0</code>
<code>environ</code>	строка, разделенная символами <code>\0</code>	Представляет список переменных окружения для указанного процесса
<code>exe</code>	символьная ссылка	Ссылается на исполняемый файл процесса
<code>maps</code>	несколько строк	Список отображенных в память процесса файлов
<code>mem</code>	бинарный	Прямой доступ к адресному пространству процесса
<code>mounts</code>	несколько строк	Список примонтированных файловых систем, доступных процессу
<code>stat</code>	строка числовых значений	Статистика активности процесса
<code>statm</code>	строка числовых значений	Статистика по использованию памяти процессом
<code>cwd</code>	символьная ссылка	Ссылается на каталог, который является текущим для процесса

Имя файла	Формат	Назначение
fd/*	символьные ссылки	Имена файлов подкаталога fd соответствуют открытым процессом дескрипторам файлов. Символьные ссылки указывают на соответствующие файлы
root	символьная ссылка	Ссылается на каталог, который процесс считает корневым
status	несколько строк	Описание состояния процесса

Все указанные данные полностью соответствуют тому, что показала бы программа `ps`, будучи запущенной в тот момент, когда просматривается соответствующий файл, поскольку утилита `ps` на самом деле просто читает данные из соответствующих файлов в `/proc`.

Создание процессов

Linux на самом деле поддерживает только один внутренний механизм создания процессов – механизм `fork+exec`. Любой процесс, который хочет создать еще один процесс, должен сначала создать свою копию с помощью системного вызова `fork`, после чего порожденный процесс, который является полной копией предыдущего за исключением нескольких параметров, таких как PID (Process ID) и PPID (Parent Process ID) использует системный вызов `exec` для того, чтобы загрузить в свое адресное пространство код новой программы и начать его выполнение. Соответственно, все процессы организуют дерево, когда у каждого процесса есть родительский процесс (исключение составляет процесс `init`, запущенный ядром на этапе загрузки).

Когда процесс завершается, код его завершения возвращается родительскому процессу. До тех пор, пока код завершения процесса не будет прочитан родительским процессом, запись об этом процессе продолжает существовать в таблице процессов в ядре. Такой процесс (уже завершившийся, но еще числящийся в таблице процессов) называют процессом – зомби (zombie process). Завершить `zombie process` может родительский процесс, прочтя код его завершения. Если у вас в системе появилось множество зомби – процессов, это скорее всего означает ошибку в программе, породившей этот процесс. Удалить процесс – зомби можно только удалив его родительский процесс.

Нередко бывает, что родительский процесс завершается раньше, чем дочерний, и тогда для дочернего процесса объявляется родительским процесс `init`, и поэтому в системе никогда не бывает процессов-”сирот”, т.е. тех, кто не имеет родителя и чей код завершения некому прочесть.

Секреты /dev

Ядро Linux реализует поддержку двух типов устройств – символьных и блочных. Основное их отличие в том, что для блочных устройств операции ввода вывода осуществляются не отдельными байтами (символами), а блоками фиксированного размера.

В Linux вся работа с устройствами ведется через специальные файлы, которые обычно расположены в каталоге `/dev`. Специальные файлы не содержат данных, а просто служат точками, через которые можно обратиться к драйверу соответствующего устройства. У каждого специального файла есть три

характеристики – тип устройства (character или block), старший номер устройства (major number) и младший номер (minor number). Для примера, посмотрим на содержимое каталога /dev:

```
[dalth@vikings proc]$ ls -lL /dev/hd* /dev/ttyS*
brw----- 1 root root 3, 0 Окт 1 20:16 /dev/hda
brw----- 1 root root 3, 1 Окт 1 20:16 /dev/hda1
brw----- 1 root root 3, 2 Окт 1 20:16 /dev/hda2
brw----- 1 dalth disk 22, 0 Янв 1 1970 /dev/hdc
crw----- 1 root root 4, 64 Янв 1 1970 /dev/ttyS0
crw----- 1 root root 4, 65 Янв 1 1970 /dev/ttyS1
crw----- 1 root root 4, 66 Янв 1 1970 /dev/ttyS2
```

Как видно, в листинге присутствует описание семи устройств, четырех блочных и трех символьных. Для каждого файла можно увидеть его тип (первая буква в списке прав доступа), пользователя-владельца, группу-владельца, major number, minor number, дату модификации и имя файла.

Для поддержки работы с устройствами в ядре хранятся две таблицы, одна для списка символьных устройств, другая для списка блочных устройств. Каждая строка таблицы сопоставлена какой-то разновидности устройств соответствующего типа – например, для типа “символьные устройства” можно выделить следующие разновидности: COM-порты, LPT-порты, PS/2-мыши, USB-мыши и т.д., для типа “блочные устройства” можно выделить SCSI-диски, IDE-диски, SCSI-CD-приводы, виртуальные диски которыми представляются RAID-контроллеры и т.п.

Каждая ячейка в этих системных таблицах сопоставляется конкретному экземпляру устройства. Таким образом, с точки зрения ядра каждое устройство оказывается однозначно проидентифицировано тремя параметрами – типом устройства (блочное или символьное) и двумя числами – номерами строки и номером столбца таблицы, в которой хранится ссылка на драйвер этого устройства.

Пример таблицы символьных устройств

	0	1	...	63	64	65	66	...	175	...
4					COM1	COM2	COM3			
6	LPT1	LPT2								
10		Мышь PS/2		Диспетчер томов LVM					Слот AGP	
14	Микшер первой зв. карты									
195	Первая видеокарта NVidia									

Пример таблицы блочных устройств

	0	1	2	...	16	...	64	65	...
3	IDE Primary Master	Раздел 1 на IDE Primary Master	Раздел 2 на IDE Primary Master		Раздел 16 на IDE Primary Master		IDE Primary Slave	Раздел 1 на IDE Primary Slave	
13	SCSI диск 1	Раздел 1 на SCSI-диске 1	Раздел 2 на SCSI-диске 1		SCSI диск 2		SCSI диск 4	Раздел 1 на SCSI-диске 4	

При попытке обращения к такому специальному файлу ядро переадресует

обращение через нужный драйвер на устройство в соответствии с теми данными, которые указаны в таблице устройств, причем конкретная таблица устройств будет выбрана в зависимости от типа устройства, строка из таблицы будет выбрана по major number, и столбец будет выбран по minor number. Если мы посмотрим на примеры наших таблиц, то увидим, что обращение на специальный файл `/dev/ttyS1`, который представляет символьное устройство со старшим номером 4 и младшим номером 65 будет адресовано на последовательный порт COM2, а обращение к файлу `/dev/hda2` (блочное устройство со старшим номером 3 и младшим номером 2) будет адресовано на 2-й раздел жесткого диска IDE, работающего в режиме primary master.

Статическая организация каталога `/dev`

В настоящее время существует два подхода к организации `/dev` – статическая организация и динамическая организация. В первом случае в каталоге `/dev` заранее создаются специальные файлы для всех возможных устройств вне зависимости от того, загружен драйвер соответствующего устройства или нет. Во втором случае специальные файлы в `/dev` создаются по мере инициализации устройств и загрузки драйверов, и удаляются при выгрузке соответствующего драйвера или удалении устройства.

Процесс работы со статическим `/dev` особых проблем не вызывает – системный администратор при необходимости просто создает отсутствующие файлы командой **`mknod`** или **`MAKEDEV`**. В том случае, когда какая-либо программа обращается к устройству, чей драйвер не загружен (или загружен, но ни одного соответствующего устройства не было обнаружено), операционная система возвращает ошибку при попытке открытия файла такого “неверного” устройства. Ниже приведен пример создания специального файла, соответствующего блочному устройству с мажором 8 и минором 33 и попытка его использования (отметим, что этот специальный файл соответствует разделу на жестком диске, который не существует на тестовой машине, где выполнялись эти команды):

```
[root@vikings root]# cd /dev
[root@vikings dev]# ls -l /dev/hda33
ls: /dev/hda33: No such file or directory
[root@vikings dev]#
[root@vikings dev]# mknod hda33 b 8 33
[root@vikings dev]#
[root@vikings dev]# ls -l hda33
brw-r--r-- 1 root root 8, 33 Окт 11 09:27 hda33
[root@vikings dev]#
[root@vikings dev]# dd if=hda33 of=/dev/null
dd: opening `hda33': No such device or address
[root@vikings dev]#
```

Сообщение **No such device or address** как раз и означает, что записи для данного устройства в таблице блочных устройств не существует.

Ядро Linux совместно с некоторыми системными утилитами поддерживает такую интересную возможность, как загрузка драйверов “по требованию”. Реализуется это следующим образом – в момент, когда какая-либо программа пытается открыть специальный файл, не связанный ни с каким драйвером, ядро делает попытку подобрать соответствующий драйвер самостоятельно. Необходимый драйвер для каждого специального файла определяется в файле `/etc/modules.conf` путем задания специального алиаса (alias) для модуля. Для

активизации автоматической загрузки драйвера какого-либо символического устройства в большинстве случаев достаточно просто записать в `/etc/modules.conf` строку следующего вида:

```
alias char-major-X-Y имя_драйвера
```

Для блочных устройств соответствующая запись слегка меняет свою форму:

```
alias block-major-X-Y имя_драйвера
```

X и Y – это major и minor специального файла, попытка открыть который должна активизировать автоматическую загрузку драйвера. Владельцы видеокарт на чипе nVidia могут увидеть этот подход в действии – программа инсталляции драйвера nVidia автоматически прописывает в `modules.conf` запись для загрузки «по требованию» той части драйвера, которая работает в режиме ядра.

Вместо X или Y может также быть подставлен символ “*”, означающий “любое число”. Например, пусть в `modules.conf` будет написан следующий текст:

```
alias char-major-81-* bttv
```

Тогда при обращении к любому символическому устройству с major number равным 81 и которое не ассоциировано ни с каким драйвером, система попытается загрузить драйвер bttv (драйвер TV-тюнера на основе чипа bt848).

Эта возможность обеспечивает Linux возможность плавной загрузки и эффективного использования ресурсов – драйвер не загружается, пока в нем не возникнет необходимости. К сожалению, за простоту этой схемы приходится платить большим количеством специальных файлов в `/dev`.

Что такое DevFS

Для того, чтобы избавить администратора от ручного создания специальных файлов и для уменьшения количества файлов в `/dev` был реализован второй способ организации `/dev` – динамическое создание специальных файлов процессе загрузки драйверов. Реализовано это было следующим образом:

Ядро монтирует к каталогу `/dev` специальную файловую систему, называемую **devfs** – эта файловая система хранится целиком в оперативной памяти и не занимает никакого места на диске. Когда какой-либо драйвер в процессе загрузки или работы обнаруживает обслуживаемое им устройство, он регистрирует это устройство и сообщает о нем драйверу devfs. Драйвер devfs создает специальный файл, который виден прикладным программам и может быть корректно открыт. При выгрузке же драйвер устройства сообщает devfs о том, что соответствующее устройство уже не активно, и драйвер devfs удаляет запись о соответствующем специальном файле из файловой системы devfs.

Файловая система devfs отличается тем, что как правило специальный файл для устройства создается с длинным путем – например, для раздела на scsi-диске путь может выглядеть примерно так: **`/dev/scsi/host1/bus1/target3/lun4/partition2`**

Эта особенность является весьма важным плюсом devfs, поскольку она позволяет адресовать дисковые устройства путем указания логического пути их подключения и избежать смены имен SCSI-дисков в некоторых случаях (об этих случаях будет рассказано позднее).

Для того, чтобы организовать более прозрачную структуру каталогов и файлов устройств, используется специальный демон **devfsd**. Он взаимодействует с драйвером devfs и ядром и в процессе активизации и деактивизации устройств он создает и удаляет символичные ссылки вида `/dev/disks/disc0` или `/dev/hda1`.

Надо отметить, что схема динамического `/dev` в некотором смысле близка к той

организации каталога `/dev`, которая используется некоторыми коммерческими UNIX-системами (например, в Solaris), когда есть виртуальная файловая система `/devices`, и на ее файлы создаются ссылки из `/dev`, только в Linux роль программы `cfgadm` играет демон `devfsd`, и все изменения в состав `/dev` вносятся автоматически.

С помощью `devfsd` файловая система `devfs` также реализует автоматическую загрузку модулей, но в этом случае выбор модуля идет не через комбинацию `type/major/minor`, а путем указания имени запрошенного файла – когда приложение пытается открыть несуществующий файл устройства, `devfs` передает имя запрошенного файла демону `devfsd`, и последний загружает необходимые модули, например такой код в файле `modules.devfs`:

```
alias /dev/nvidia* nvidia
```

Приведет к тому, что при попытке обращения к любому файлу, чей полный путь начинается строкой `/dev/nvidia`, будет произведена попытка загрузить драйвер `nvidia.o` (для ядра 2.6 `nvidia.ko`)

В принципе, на сегодняшний день выбор того, каким именно образом необходимо организовывать `/dev`, остается за пользователем и создателем дистрибутива. Например, в Mandrake Linux используется `devfs`, а в RedHat, Fedora и SUSE каталог `/dev` организован статическим образом, а опытные пользователи часто меняют способ организации `/dev` в зависимости от своих предпочтений.

Немного о UDEV

В современных дистрибутивах и ядрах поддержка `devfs/devfsd` отключена, и на смену этой паре пришел специальный демон, называемый `udev`. В отличие от `devfsd`, который требовал поддержки со стороны ядра, `udev` такой поддержки не требует. При инициализации устройства ядро подает сигнал через файловую систему `sysfs`, и демон `udev`, получив сигнал об этом событии, самостоятельно создает соответствующий специальный файл устройства в каталоге `/dev` в соответствии с правилами, описанными в его конфигурационных файлах. При необходимости в этих файлах можно указать например вызов некоторой внешней программы, создание символьной ссылки и так далее.

Например, если некоторое устройство после подключения перед началом работы требует дополнительной настройки с использованием внешних программ, можно создать соответствующее правило для `udev`, в котором будет указано какую программу вызвать и какие параметры ей необходимо передать – в частности, это может потребоваться для data-кабелей к некоторым мобильным телефонам Nokia, для устройств которым для корректной работы требуется `firmware`, или для сохранения или восстановления текущих настроек устройства.

Тем не менее, несмотря на внешние отличия между статической организацией `/dev`, `devfs` и `udev`, следует помнить что это всего лишь способ заполнения каталога `/dev`, и во всех случаях в конечном итоге на файловой системе создаются те же самые файлы символьных и блочных устройств.

Блочные устройства

Любое устройство, подключенное к компьютеру, имеет свое назначение, и блочные устройства в большинстве своем предназначаются для хранения информации. Как организована работа с блочными устройствами в Linux?

Во-первых, следует определиться с типами блочных устройств. Их следует

поделить на две категории: к первой отнесем логические (виртуальные) устройства (loop-устройства, software RAID-устройства, устройства Volume Management, поддержка различных таблиц разделов), ко второй категории – физические устройства (SCSI диски и CD-ROM'ы, IDE-диски, USB-storage, RAM-диск).

Виртуальные устройства являются на самом деле просто оберткой, дополнительным слоем. В реальности драйверы логических устройств не работают с периферийными устройствами напрямую, они лишь переадресовывают запросы на драйверы других логических или физических устройств.

Драйверы физических устройств работают совместно с драйверами контроллеров, позволяя производить доступ к соответствующим устройствам на блочном уровне и предоставляя тем самым фактически прямой доступ к носителю – но, поскольку в большинстве случаев дисковые устройства имеют значительный объем, они часто делятся на *разделы*. Раздел является постоянным непрерывным фрагментом дискового пространства, местоположение которого на жестком диске записано в специальной области диска – таблице разделов.

Существует множество различных форматов разбиения диска на разделы – например, DOS partition table, BSD disklabels, UnixWare slices и многие другие. Как правило, во всех случаях соответствующая спецификация предусматривает возможность перечисления ограниченного количества разделов путем указания номеров первой и последней дорожек, занимаемых каждым из разделов. Каждый раздел видится как отдельное блочное устройство.

По традиции имена блочных устройств, соответствующих IDE-дискам и созданным на них разделам начинаются с *hd* и имеют вид */dev/hd<N>[<M>]* где **N** – это буква, зависящая от контроллера и канала IDE, к которому подключено устройство, и режима устройства (master/slave). **M** – это некоторое число от 1 до 63 (фактически номер раздела на диске). Если число не указано, подразумевается весь диск. SCSI-дискам в */dev* присваиваются имена *sda*, *sdb*, *sdc* и т.д. Ниже приводится небольшая таблица соответствия устройств и имен специальных файлов для IDE-дисков:

Контроллер	Канал	Режим		Имя файла
1	1	master	primary master	hda
1	1	slave	primary slave	hdb
1	2	master	secondary master	hdc
1	2	slave	secondary slave	hdd
2	1	master	tertiary master	hde
2	1	slave	tertiary slave	hdf
2	2	master	quaternary master	hdg
2	2	slave	quaternary slave	hdh

Когда драйвер блочного устройства, поддерживающего разбиение на разделы, в процессе загрузки или работы обнаруживает обслуживаемое им устройство, он считывает с него таблицу разделов, определяет ее разновидность и составляет

таблицу разделов, запоминая начало и конец каждого из разделов. Впоследствии, если какая-либо программа производит обращение не непосредственно к физическому устройству, а к разделу на этом устройстве, драйвер с использованием построенной таблицы разделов определяет реальный адрес блока, над которым нужно произвести запрошенную операцию ввода/вывода.

Особой разновидностью раздела можно назвать расширенный (*extended*) раздел DOS. Расширенный разделы DOS может быть разбит на произвольное количество вложенных разделов, но в настоящий момент без использования LVM ядро Linux поддерживает до 63 разделов на IDE-диске и до 15 разделов на SCSI-диске. Такое ограничение связано с распределением мажоров и миноров блочных устройств.

Когда используется разбиение диска на разделы с использованием таблицы разделов DOS, следует помнить, что на жестком диске может быть не более 4 первичных разделов. Если администратору нужно, чтобы на жестком диске было более 4 разделов, необходимо объявить один из первичных разделов как расширенный раздел. Первичные разделы при использовании таблицы разделов DOS нумеруются от 1 до 4, логические разделы нумеруются начиная с 5, вне зависимости от количества первичных разделов. Только один из первичных разделов может быть объявлен расширенным.

Рассмотрим вывод команды `fdisk`, которая обычно используется в Linux для разбиения диска на разделы:

```
[root@stend root]# fdisk /dev/hda -l

Disk /dev/hda: 6442 MB, 6442450944 bytes
16 heads, 63 sectors/track, 12483 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1    *           1           203     102280+   83  Linux
/dev/hda2             204          2032      921816   83  Linux
/dev/hda3          2033          3072     524160   82  Linux swap
/dev/hda4          3073          12483    4743144   f  W95 Ext'd (LBA)
/dev/hda5          3073          4088     512032+   83  Linux
/dev/hda6          4089          12483    4231048+   83  Linux

[root@stend root]#
```

Как видно, на жестком диске IDE созданы 6 разделов, из них 4 первичных (разделы с номерами от 1 до 4) и два логических раздела с номерами 5 и 6, созданных внутри *extended*-раздела `hda4`.

Таблица разделов диска не может быть изменена в том случае, если хотя-бы один из разделов этого диска используется. В этом случае ядро продолжает использовать “старую” разметку (с которой оно работало до изменения), а изменения записываются на диск и вступают в силу после перезагрузки компьютера.

Если нет возможности запустить программу `fdisk`, то для получения данных о разметке блочных устройств на разделы и о состоянии блочных устройств, доступных в настоящий момент, можно использовать некоторые файлы из файловой системы `/proc`:


```
[dalth@viking dalth]$ cat /proc/partitions
major minor #blocks name

    3      0   78150744 hda
    3      1    1084356 hda1
    3      2   77063805 hda2
  253      0    1048576 dm-0
  253      1    1048576 dm-1
  253      2   10485760 dm-2
  253      3   10485760 dm-3
  253      4    1048576 dm-4
  253      5   41943040 dm-5
  253      6     360448 dm-6
  253      7     258048 dm-7
  253      8     258048 dm-8
  253      9     258048 dm-9
  253     10      53248 dm-10
    7      0     651884 loop0
    7      1     650198 loop1
    7      2     653336 loop2
    7      3     198962 loop3
```

Например, эти данные могут быть использованы для восстановления таблицы разделов, если системный администратор по ошибке ее исправил.

Интересной особенностью таблицы разделов диска является то, что она не всегда изменяема. Это приводит к тому, что во многих случаях невозможно изменить размер какого-либо раздела или внести какие-либо другие изменения в таблицу разделов без перезагрузки – т.е. системный администратор может делать любые изменения, но они вступят в силу только после перезагрузки системы.

Распределение мажоров и миноров IDE дисков

IDE-диски в настоящее время наиболее часто используются в офисных и домашних компьютерах, поэтому знать особенности распределения мажоров и миноров для этих типов устройств достаточно важно. IDE-устройства отличаются низкой ценой и неплохой скоростью передачи данных, но у этой шины есть архитектурные недостатки – например, все устройства работают со скоростью самого медленного из них, на один шлейф (на один канал) можно подключить только 2 устройства.

Всем устройствам, находящимся на одном канале IDE, присвоен один мажор. В настоящий момент ядро Linux выделяет для каждого устройства 64 минора, из которых первый минор зарезервирован для всего диска, и 63 минора остается для идентификации разделов. Таким образом, для IDE-дисков мажор идентифицирует канал, а по минору можно определить номер устройства на канале (режим master/slave) и номер раздела. Более детально это можно увидеть в следующей таблице:

Распределение номеров устройств для IDE-дисков					
Канал	Устройство	Раздел	Major number	Minor number	Имя в /dev
1	1	Весь диск	3	0	/dev/hda
		Раздел 1		1	/dev/hda1
		Раздел 2		2	/dev/hda2
		Раздел 3		3	/dev/hda3
		Раздел 4		4	/dev/hda4
		...			
		Раздел 63		63	/dev/hda63
	2	Весь диск		64	/dev/hdb
		Раздел 1		65	/dev/hdb1
		Раздел 2		66	/dev/hdb2
		Раздел 3		67	/dev/hdb3
		...			
		Раздел 63		127	/dev/hdb63
2	1	Весь диск	22	0	/dev/hdc
		Разделы		1-63	/dev/hdc[1..63]
	2	Весь диск		64	/dev/hdd
		Разделы		65-127	/dev/hdd[1..63]

По таблице становится видно, что на каждый 64-й минор происходит смена физического диска. Драйвер IDE в текущей версии ядра Linux поддерживает до четырех каналов IDE – т.е. до 8 устройств, по два устройства на канал..

Рекомендации, которую можно дать владельцам компьютеров с интерфейсом IDE: устройства по возможности рекомендуется держать на разных каналах. Если нет свободных каналов, то “быстрые” устройства лучше подключать на один канал, медленные – на другой.

Распределение мажоров и миноров для SCSI-дисков

Шина SCSI свободна от некоторых недостатков IDE, например количество устройств на одном канале может быть 15 (на самом деле 16, но одним устройством считается сам контроллер), все SCSI устройства работают на своей максимальной скорости, и ограничены только возможностями шины и контроллера – но SCSI-устройства и дороже, и поэтому шина SCSI используется в основном на серверах и рабочих станциях. Для SCSI-дисков ситуация немного меняется – мажоры не привязаны к контроллерам (т.е. один major number может использоваться дисками с разных хост-адаптеров). На каждом SCSI-диске система поддерживает до 16 разделов, а нумерация дисков производится в порядке их подключения по схеме, аналогичной IDE-дискам – но только переход на следующий диск происходит на каждом 16-м миноре (т.е. разделы на SCSI-дисках нумеруются от 1 до 15). Увидеть это можно в следующей таблице и

листинге /dev:

Распределение номеров устройств для SCSI-дисков				
Номер диска в порядке подключения	Major number	Minor number	Раздел	Имя файла
1	8	0	Весь диск	sda
		1	Первый первичный	sda1
		2	Второй первичный	sda2
		3	Третий первичный	sda3
		4	Четвертый первичный	sda4
		5	Первый логический	sda5
		...		
		15	Одиннадцатый логический	sda15
2	8	16	Весь диск	sdb
		17	Первый первичный	sdb1
		18	Второй первичный	sdb2
		19	Третий первичный	sdb3
		20	Четвертый первичный	sdb4
		21	Первый логический	sdb5
		...		
		31	Одиннадцатый логический	sdb15

В текущей версии ядро Linux поддерживает до 4096 SCSI-дисков – и для абсолютного большинства компьютеров этого должно быть достаточно. Очень важной особенностью SCSI-дисков является то, что мажор устройства, соответствующего физическому диску, не зависит от контроллера. В результате, если у вас есть 3 диска SCSI, то они всегда именуются sda, sdb и sdc, и если вы выключите компьютер и отключите первый диск, то после перезагрузки второй диск (который ранее назывался sdb) станет называться sda, а третий диск (который назывался sdc) станет называться sdb, поэтому при работе со SCSI-устройствами можно использовать devfs (которая позволяет адресовать диски через путь их подключения), либо использовать специализированные средства управления дисковым пространством, которые помечают носители и впоследствии правильно их идентифицируют даже после переименования устройств - например, средства md (software RAID) или LVM. Надо заметить, что в свете удаления devfs из основной ветви ядра, использование LVM стало фактически обязательным на серверах.

Устройства SATA и переход с IDE на PATA

SATA, «осовремененная» версия интерфейса IDE, по своей структуре приблизилась к подсистеме SCSI. Поэтому в целях унификации подсистемы ввода-вывода в ядре Linux поддержка SATA была реализована через интерфейс

SCSI-, соответственно SATA-диски и контроллеры видятся ядром (и пользователем) как SCSI-устройства.

В новейших ядрах линейки 2.6 также появилась возможность работы с IDE-дисками через подсистему SCSI-, и в новых дистрибутивах, таких как Fedora 7, даже обычные Parallel ATA (они же IDE) диски и контроллеры представляются как SCSI-устройства, что привело к унификации подсистемы дискового ввода-вывода, и теперь все диски и CD/DVD приводы для пользователя представляются как `/dev/sdX` или `/dev/scdX`. Драйверы для обычных контроллеров IDE, которые работают по новой схеме, начинаются с префикса `pata`, например: `pata_via` – это драйвер IDE-контроллеров с чипсетом VIA, а `pata_piix` – это драйвер для IDE-контроллеров Intel, работающие через подсистему SCSI.

А говоря проще, это означает следующее – если у вас новый дистрибутив или имеются SATA-диски, вы можете смело работать со всеми дисками как со SCSI-устройствами.

Logical Volume Manager

Использование таблиц разделов для управления дисковым пространством – достаточно часто используемое решение. К сожалению, оно не свободно от определенных недостатков – например, нет возможности расширить раздел или уменьшить его размер, нет возможности создать один раздела на нескольких дисках и т.д. Решить эту задачу призван LVM (Logical Volume Manager).

LVM работает следующим образом: пользователь может пометить какие-либо блочные устройства как разделы, используемые LVM. Каждое из таких помеченных блочных устройств (их называют физическими томами, или *physical volumes*) может быть присоединено к какой либо группе логических томов (*logical volume groups*). Внутри групп логических томов могут создаваться уже собственно логические тома (*logical volumes*). Дисковое пространство любого физического тома из некоторой группы может быть выделено любому логическому тому из этой группы. Реализовано это через так называемые “экстенты” (*extents*) дискового пространства. Физические тома LVM разбиваются на экстенты, после чего из экстентов и составляются логические тома. Именно за счет этого можно динамически менять конфигурацию дискового пространства – экстенст может быть удален из одного тома, и добавлен к другому. Каждый объект LVM – будь то логический том, физический том или группа томов, имеет свой уникальный идентификатор.

Осуществляется это комплексно драйвером *device mapper* и специализированными программами из пакета *lvml*. Эти программы читают файлы конфигурации и служебную информацию из заголовков физических томов, и на основании этой информации сообщают драйверу инструкции о том, из каких фрагментов каких блочных устройств каким именно образом должны быть скомбинированы логические тома, после чего драйвер для каждого логического тома создает отдельное блочное устройство. При обращении к такому блочному устройству *device mapper* определяет, на основании ранее переданных параметров к какому блоку какого блочного устройства на самом деле должен быть переадресован запрос, и запрашивает соответствующее блочное устройство для окончательного выполнения операции, после чего возвращает результат выполнения операции (прочитанные данные, сообщение об ошибке, код завершения операции) обратившейся программе.

Использование LVM позволяет гибко управлять распределением дискового

пространства и избежать ограничений, связанных с классическим распределением дискового пространства путем создания разделов на жестких дисках. Единственное правило, которое я бы советовал соблюдать при использовании LVM – не создавать корневой раздел системы на логическом томе LVM: инициализации тома LVM, на котором находится корневая файловая система, необходимо вмешательство некоторых утилит, которые находятся на еще не смонтированной корневой файловой системы. Это решаемая проблема, но она потребует некоторого опыта.

Ниже идет пример создания и инициализации физического тома, группы томов и примеры нескольких операций с логическими томами. На первом фрагменте протокола продемонстрирована инициализация таблицы разделов для использования LVM. Порядок действий для использования LVM в общем случае следующий: один или несколько разделов жесткого диска с помощью `fdisk` помечаются как разделы LVM. Затем эти разделы инициализируются и передаются в группы томов, после чего их дисковое пространство можно использовать для создания логических томов:

```
[root@inferno dalth]# fdisk /dev/hdb

The number of cylinders for this disk is set to 79408.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
 1) software that runs at boot time (e.g., old versions of LILO)
 2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): p

Disk /dev/hdb: 40.9 GB, 40982151168 bytes
16 heads, 63 sectors/track, 79408 cylinders
Units = cylinders of 1008 * 512 = 516096 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hdb1            1         79408     40021600+   8e  Linux LVM

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@inferno dalth]#
```

В приведенном выше выводе `fdisk` видно, что на IDE-диске primary slave создан один раздел типа LVM (код типа раздела 0x8E). В следующем листинге показан процесс инициализации физического тома и создания группы томов **aurora**, в которую включается инициализированный командой `pvccreate` физический том `/dev/hdb1`:

```
[root@inferno dalth]# pvcreate /dev/hdb1
No physical volume label read from /dev/hdb1
Physical volume "/dev/hdb1" successfully created
[root@inferno dalth]# vgcreate aurora /dev/hdb1
Volume group "aurora" successfully created
[root@inferno dalth]#
```

Третий фрагмент демонстрирует создание нескольких томов и изменение размеров томов, сделанное с помощью LVM. В этом примере создается

логический том размеров в 20GB, затем размер этого тома увеличивается до 30GB, создается еще один логический том, и после этого оба созданных логических тома удаляются.

```
[root@inferno dalth]#
[root@inferno dalth]# lvcreate -L 20G -n ftpdata aurora
Logical volume "ftpdata" created
[root@inferno dalth]#
[root@inferno dalth]# lvscan
ACTIVE                               '/dev/aurora/ftpdata' [20,00 GB] next free
(default)
[root@inferno dalth]#
[root@inferno dalth]# lvresize -L +10G /dev/aurora/ftpdata
Extending logical volume ftpdata to 30,00 GB
Logical volume ftpdata successfully resized
[root@inferno dalth]#
[root@inferno dalth]# lvscan
ACTIVE                               '/dev/aurora/ftpdata' [30,00 GB] next free
(default)
[root@inferno dalth]# lvcreate -L 8G -n home_dirs aurora
Logical volume "home_dirs" created
[root@inferno dalth]#
[root@inferno dalth]# lvscan
ACTIVE                               '/dev/aurora/ftpdata' [30,00 GB] next free
(default)
ACTIVE                               '/dev/aurora/home_dirs' [8,00 GB] next free
(default)
[root@inferno dalth]#
[root@inferno dalth]# lvremove /dev/aurora/ftpdata
Do you really want to remove active logical volume "ftpdata"?
[y/n]: y
Logical volume "ftpdata" successfully removed
[root@inferno dalth]#
[root@inferno dalth]# lvremove /dev/aurora/home_dirs
Do you really want to remove active logical volume "home_dirs"? [y/n]: y
Logical volume "home_dirs" successfully removed
[root@inferno dalth]#
```

Еще один фрагмент демонстрирует удаление группы томов и очистку физического тома:

```
[root@inferno dalth]# vgremove aurora
Volume group "aurora" successfully removed
[root@inferno dalth]# pvremove /dev/hdb1
Labels on physical volume "/dev/hdb1" successfully wiped
[root@inferno dalth]#
```

Каждый логический том LVM имеет свой собственный `minor number`, а `major number` для всех томов LVM равен 253. Для доступа к томам LVM можно создавать блочные устройства с помощью команды `mknod`, а можно воспользоваться возможностями, предоставляемыми утилитой `devlabel`. Эта утилита создает символичные ссылки и каталоги внутри подкаталога `/dev`, причем для каждой группы томов в `/dev` создается каталог с именем этой группы, а логические тома представляются символическими ссылками из этих каталогов на блочные устройства, обслуживаемые драйвером `device mapper`, и тогда любой том LVM можно адресовать следующим путем: `/dev/<имя_группы>/<имя_тома>`. На листинге ниже показан пример того, как можно распределить дисковое пространство с помощью LVM:

```
[root@vikings root]#
[root@vikings root]# ls -la /dev/chimera
lr-xr-xr-x 1 root root 23 Окт  8 13:53 opt -> /dev/mapper/chimera-
opt
lr-xr-xr-x 1 root root 24 Окт  8 13:53 swap ->
/dev/mapper/chimera-swap
lr-xr-xr-x 1 root root 24 Окт  8 13:53 temp ->
/dev/mapper/chimera-temp
lr-xr-xr-x 1 root root 23 Окт  8 13:53 usr -> /dev/mapper/chimera-
usr
lr-xr-xr-x 1 root root 23 Окт  8 13:53 var -> /dev/mapper/chimera-
var
[root@vikings root]#
[root@vikings root]# lvscan
ACTIVE          '/dev/chimera/swap' [1,00 GB] next free (default)
ACTIVE          '/dev/chimera/temp' [1,00 GB] next free (default)
ACTIVE          '/dev/chimera/usr' [10,00 GB] next free (default)
ACTIVE          '/dev/chimera/opt' [10,00 GB] next free (default)
ACTIVE          '/dev/chimera/var' [1,00 GB] next free (default)
[root@vikings root]#
[root@vikings root]# mount | grep chimera
/dev/mapper/chimera-var on /var type ext3 (rw)
/dev/mapper/chimera-temp on /tmp type ext3 (rw)
/dev/mapper/chimera-usr on /usr type ext3 (rw)
/dev/mapper/chimera-opt on /opt type ext3 (rw)
[root@vikings root]#
```

Таким образом, возможности LVM позволяют системному администратору максимально эффективно использовать дисковое пространство, оперативно реагируя на меняющиеся условия эксплуатации. Еще одной интересной возможностью LVM является так называемый multipath I/O. В случае активации соответствующей опции в ядре device mapper знает о том, что физический том с некоторым UUID может быть доступен через несколько контроллеров, и в случае отказа одного контроллера динамически происходит переключение ввода-вывода на другой. Опытные системные администраторы также оценят такую возможность, как создание снимка (snapshot) логического тома: при создании снимка создается моментальная копия логического тома, которая начинает «жить» независимо от того тома, на основе которого она была создана:

```
# xfs_freeze /home
# lvcreate -s -L 10G -n home_snapshot /dev/chimera/home
# xfs_freeze -u /home
# dd if=/dev/chimera/home_snapshot of=/dev/st0
# lvremove /dev/chimera/home_snapshot
```

В приведенном примере системный администратор «замораживает» файловую систему XFS, при этом драйвер XFS сбрасывает все закешированные операции на диск, и после этого блокирует все процессы, которые пытаются писать на «замороженную» файловую систему. Затем системный администратор создает снимок тома home из группы томов chimera, на котором «живет» файловая система /home, и этот снимок называет home_snapshot, при этом на удержание копии измененных данных выделяется 10 гигабайт дискового пространства. После создания снимка файловая система /home размораживается, но каждый раз, когда будет переписываться какой-либо блок логического тома home, первоначальная версия изменяемого блока будет копироваться в те 10GB дискового пространства, которые мы выделили под снимок тома, и мы можем считать содержимое тома

home_snapshot неизменным, и скопировать его на ленту. В процессе чтения, если читаемый блок не изменялся с момента создания снимка, то он читается из исходного тома (home), если же блок менялся с момента создания снимка, то используется его копия, хранимая в зарезервированном при создании снимка пространстве. После окончания копирования мы удаляем снимок командой `lvremove`.

Software RAID

Ядро Linux содержит средства для организации software raid (программных RAID-устройств). Эта возможность поддерживается драйвером устройств *md*. В отличие от *device mapper*, драйвер *md* умеет работать в “самостоятельном” режиме, получая конфигурацию из параметров, которые пользователь указал ядру при загрузке системы, что позволяет организовывать загрузку системы с RAID-устройств. Все устройства *md* имеют мажор 254 и миноры от 0 и до 16383.

В отличие от LVM, основной задачей которого является динамическое распределение дискового пространства (деление разделов на фрагменты и построение из фрагментов новых блочных устройств), задачей подсистемы RAID является построение новых блочных устройств путем объединения существующих.

Каждое из устройств, входящих в создаваемый дисковый массив, может определенным образом помечаться. Впоследствии эти метки (их также называют *array superblocks*) используются для повторной сборки массива. В частности, например, суперблок массива содержит его уникальный идентификатор, который можно использовать при сборке ранее созданного массива после перезагрузки. Если программный RAID-массив был помечен в процессе создания (т.е. на нем был создан суперблок массива), это дает возможность автоматической сборки массива вне зависимости от того, поменялся или нет порядок следования устройств. Например, такая необходимость может возникнуть в ситуации, когда порядок нумерации блочных устройств изменился – например, один из SCSI-дисков, участвовавших в построении массива, был удален.

Естественно, суперблок не является обязательным – то есть можно создавать массивы без суперблока, но управления ими может быть затруднено вследствие необходимости «руками» контролировать корректность указания устройств при переконфигурации массива.

Из интересных особенностей драйвера *md* стоит отметить то, что он поддерживает разбиение *md*-устройств на разделы, при этом минорные номера присваиваются разделам аналогично тому, как они присваиваются разделам на дисках IDE, т.е. на каждом RAID-устройстве можно создать до 63 разделов. Определить минор раздела, созданного на RAID-устройстве, можно с помощью вычисления значения следующего выражения: $64 * N + M$, где *N* – это номер массива (номер RAID-устройства) из диапазона 0 ... 255, а *M* – это номер раздела из диапазона 1 ... 63.

Следует сказать, что по умолчанию в большинстве дистрибутивов специальные файлы для разделов на *md*-устройствах не создаются, и их необходимо создать вручную командой `mkknod`. В настоящий момент оптимальным, наверное, следует считать комбинирование использования LVM и *md*, что позволяет достигнуть надежности за счет дублирования данных средствами *md*, и гибкости распределения дискового пространства за счет возможностей LVM.

Драйвер *md* хорошо подходит для создания RAID-устройств уровней 0, 1 или 0+1,

но не будет являться оптимальным вариантом в случае использования, например RAID уровня 5 (чередование данных по устройствам с вычислением контрольной суммы и кодом исправления ошибок), поскольку это создаст значительную нагрузку на процессор при большом объеме передаваемых данных. Возможно, что в таких случаях стоит подумать о приобретении аппаратного контроллера RAID - например, HP NetRaid (сделан на основе AMI MegaRAID) или Compaq Smart Array (сейчас называется HP Smart Array).

На листинге демонстрируется пример создания, активизации и остановки программных RAID-устройств уровня 0 и уровня 1 средствами драйвера md и системной утилиты mdadm. Утилита mdadm имеет конфигурационный файл /etc/mdadm.conf, но для того, чтобы проделать некоторые тесты и демонстрационные примеры нет необходимости его изменять.

В первом примере будем считать, что на жестком диске hdb создано два раздела, с которыми мы и будем экспериментировать. Для начала необходимо произвести инициализацию md-устройства. Соответственно, для успешной необходимо указать тип RAID-массива, специальный файл md-устройства, которое мы хотим инициализировать, и список блочных устройств, на которых будет располагаться получившийся массив:

```
[root@inferno dalth]#
[root@inferno dalth]# mdadm --create \
>                               /dev/md0 --level=0 \
>                               --raid-devices=2 /dev/hdb1 /dev/hdb2
mdadm: array /dev/md0 started.
[root@inferno dalth]#
[root@inferno dalth]# mdadm -Q /dev/md0
/dev/md/d0: 983.25MiB raid0 2 devices, 0 spares.
[root@inferno dalth]#
[root@inferno dalth]# mdadm -S /dev/md0
[root@inferno dalth]#
[root@inferno dalth]# mdadm --create \
>                               /dev/md0 --level=1 \
>                               --raid-devices=2 /dev/hdb1 /dev/hdb2
mdadm: array /dev/md0 started.
[root@inferno dalth]#
[root@inferno dalth]# mdadm -Q /dev/md0
/dev/md0: 491.63MiB raid1 2 devices, 0 spares.
[root@inferno dalth]#
[root@inferno dalth]# mdadm -S /dev/md0
[root@inferno dalth]#
[root@inferno dalth]# mdadm --assemble /dev/md0 /dev/hdb1 /dev/hdb2
mdadm: /dev/md0 has been started with 2 drives.
[root@inferno dalth]#
```

Последняя команда демонстрирует активизацию массива путем указания имени md-устройства и нескольких блочных устройств, на которых оно базируется. Все остальные параметры (размеры блоков, разновидность RAID и т.д.) утилита mdadm извлекла из суперблока массива. Как уже отмечалось, суперблок массива содержит еще и уникальный идентификатор массива, что дает возможность проидентифицировать каждое исходное блочное устройство на предмет его принадлежности к какому-либо массиву. Ниже приведен пример вывода утилиты **mdadm**, демонстрирующий как можно получить некоторые полезные данные о массиве, а строка, содержащая UID массива выделена жирным текстом:

```
[root@vikings root]# mdadm -Q -D /dev/md0
/dev/md0:
    Version : 00.90.01
  Creation Time : Fri Oct  8 14:29:21 2004
    Raid Level : raid1
    Array Size : 102336 (99.94 MiB 104.79 MB)
    Device Size : 102336 (99.94 MiB 104.79 MB)
    Raid Devices : 2
    Total Devices : 2
Preferred Minor : 0
    Persistence : Superblock is persistent

    Update Time : Fri Oct  8 14:32:22 2004
      State : clean, no-errors
    Active Devices : 2
    Working Devices : 2
    Failed Devices : 0
    Spare Devices : 0

    Number   Major   Minor   RaidDevice State
       0         253       24         0     active sync   /dev/hdb1
       1         253       25         1     active sync   /dev/hdb2
    UUID : 8696ffc0:52547452:ba369881:d1b252d0
    Events : 0.3
[root@vikings root]#
```

Впоследствии этот идентификатор может быть использован в файле конфигурации для утилиты mdadm. В конфигурационном файле `/etc/mdadm.conf` можно указать список устройств и правила их построения, после чего описанные в нем md-устройства будут автоматически собираться и разбираться без указания списка исходных устройств:

```
[root@vikings root]# cat /etc/mdadm.conf
MAILADDR root
ARRAY /dev/md0 UUID=8696ffc0:52547452:ba369881:d1b252d0
DEVICE /dev/hdb*
[root@vikings root]#
```

В листинге видно, что устройство массив md0 имеет указанный идентификатор, а также указано, что для построения массивов могут быть использованы все разделы диска hdb. В этом примере если системный администратор напишет команду **mdadm --assemble /dev/md0**, то mdadm просканирует все файлы устройств с именами, совпадающими с шаблоном `/dev/hdb*` и подключит к массиву md0 те из них, на которых будет найдена суперблок массива с тем идентификатором, который указан в параметре **ARRAY** для устройства `/dev/md0`. Системный администратор, который хочет расположить корневую файловую систему на md-устройстве, должен указать ядру при загрузке какие именно физические блочные устройства должны входить в md-устройство, на котором содержится корневая файловая система. Обычно это делается путем загрузки ядра командной строкой с опциями следующего вида:

```
linux md0=0,/dev/hdb1,/dev/hdb2 root=/dev/md0
```

Данный пример приведен скорее как иллюстративный, поскольку в зависимости от опций, использованных при создании RAID-устройства, на котором расположена корневая файловая система, командная строка ядра может меняться. В современных дистрибутивах при необходимости инициализации md-устройств для загрузки системы, как правило код и утилиты инициализации устройства

помещаются в `initrd`.

Суперблок массива записывается не в начале блочного устройства, а ближе к его середине или концу. Сделано это было для того, чтобы можно было создать RAID-массив с `boot`-сектором, который сможет быть прочитан не только ядром Linux с драйвером `md`, но и базовым загрузчиком BIOS, вследствие чего можно объединить в RAID-массив не *разделы* жестких дисков, а непосредственно *физические диски*. Тогда загрузчик, установленный в начало RAID-устройства, окажется установленным в начало жесткого диска, после чего можно использовать при загрузке ядра следующую командную строку:

```
linux md0=d0,/dev/hda,/dev/hdb root=/dev/md_d0p1
```

Драйвер `md` также поддерживает возможность задания hotswap-устройств для массивов, т.е. резервных устройств, которые могут быть активизированы в прибое одного из основных устройств в массиве.

Поддержка устройств software RAID в Linux дает возможность создавать серверы с высокой отказоустойчивостью и быстродействием.

Device mapper

В ядрах линейки 2.6 появилась еще одна подсистема по некоторым функциям аналогичная подсистеме MD, и называемая **device-mapper**. Это модульная компонентная подсистема, позволяющая с помощью специальных команд создать одно блочное устройство из нескольких кусков других блочных устройств, а также определить правила, по которым производится запись на эти «нижележащие» блочные устройства.

LVM работает именно через подсистему device mapper, и на самом деле все утилиты LVM на самом деле просто передают инструкции о том из каких фрагментов каких блочных устройств состоит какой том LVM в драйвер device mapper, в каком порядке осуществляется запись и чтение данных, и впоследствии при записи на том LVM или чтении с него, работа на самом деле ведется с устройствами, обслуживаемыми драйвером device mapper, который и делает всю работу.

Данная многоуровневая архитектура позволяет значительно упростить и таким образом значительно повысить стабильность работы системы, поскольку реализация нескольких небольших узкофункциональных компонентов в общей сложности содержит меньше ошибок, чем реализация всех этих функций в одной подсистеме.

Host-RAID, или дешевых RAID-контроллеров не бывает

Сегодня даже для дешевых современных материнских плат фирмы-производители часто декларируют «аппаратную поддержку RAID» и у многих пользователей этот факт вызывает недоумение – как же так, мой Linux не умеет работать с RAID?! ***На самом деле все проще – задекларированная и разрекламированная поддержка RAID-массивов на материнских платах для офисных и домашних компьютеров – это миф.***

Вся поддержка RAID в таких «контроллерах» на самом деле представляют собой просто небольшое расширение в BIOS и без специальных драйверов в 32/64-битных операционных системах не работают, а все функции RAID для них выполняются драйвером. Если в Windows для каждого из таких контроллеров фирма-производитель пишет драйвер, то в Linux ситуация немного иная.

Для реализации возможности работы с такими псевдо-RAID контроллерами (иногда называемыми fake-RAID) была разработана утилита **dmraid**. При запуске она сканирует жесткие диски в поисках специальных блоков (функционально аналогичных суперблокам уже знакомых нам md-устройств), записываемых такими fake-RAID контроллерами, и если ей удалось распознать формат этого специального блока, то dmraid инструктирует подсистему device-mapper о том, в каком порядке следует считывать блоки с жестких дисков.

После этого device-mapper создает блочное устройство, при записи или чтении данных с которого данные автоматически читаются и пишутся так, как сделал бы это драйвер от производителя материнской платы или контроллера.

Сетевая подсистема

Ключевым (с нашей точки зрения) объектом сетевой подсистемы Linux является интерфейс. Сетевой интерфейс в Linux – это абстрактный именованный объект, используемый для передачи данных через некоторую линию связи без привязки к ее (линии связи) реализации. Конечно, сказано мудрено – но попробуем объяснить «на пальцах».

Например, если в системе существует интерфейс *eth0*, то в большинстве случаев на современных компьютерах он сопоставлен Ethernet-адаптеру, встроенному в материнскую плату. Интерфейс с именем *ppp0* отвечает за некоторое соединение «точка-точка» с другим компьютером. Интерфейс с именем *lo* является виртуальным и представляет как бы замкнутый сам на себя (вход непосредственно подключен к выходу) сетевой адаптер.

Основная задача интерфейса – абстрагироваться от физической составляющей канала. То есть программы и система будут использовать один и тот же метод «отправить пакет» для отправки данных через любой интерфейс – хоть *lo*, хоть *ethX*, хоть *pppY*, и точно так же использовать один и тот же метод «принять пакет» – то есть создается унифицированный API передачи данных, независимый от носителя.

Для того, чтобы ознакомиться с интерфейсами, можно воспользоваться командой *ifconfig*:

```
$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:11:2F:A8:DE:A4
          inet  addr:172.23.2.114      Bcast:172.23.2.255
Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:11 Base address:0x4000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:60 errors:0 dropped:0 overruns:0 frame:0
          TX packets:60 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4707 (4.5 KiB)  TX bytes:4707 (4.5 KiB)
```

В данном случае мы видим в системе два активных интерфейса, *eth0* и *lo*, а также

некоторую информацию об их состоянии, настройках и параметрах и состоянии аппаратуры для интерфейса *eth0*. В частности, поле *Link encap* характеризует тип интерфейса, *HWAddr* – аппаратный адрес устройства (например, MAC-адрес для Ethernet), *MTU* – максимальный размер передаваемого пакета. Также могут представлять интерес поля статистики – они сообщают, какой объем данных был передан и получен через соответствующий интерфейс.

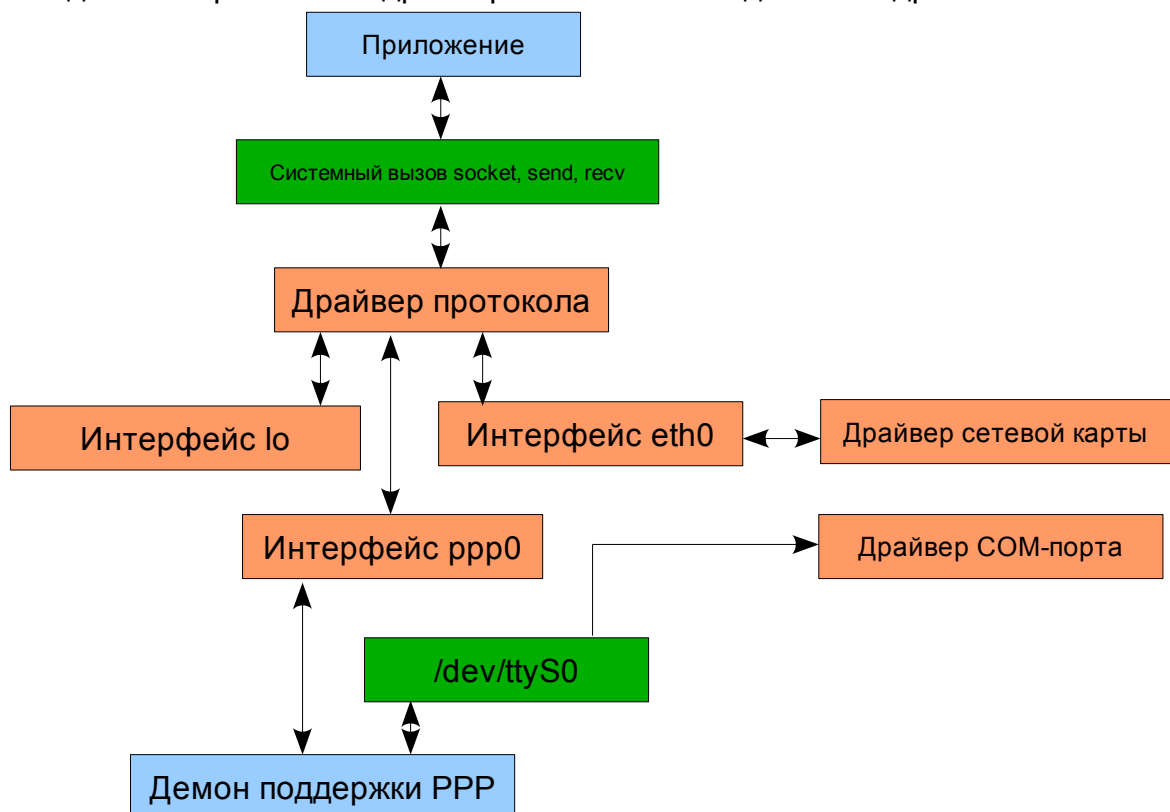
Наиболее часто встречающиеся типы интерфейсов:

- *eth* (Ethernet) – обычно соответствует отдельному сетевому адаптеру
- *ppp* (Point-To-Point) – соединение точка-точка, например при коммутируемом доступе, но очень часто используется и при организации VPN
- *slip* (Serial Line IP) – соединение точка-точка, устаревший протокол
- *wl* (Wireless) – беспроводной интерфейс, обычно сопоставлен соответствующему адаптеру
- *lo* (Loopback) – интерфейс-«петля», то есть что послал, то и получил – используется для общения между сетевыми приложениями в рамках одного компьютера

Команда *ifconfig* может также использоваться для остановки или активации интерфейса, а также изменения его параметров, связанных с протоколом IP:

```
# ifconfig eth0 down
# ifconfig eth0 up
# ifconfig eth0 inet 192.168.2.210 netmask 255.255.255.0
# ifconfig eth0 mtu 296
```

Для управления параметрами других протоколов используются другие команды – например, *ipx_config* для управления параметрами, связанными с протоколом IPX. Рассмотрим также картинку, на которой приведена приблизительная схема взаимодействия различных драйверов и сетевых подсистем ядра:



Предположим, что приложение пытается отправить пакет. Перед отправкой через системные вызовы группы `socket` (`bind`, `connect` и пр.) приложение настраивает специальный файловый дескриптор. После окончания настройки каждый записанный в этот дескриптор пакет должен быть отправлен по сети получателю. Как движется пакет в нашей системе? Прежде всего, пакет попадает в драйвер протокола. Этот драйвер определяет через какой интерфейс должна производиться отправка, дописывает к пакету необходимые заголовки и отдает пакет на обработку соответствующему интерфейсу (точнее, ставит пакет в очередь, связанную с этим интерфейсом).

Что драйвер сделает с пакетом, это уже его дело. Например, драйвер интерфейса `loopback` этот пакет вынет из очереди и сразу поставит в очередь «принятых», откуда его впоследствии заберет драйвер протокола (левая цепочка на схеме). Драйвер интерфейса `eth0` допишет к пакету заголовки Ethernet и передаст пакет драйверу сетевого адаптера, и уже тот непосредственно проинструктирует сетевой адаптер, откуда взять и как отправить пакет (правая цепочка). В средней же цепочке мы видим схему работы PPP, когда пакет помещается в очередь интерфейса `ppp0`, откуда его заберет демон `pppd`. Демон допишет в пакет нужные заголовки, и через символичный специальный файл `/dev/ttyS0` передаст пакет драйверу COM-порта, а тот непосредственно будет работать с аппаратурой. Соответственно, при приемке данных цепочки проходятся в обратном порядке.

Маршрутизация IP и форвардинг

Маршрутизация транзитных IP-пакетов (не предназначенных для этого компьютера), или IP-форвардинг, является опциональной возможностью IP-стека Linux. По умолчанию функция форвардинга не активируется, и система не пересылает транзитные пакеты через свои интерфейсы, а только обрабатывает адресованные ей пакеты. Включение форвардинга IP-пакетов производится через параметр `net.ipv4.ip_forward` интерфейса `sysctl`. Если значение этого параметра равно 0, то форвардинг отключен, если же значение параметра не равно 0, форвардинг включен:

```
[dalth@viking dalth]$ sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
[dalth@viking dalth]$
```

Кроме того, возможно разрешать или запрещать участие в форвардинге для каждого интерфейса индивидуально:

```
[dalth@viking dalth]$ sysctl -a | grep forward | grep v4
net.ipv4.conf.vmnet1.mc_forwarding = 0
net.ipv4.conf.vmnet1.forwarding = 0
net.ipv4.conf.eth0.mc_forwarding = 0
net.ipv4.conf.eth0.forwarding = 0
net.ipv4.conf.lo.mc_forwarding = 0
net.ipv4.conf.lo.forwarding = 0
net.ipv4.conf.default.mc_forwarding = 0
net.ipv4.conf.default.forwarding = 0
net.ipv4.conf.all.mc_forwarding = 0
net.ipv4.conf.all.forwarding = 0
net.ipv4.ip_forward = 0
[dalth@viking dalth]$
```

По умолчанию форвардинг включается и выключается для всех интерфейсов одновременно, но для отдельных интерфейсов возможно сменить флаг участия в форвардинге. Изменять параметры форвардинга может только системный

администратор или пользователь, который имеет право записи в необходимые файлы интерфейса `sysctl`. Следующий листинг демонстрирует включение форвардинга через все интерфейсы путем вызова программы `sysctl`:

```
[root@viking dalth]# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[root@viking dalth]# sysctl -a | grep forward | sort
net.ipv4.conf.all.forwarding = 1
net.ipv4.conf.all.mc_forwarding = 0
net.ipv4.conf.default.forwarding = 1
net.ipv4.conf.default.mc_forwarding = 0
net.ipv4.conf.eth0.forwarding = 1
net.ipv4.conf.eth0.mc_forwarding = 0
net.ipv4.conf.lo.forwarding = 1
net.ipv4.conf.lo.mc_forwarding = 0
net.ipv4.conf.vmnet1.forwarding = 1
net.ipv4.conf.vmnet1.mc_forwarding = 0
net.ipv4.ip_forward = 1
[root@viking dalth]#
```

В процессе маршрутизации для выбора интерфейса и следующего узла для доставки пакета (next hop) ядро использует *таблицу маршрутизации*. Эта таблица представляет список критериев, в соответствии с которыми выбирается следующий узел. В частности, в таблице маршрутизации фигурируют следующие условия: адрес сети получателя пакета, маска подсети получателя пакета, IP-адрес следующего узла, метрика маршрута и служебные поля (например, тип и возраст записи). Таблица маршрутизации используется не только в IP-форвардинге, но и даже при простой отсылке IP-пакета для выбора интерфейса, через который будет производиться отсылка пакета.

Запись о сети с адресом 0.0.0.0 и маской подсети 0.0.0.0 называют маршрутом по умолчанию, или *default route*. Узел, чей адрес указан в поле gateway для маршрута по умолчанию, называют маршрутизатором по умолчанию, или *default gateway* или *default router*. В системе может быть произвольное количество маршрутов по умолчанию, но они должны быть как минимум с разными метриками. Для просмотра таблицы маршрутизации можно воспользоваться командой `route`. Эта команда позволяет оперировать с таблицей маршрутов, добавляя и удаляя из нее записи.

```
[root@inferno dalth]# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use
Iface
10.80.1.113 0.0.0.0 255.255.255.255 UH 0 0 0
ppp0
127.0.0.0 0.0.0.0 255.0.0.0 U 0 0 0 lo
0.0.0.0 10.80.1.113 0.0.0.0 UG 0 0 0
ppp0
[root@inferno dalth]# route del default
[root@inferno dalth]# route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use
Iface
10.80.1.113 0.0.0.0 255.255.255.255 UH 0 0 0
ppp0
127.0.0.0 0.0.0.0 255.0.0.0 U 0 0 0 lo
[root@inferno dalth]# route add default dev ppp0
[root@inferno dalth]#
```

В данном выводе таблица упорядочена по маске подсети, что соответствует порядку ее просмотра ядром. Столбцы *Destination* и *Genmask* содержат адрес и маску сети получателя пакета, столбец *Metric* фактически указывает приоритет маршрута (маршрут с меньшей метрикой более приоритетен), поле *Gateway* указывает IP-адрес следующего узла для передачи пакета. Некоторые типы интерфейсов (в частности, интерфейсы типа точка-точка, или point-to-point) подразумевают, что на принимающем конце линии связи всегда находится не более одного узла, и поэтому в этой ситуации IP-адрес следующего узла можно не указывать. В данном случае мы видим, что в приведенном примере некоторые узлы доступны через интерфейс `ppp0` типа точка-точка. В частности, именно из-за этого свойства приведенная выше таблица оказывается эквивалентна следующей ниже. Жирным шрифтом помечена измененная строка, демонстрирующая “точечную” природу PPP-соединения:

```
[root@inferno dalth]# route -n
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use
Iface						
10.80.1.113	0.0.0.0	255.255.255.255	UH	0	0	0
ppp0						
127.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0
lo						
0.0.0.0	0.0.0.0	0.0.0.0	U	0	0	0
ppp0						

Специфика использования протокола PPP (обычно используемого при модемных соединениях) такова, что любой PPP-интерфейс является интерфейсом типа точка-точка, более того – PPP и расшифровывается как *Point-to-Point Protocol*. Также интерфейсами точка-точка являются интерфейсы *SLIP* (Serial Line IP) и практически все разновидности туннельных интерфейсов.

При деактивизации интерфейса из таблицы маршрутизации автоматически исключаются все маршруты, для которых в поле *Iface* был указан отключившийся интерфейс. Для некоторых типов интерфейсов при активизации в таблице маршрутизации также создаются служебные записи о маршрутах, которые нельзя удалить.

В большинстве случаев таблица маршрутизации имеет не слишком большой размер, но в некоторых ситуациях (в частности, на шлюзовых машинах в больших сетях) таблица может иметь весьма значительный размер и изменяется не “вручную” с помощью команды `route`, а специальными программами – демонами поддержки протоколов динамической маршрутизации. С некоторыми упрощениями алгоритм работы этих демонов можно описать следующим образом: демон “слушает” приходящие пакеты для обслуживаемых протоколов динамической маршрутизации, и по получении (или неполучении) такого пакета отдает ядру команду на изменение таблицы маршрутизации.

Следует также заметить, что команда `route` в режиме вывода таблицы маршрутизации фактически просто фильтрует и форматирует данные, содержащиеся в специальном файле, называемом `/proc/net/route`, используемом для доступа к таблице маршрутизации, ведущейся ядром.

Фильтры пакетов

Возможность инсталляции фильтров пакетов является очень интересной возможностью, предоставляемой стеком TCP/IP ядра Linux. Не углубляясь в

детали просто отметим, что IP-стек Linux позволяет различным модулям установить “ловушки” (hooks) для пакетов. При этом каждый пакет, попадающий в такую ловушку, передается для обработки драйверу, установившему эту ловушку. Драйвер, в свою очередь, может проанализировать пакет, проделать какие-либо действия с пакетом, после чего вернуть код обработки, инструктируя таким образом ядро о том, что следует делать с пакетом дальше – вернуть ли отправителю сообщение об ошибке, прервать ли обработку и уничтожить пакет, либо продолжать обработку пакета обычным образом.

В настоящее время для фильтрации пакетов наиболее часто используются средства *iptables*. *iptables* – это название утилиты, которая позволяет настроить множество драйверов сетевой подсистемы NETFILTER ядра Linux, позволяющих осуществить анализ, произвести преобразование, или изменить обработку IP-пакетов. Основным объектом в *iptables* является цепочка правил (*chain*). Каждое правило в цепочке содержит набор условий совпадения (*condition matches*) и действие (*action*). Цепочки сгруппированы в таблицы (*tables*).

Действий есть две разновидности – прерывающие обработку пакета в цепочке, например действия DROP или ACCEPT, и не прерывающие обработку пакета цепочкой – например, LOG или MARK.

Цепочки используются для проверки пакетов – то есть пакет поочередно последовательно сравнивается с каждым из правил цепочки, и если он удовлетворяет всем условиям в правиле, к пакету применяется действие, указанное в этом правиле. Если действие является прерывающим, то на этом обработка пакета этой цепочкой заканчивается, если действие не прерывающее, то пакет продолжает проверяться этой же цепочкой.

Стандартные цепочки также содержат специальное неявное действие по умолчанию, называемое политикой цепочки (*chain policy*). Действие, указанное как политика цепочки, применяется ко всем пакетам, которые не попали ни под одно правило с “прерывающим” действием.

Стандартные таблицы и цепочки

Подсистема пакетного фильтра содержит три таблицы, в каждой из которых содержатся несколько цепочек (наборов правил). Кроме того, администратор может создавать собственные цепочки правил. Ниже перечисляются стандартные таблицы и цепочки:

Таблица	Назначение	Цепочки	Назначение
mangle	Модификация пакетов	PREROUTING	Модификация всех пришедших пакетов
		INPUT	Модификация пакетов, пришедших на адрес компьютера
		FORWARD	Модификация пакетов, которые должны быть отмаршрутизированы (пересланы на другой хост)
		OUTPUT	Модификация сгенерированных пакетов, процессами

Таблица	Назначение	Цепочки	Назначение
			данного хоста
		POSTROUTING	Модификация всех переданных пакетов
filter	Фильтрация пакетов – принятие решения об их дальнейшей обработке или отказе от обработки)	INPUT	Фильтрация адресованных этому компьютеру пакетов
		OUTPUT	Фильтрация сгенерированных этим компьютером пакетов
		FORWARD	Фильтрация маршрутизируемых (транзитных) пакетов
nat	Трансляция адресов	PREROUTING	Трансляция адресов всех принимаемых пакетов
		FORWARD	Трансляция адресов транзитных пакетов
		POSTROUTING	Трансляция адресов всех передаваемых пакетов

Таблица *nat* обладает “двойственным” действием, т.е. если вы включите преобразование для входящих пакетов, исходящие пакеты также будут модифицироваться, и наоборот. Таблицы *mangle* и *nat* изменяют пакеты, но *mangle* не ведет список изменений, т.е. является “однонаправленной” таблицей.

Порядок применения стандартных таблиц и цепочек

Рассмотрим, какой путь проходит пакет в цепочках и таблицах *iptables*. Для входящих пакетов (адресованных компьютеру, на котором активизирована поддержка *iptables*) верна следующая последовательность применения цепочек:

1. *mangle.PREROUTING*
2. *nat.PREROUTING*
3. *mangle.INPUT*
4. *filter.INPUT*

Для пакетов, отправляемых с компьютера, реализуется следующая цепочка обработки:

1. *filter.OUTPUT*
2. *mangle.OUTPUT*
3. *nat.POSTROUTING*
4. *mangle.POSTROUTING*

Пакеты, являющиеся транзитными (маршрутизируемыми), т.е. не адресованные фильтрующему компьютеру и не сгенерированные фильтрующим компьютером, проходят по следующей последовательности цепочек и таблиц:

```

1. mangle.PREROUTING
2. nat.PREROUTING
3. mangle.FORWARD
4. filter.FORWARD
5. nat.FORWARD
6. nat.POSTROUTING
7. mangle.POSTROUTING

```

Стандартные действия

Каждое правило в любой цепочке может ссылаться на одно из стандартных или дополнительных действий, либо на какую-либо пользовательскую цепочку. Основное различие между стандартными и дополнительными действиями в том, что стандартные действия могут указываться в правилах любых цепочек любых таблиц, а дополнительные действия можно указывать только для некоторых цепочек некоторых таблиц. Перечислим стандартные действия:

ACCEPT	Прервать проверку пакета цепочкой и перейти к следующей в порядке обработки пакета стандартной цепочке
DROP	Прервать обработку пакета, сам пакет удалить
RETURN	Прервать проверку пакета цепочкой и вернуться к проверке вышестоящей цепочкой, а если действие встретилось в одно из стандартных цепочек, поступить с пакетом так, как предписано в политике цепочки (chain policy)
QUEUE	Передать пакет некоторому процессу для дальнейшей обработки

Интересной особенностью также является то, что существуют так называемые *target extensions*, которые реализованы как модули и также могут использоваться для указания проводимого над пакетом действия. В частности, к таким действиям, например, относятся действия LOG – запротоколировать факт получения пакета, MASQUERADE – подменить IP-адрес отправителя пакета, MARK – пометить пакет и многие другие. Некоторые действия могут встречаться не во всех цепочках, а только в некоторых цепочках некоторых таблиц.

Еще одним специфическим действием можно назвать переход к пользовательской цепочке. При этом, если пакет в процессе обработки попадает под действие правила, у которого в качестве действия указан переход к другой цепочке, то пакет начинает проверяться ее правилами. Это часто используется, чтобы сходным образом обрабатывать некоторые виды пакетов.

Условия отбора

Условия отбора делятся на две группы – стандартные условия, которые применимы ко всем пакетам, и расширенные условия, называемые также *match extensions*. Расширенные условия могут применяться не для всех пакетов, а только для некоторых из них – например, дополнительные условия для протокола UDP включают в себя адреса портов отправителя и получателя, а для ICMP – тип и код ICMP-сообщения.

Примеры конфигураций iptables

Попробуем рассмотреть несколько простых примеров, достаточно часто используемых в реальных конфигурациях. Стоит заметить, что самостоятельная

конфигурация пакетного фильтра требует некоторых (точнее, достаточно значительных) знаний сетевых протоколов, поскольку в конфигурации необходимо задавать множество критериев, которые сильно зависят от ситуации и от используемых сервисов.

Пример 1: простейшая конфигурация iptables для домашнего компьютера, подключенного к Internet. В этой конфигурации мы запретим все входящие соединения, а также все исходящие пакеты UDP кроме тех, которые необходимы для нормальной работы в Internet с использованием PPP-соединения. В этой конфигурации мы разрешаем передачу всех пакетов в рамках локальной машины, разрешаем исходящие TCP-пакеты, разрешаем входящие пакеты TCP для уже установленных соединений, и разрешаем передачу и прием UDP-пакетов для службы DNS и пакетов автоматической конфигурации соединения PPP (пакетов DHCP). Кроме того, следует разрешить прием управляющих пакетов ICMP и отправку запросов и прием ответов PING:

```
# iptables -P INPUT DROP
# iptables -A INPUT -j ACCEPT -i lo
# iptables -A INPUT -j ACCEPT -p tcp ! --syn
# iptables -A INPUT -j ACCEPT -p udp --source-port 53
# iptables -A INPUT -j ACCEPT -p udp --source-port 67 --destination-port 68
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type destination-unreachable
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type time-exceeded
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type parameter-problem
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type echo-reply
# iptables -P OUTPUT DROP
# iptables -A OUTPUT -j ACCEPT -p tcp
# iptables -A OUTPUT -j ACCEPT -p udp --destination-port 53
# iptables -A OUTPUT -j ACCEPT -p udp --destination-port 67 --source-port 68
# iptables -A OUTPUT -j ACCEPT -p icmp --icmp-type echo-request
```

Пример 2: то же самое, что в примере 1, но все “отбитые” пакеты протоколируются. Для того, чтобы добиться такого эффекта, нужно создать дополнительную цепочку, которая будет протолировать и удалять пакеты. Эту цепочку мы назовем KILLER – вполне обоснованно, не так ли? Кроме того, мы исправим политики стандартных цепочек так, чтобы запрещенные пакеты не удалялись, а “забрасывались” в созданную нами цепочку KILLER, а нашей основной цели (сначала протолировать, потом удалить пакет) можно добиться просто указав два действия – сначала LOG, затем DROP. Поскольку действие LOG не является прерывающим обработку, мы получим требуемый нам эффект:

```
# iptables -N KILLER
# iptables -A KILLER -j LOG
# iptables -A KILLER -j DROP
# iptables -P INPUT KILLER
# iptables -A INPUT -j ACCEPT -i lo
# iptables -A INPUT -j ACCEPT -p tcp ! --syn
# iptables -A INPUT -j ACCEPT -p udp --source-port 53
# iptables -A INPUT -j ACCEPT -p udp --source-port 67 --destination-port 68
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type destination-unreachable
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type time-exceeded
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type parameter-problem
# iptables -A INPUT -j ACCEPT -p icmp --icmp-type echo-reply
# iptables -P OUTPUT KILLER
# iptables -A OUTPUT -j ACCEPT -p tcp
# iptables -A OUTPUT -j ACCEPT -p udp --destination-port 53
# iptables -A OUTPUT -j ACCEPT -p udp --destination-port 67 --source-port 68
# iptables -A OUTPUT -j ACCEPT -p icmp --icmp-type echo-request
```

Пример 3: маскаррад пакетов. Маскарадом называют преобразование IP-адресов проходящих пакетов так, чтобы они выглядели как отправленные с системы-маршрутизатора, а не с какого-либо узла “за” маршрутизатором. Достигается это путем изменения IP-адреса (и, возможно, номера порта) в “транзитных” пакетах. Собственно преобразование задается путем указания действий SNAT – замена адреса отправителя, DNAT – замена адреса получателя, или MASQUERADE – функционально аналогично SNAT, но без указания конкретного IP-адреса (IP-адресом для замены назначается IP-адрес интерфейса через который уходит пакет, со всеми отсюда вытекающими – например, если интерфейс меняет IP-адрес или просто деактивируется все «маскированные» через него соединения сбрасываются). Предположим, что наш “внешний” интерфейс имеет адрес 193.267.14.6, а внутренняя сеть имеет адрес 192.168.0.0/24. Тогда для того, чтобы дать всем компьютерам нашей сети доступ по протоколу TCP “наружу”, мы должны подать примерно следующую команду:

```
# iptables -A POSTROUTING -t nat -j SNAT -o ppp0 \
> --to-source 193.267.14.6 -p tcp \
> --source 192.168.0.0/24 \
> --destination ! 192.168.0.0/24
```

Если у нас внешний адрес динамический, а не статический (мы работаем по dialup – соединению), то мы можем использовать динамический маскаррад без привязки к внешнему адресу – ну или с использованием динамической привязки, кому как больше нравится:

```
# iptables -A POSTROUTING -t nat -j MASQUERADE -o ppp0 \
> --source 192.168.0.0/24 \
> --destination ! 192.168.0.0/24
```

Действие SNAT более эффективно, MASQUERADE проще в использовании, но обладает рядом существенных недостатков (не вдаваясь в подробности, просто заметим, что на системе с несколькими интерфейсами и сложной таблицей маршрутизации проблемы почти наверняка будут). Особое внимание нужно обратить на указание -o ppp0, то есть действие применяется ТОЛЬКО для пакетов, отправляемых через интерфейс ppp0. Еще вы можете увидеть, что мы указываем это правило только один раз, и обратного к нему правила не строим -

об этом позаботится функция connection tracking (отслеживание состояния соединений), и обратная замена адресов в отправляемых в ответ на наши запросы пакетах будет произведена системой автоматически.

Пример 4: “проброс” пакетов во внутреннюю сеть. Обычно это используется, если мы хотим перебросить пакеты, пришедшие на адрес маршрутизатора, на какую-либо из машин внутренней сети (например, так можно предоставить доступ ко внутреннему WWW-серверу). Достигается это использованием действия DNAT (destination NAT). В нашем случае мы перебрасываем все TCP-пакеты, пришедшие на интерфейс маршрутизатора ppp0 на порт 80, на порт 85 компьютера с адресом 192.168.0.6:

```
# iptables -A PREROUTING -t nat -j DNAT -i ppp0 \  
> --to-destination 192.168.0.6:85 -p tcp --destination-ports 80
```

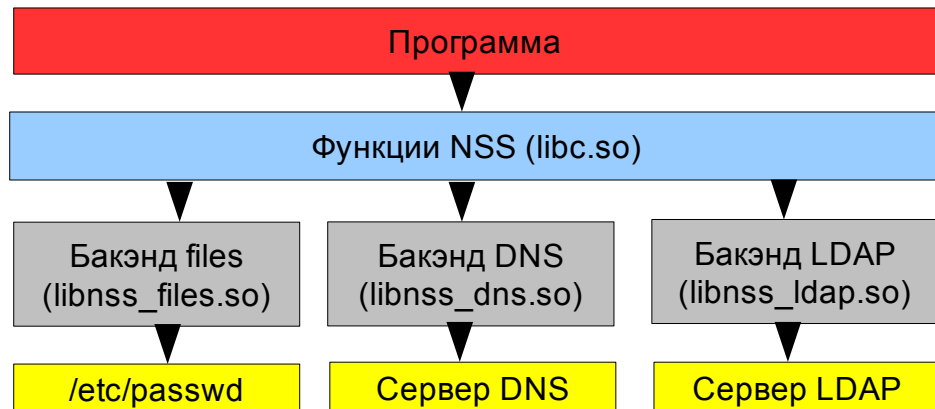
Конечно, приведенными примерами возможности iptables не исчерпываются, скорее даже наоборот – это лишь малая часть того, что умеет эта подсистема. Приведенные же примеры демонстрируют наиболее простые случаи, позволяющие составить некоторое представление об iptables и возможностях этой технологии.

В системах, основанных на RedHat Linux и его вариациях, есть специальный стартовый сценарий загрузки, также называемый iptables. Расположен он как правило в каталоге **/etc/rc.d/init.d/**. Этот сценарий при загрузке системы устанавливает правила, созданные администратором, и его можно использовать для сохранения конфигурации iptables. В процессе конфигурации системный администратор задает конфигурацию подсистемы iptables, используя утилиту **/sbin/iptables**, а после окончания настройки дает команду **/etc/rc.d/init.d/iptables save**, после чего текущая конфигурация сохраняется в файл **/etc/sysconfig/iptables**. Существует также множество “фронтендов” для настройки iptables, которые могут использоваться начинающими пользователями и не слишком опытными администраторами, но “ручной” способ настройки все-таки предпочтительней, поскольку позволяет очень точно настроить подсистему iptables.

Управление пользователями, NSS и PAM

Linux – система многопользовательская. По умолчанию, большинство дистрибутивов используют «классический» набор файлов в которых хранится информация о пользователях и группах: **/etc/passwd**, **/etc/group**, **/etc/shadow**, **/etc/gshadow**. Во многих ситуациях этого вполне достаточно, но иногда возникает необходимость в интеграции Linux в более или менее чужеродное, либо просто распределенное окружение, и именно в этот момент к нам на помощь приходят такая интересная подсистема, как NSS – Name Service Switch. Основная задача NSS – создать модульное окружение для управления пользователями. Реализовано это посредством загружаемых библиотек. Основные вызовы NSS реализованы в библиотеке **libc**, а **libc** в свою очередь загружает и вызывает бэкэнды

NSS:



При инициализации программы, так или иначе связанной с NSS, загружаются основная библиотека `libc.so`, которая считывает конфигурацию из файла `/etc/nsswitch.conf`, после чего также загружаются те библиотеки NSS, которые указаны в этом файле.

Впоследствии при работе программы, если программе требуется работать с именованными сущностями, соответствующие вызовы функций `glibc` будут обращаться к функциям NSS и использовать в те источники данных, которые указаны в `nsswitch.conf`.

В частности, через NSS можно разрешать (определять) имена и идентификаторы протоколов, номера портов служб (сервисов), имена и идентификаторы пользователей и групп, IP-адреса и имена компьютеров и некоторые другие данные.

Пример файла `nsswitch.conf`:

```
[viking@alpha etc]$ cat /etc/nsswitch.conf
passwd:      files ldap
shadow:      files ldap
group:       files ldap
hosts:       files dns
bootparams:  nisplus [NOTFOUND=return] files
ethers:      files
netmasks:   files
networks:    files
protocols:   files
rpc:         files
services:    files
netgroup:    nisplus
publickey:   nisplus
automount:   files nisplus
aliases:     files nisplus
[viking@alpha etc]$
```

В данном примере указано, что для определения имен пользователей и групп используются сначала текстовые файлы и затем LDAP, для определения имен компьютеров и IP-адресов используются сначала текстовые файлы и затем DNS, для определения алиасов и настроек автоматического монтирования каталогов используются сначала текстовые файлы и затем служба NIS+.

Библиотеки бакэндов системы NSS хранятся в файлах `libnss_XXX.so`, где XXX –

это имя бэкэнда. Например `libnss_files.so` – это бэкэнд NSS использующий в качестве источника данных текстовые файлы, `libnss_db.so` – бэкэнд использующий файлы BerkleyDB, `libnss_ldap.so` – бэкэнд позволяющий хранить данные в каталоге LDAP. Как правило, каждый бэкэнд имеет свои дополнительные конфигурационные файлы.

Как следствие, если у вас возникла необходимость использовать вашу Linux-систему в сетевом или чужеродном окружении и обеспечить ее интеграцию с ним, вы можете воспользоваться NSS и получить доступ к информации через соответствующий бэкэнд – например, для интеграции в среду Solaris, вы можете воспользоваться бэкэндом NIS/NIS+, для интеграции в ActiveDirectory – бэкэндом LDAP.

При этом модульность NSS позволяет вам объединять различные источники данных – например, использовать текстовые файлы и DNS для определения имен компьютеров, NIS для определения имен протоколов и сервисов, и текстовые файлы и LDAP для определения имен и идентификаторов пользователей и групп. Подсистема PAM (Pluggable Authentication Modules) идейно очень схожа с NSS, но отличается от нее назначением. Основная задача PAM – аутентификация пользователей (проверка паролей, прав доступ, ограничений и так далее). Как и NSS, PAM состоит из набора основных библиотек и бэкэндов, причем необходимые бэкэнды, порядок их вызова и некоторые опциональные параметры определяются в конфигурационных файлах PAM, обычно они расположены в каталоге `/etc/pam.d`. Главным отличием PAM от NSS (кроме естественно назначения) является то, что PAM является не составной и неотъемлемой частью `libc`, а отдельным множеством библиотек.

Основная часть стандартных утилит UNIX для управления пользователями и группами и получения информации о них, в большинстве дистрибутивов Linux общего назначения, адаптирована и собрана с поддержкой NSS и PAM. К таким утилитам относятся `passwd`, `chsh`, `chfn`, `id`, `who` и другие. NSS также используется даже такими утилитами как `ls`, `find`, `ps` – то есть всеми теми программами, которые отображают имя пользователя. Соответственно, если программа запрашивает у пользователя пароль – скорее всего она использует и NSS, и PAM (например XDM или GDM). Большинство программ в чьи функции входит обработка почты также используют NSS. Соответственно, можно уверенно говорить что подсистемы NSS и PAM и базовые знания об их предназначении на сегодняшний день являются необходимыми для администратора Linux-систем.

X11 и все-все-все

Большинство нынешних дистрибутивов по умолчанию устанавливают для пользователя графическую среду X11 (X11 Windows System), под управлением которой и выполняются все графические приложения. Как «внутри» устроена X11? Прежде всего, X11 – это распределенная модульная среда, состоящая из двух основных компонентов: X-сервера и X-клиента.

Клиент-серверная архитектура X11

Х-сервер – это программа, которая организует работу с устройствами ввода/вывода, производит отрисовку видимых элементов, запущена у пользователя и предоставляет свои ресурсы (те же самые устройства ввода-вывода) для X-клиентов. X-сервер загружает драйверы устройств (например

видеокарты, мыши или клавиатуры), он же управляет переключением раскладок клавиатуры и т.п. Кроме того, X-сервер частично берет на себя функции работы со шрифтами. Задача использования аппаратного ускорения для отрисовки также является прерогативой X-сервера.

В современных дистрибутивах как правило используется открытый свободно распространяемый X-сервер называющийся Xorg. Его конфигурационный файл называется `xorg.conf` и расположен в каталоге `/etc/X11`. В конфигурационном файле описываются все устройства ввода, которые будет использовать X-сервер, настройки клавиатуры, драйвер видеокарты и многое другое. Более подробную информацию можно получить из справочного руководства [`man Xorg`, `man xorg.conf`].

X-клиент – это собственно пользовательская программа – браузер, почтовый клиент, видеоплеер, клиент мгновенных сообщений, игры, графические редакторы и просмотрщики и т.д.

Когда пользователь запускает графическое приложение, оно соединяется с X-сервером по стандартному протоколу X11, получает от X-сервера события о перемещении мыши, нажатиях кнопок клавиатуры и соответственно на них реагирует. Когда необходимо провести отрисовку, X-клиент отправляет соответствующие инструкции X-серверу, и уже X-сервер производит непосредственную отрисовку используя драйвер видеокарты. Команды протокола X11 могут передаваться как через разделяемую память или локальное соединение (если X-клиент и X-сервер запущены на одном компьютере), так и по сети – при этом X-клиент и X-сервер могут быть запущены на разных компьютерах.

X-сервер никаким образом не отвечает за внешний вид окна X-клиента – он отрисовывает все в точности так, как это указал клиент, более того – сам по себе X-сервер он не отрисовывает ни обрамлений окон, ни каких либо управляющих элементов, не обрабатывает никаких горячих клавиш – то есть он занимается только отрисовкой картинки присланной X-клиентом, а также чтением данных из устройств ввода и передачей соответствующих событий X-клиенту.

Обрамление окон, иконки рабочего стола, панели и кнопки – это все отрисовывается X-клиентами. Рассмотрим пример окна некоторого приложения (в нашем случае файлового менеджера Nautilus из состава среды GNOME):

На этой картинке на самом деле показан результат работы двух X-клиентов: собственно файлового менеджера `nautilus` (именно он управляет отрисовкой меню, строки статуса, панелей, иконок и так далее). Второй X-клиент – менеджер окон (`window manager`) под названием `metacity` нарисовал рамку окна и кнопки на этой рамке. При этом горячие клавиши для закрытия окна, сворачивания, перемещения и других операций с окном обрабатывает именно `metacity`, а горячие клавиши копирования файла, перехода по каталогам, навигации по меню, подсвечивание иконок и тому подобное обрабатывает уже сам `nautilus`.

Графическая среда пользователя

Графической средой мы будем называть набор программ, для пользователя для выполнения им повседневных функций. Каждая из этих программ как правило является самостоятельным X-клиентом и может работать сама по себе, даже без своих «коллег по окружению», но будучи собранными вместе, они начинают предоставлять пользователю цельный и органичный интерфейс. Достигается это обычно следующим образом:

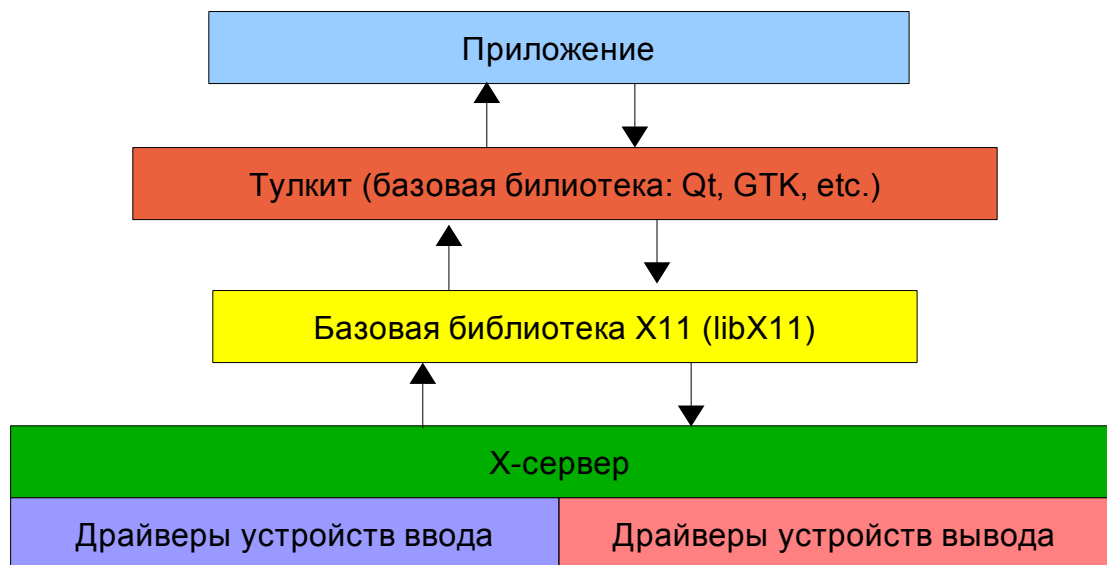
- 1) Все программы данного графического окружения используют одну и ту же библиотеку для отрисовки своих элементов управления
- 2) В графическое окружение включается какой-либо менеджер окон
- 3) Все программы для данного окружения разрабатываются с соблюдением определенных общих требований
- 4) В состав окружения включаются самостоятельные программы для выполнения базовых функций - текстовые и табличные процессоры, графический редактор, браузер, IM-клиент и другие
- 5) В состав окружения включаются служебные программы, предоставляющие пользователю возможность вызова других программ – например, приложение которое отрисовывает панель, на которой размещается кнопка для вызова меню со списком установленных программ и список окон
- 6) В состав окружения включают утилиты для настройки оборудования и дополнительных функций

В результате внешний вид окружения становится единообразным, а тот факт что оно состоит из множества небольших программ делает графическую среду гибкой в настройке и стабильной в работе. В то же время обширный набор программ для различных функций делает позволяя графическому окружению удовлетворить большинство потребностей пользователя, а привычная вам картина GNOME или KDE, которую вы видите на экране в процессе работы, является результатом совместной работы множества X-клиентов соответствующего окружения... Или нескольких окружений одновременно.

Наиболее распространенные графические среды в Linux – это GNOME и KDE. И та и другая среда имеют свою библиотеку для отрисовки элементов управления внутри окна (для GNOME это GTK, для KDE это Qt), включают в свой состав почтовый клиент, браузер, мультимедиа-проигрыватель, клиент мгновенных сообщений и графический редактор, файловый менеджер (он же по совместительству отрисовывает иконки на рабочем стол и отвечает за фон рабочего стола), программу которая отрисовывает боковые панели и набор маленьких программ-апплетов, которые встраиваются при необходимости в панели, игры, программы управления для настройки окружения и менеджер окон. Некоторые X-клиенты не входят в состав какого-либо окружения – например, браузеры Firefox и Opera, медиапроигрыватель Mplayer, офисный пакет OpenOffice.Org. Самое главное что следует запомнить – что графическое окружение, или графическая среда – это всего лишь набор программ, а не одна большая программа вида «все-в-одном», и вы можете, работая в основном в одном окружении, абсолютно спокойно использовать программы другого.

Внутри X11. Упрощенная схема

X11 – система многослойная, но внутренне логичная и понятная. Схематически ее можно изобразить следующим образом:



Когда приложение отрисовывает что-либо, оно обращается к своей базовой библиотеке, она в свою очередь переадресовывает вызов (или уже цепочку вызовов) в библиотеку libX11, которая используя X11 Core Protocol передает команды на X-сервер. X-сервер интерпретирует команды и передает их на отрисовку драйверам устройств вывода. Воздействие же пользователя на устройства ввода считываются X-сервером через драйвера устройств ввода, через X11 Core Protocol передаются X-клиенту, libX11 переводит команды протокола в события и передает события в туллит, и уже туллит передает события программе для того чтобы та на них отреагировала.

То как приложение выглядит – то есть внешний вид строк ввода, полос скроллинга, панелей и кнопок – за все это отвечает библиотека-туллит. Два наиболее распространенных на сегодняшний день туллита, Qt и GTK, примерно равны по возможностям – они обеспечивают переносимость приложений, поддержку «тем» внешнего вида, предоставляют программисту объектно-ориентированный интерфейс и некоторый набор служебных функций и возможностей.

Базовая библиотека X11 (libX11, или Xlib) позволяет программе абстрагироваться и не связываться с низкоуровневыми функциями протокола X11, и таким образом обеспечивает сетевую прозрачность X11, то есть позволяет программам работать как с локальным X-сервером, так и с находящимся на другом компьютере, причем делается это незаметно для программы.

Внутри X11. Растровые шрифты и их отображение

С момента разработки в среде X11 отображение шрифтов управлял X-сервер. Приложение, когда ему необходимо вывести некоторый текст, просто инструктировало X-сервер «отобрази вот этот текст вот таким шрифтом в указанном месте». В ответ на это X-сервер выбирал из своей базы шрифтов наиболее подходящий, и использовал его для выполнения инструкций клиента, причем изначально эти шрифты были растровыми, то есть фактически содержали наборы заранее отрисованных глифов (символов).

Эта технология, называемая X Core Fonts, поддерживается в X11 и сейчас, поэтому в наборе пакетов любого дистрибутива всегда можно встретить наборы широко распространенных семейств растровых шрифтов – fixed, hevetica, times,

courier. При этом каждый шрифт представляется множеством файлов, для различных сочетаний размера, ширины и начертания – например, если шрифт имеет десять вариантов размеров (от 8 до 18), две ширины (обычный и жирный) и два начертания (стандартное и курсив), то он будет представляться 40 файлами – по одному файлу для каждого из сочетаний размера, начертания и ширины.

Поскольку количество файлов получается очень большим, чтобы не устанавливать все эти шрифты на все компьютеры где запущены X-серверы, в систему X11 был введен такой объект как сервер шрифтов (font server). Системный администратор может настроить и запустить один сервер шрифтов для всей локальной сети, и указать X-серверу при запуске использовать шрифты с соответствующего сервера, что позволяет поддерживать на всех серверах один и тот же набор шрифтов с минимальными усилиями и избежать ситуации когда шрифт есть на одном компьютере, но его нет на другом.

В большинстве дистрибутивов Linux сервер шрифтов включен в поставку X11 по умолчанию, и называется xfs. Его конфигурационный файл как правило находится в каталоге /etc/X11/fs.

В конфигурационном файле сервера шрифтов перечисляются каталоги с растровыми шрифтами, а в конфигурационном файле X-сервера указано, что основным источником шрифтов является сервер шрифтов, запущенный на этом же компьютере.

Такая методика позволяла добиться достаточно качественного отображения шрифта в большинстве случаев, но в то же время такая она имеет определенные недостатки – как известно, растровые шрифты очень плохо «вращаются» на угол не кратный 90 градусам, и плохо масштабируются, а с появлением устройств высокой четкости (LCD-мониторов) возникла необходимость еще и отрисовывать шрифты в «сглаженном» виде, с мягкими переходами цвета, чего растровые шрифты также предоставить не могли.

Наиболее распространенное приложение, использующее методику X Core Fonts – это графический эмулятор терминала xterm, который есть в составе всех дистрибутивов, или простейший менеджер графического входа в систему XDM.

Внутри X11. FreeType и XFT

Для устранения этого недостатка растровых шрифтов была разработана специальные библиотеки XFT и FreeType, которые обеспечивают отображение векторных шрифтов и реализацию таких возможностей как сглаживание шрифтов, их поворот на произвольный угол и хинтинг (подгонку символов шрифта друг к другу оптимальным для данного шрифта способом). Поскольку данная методика плохо согласовывалась с уже сложившейся шрифтовой архитектурой X11, отображение векторных шрифтов было возложено на X-клиента.

При этом схема отображения шрифта меняется: X-клиент не передает на X-сервер запрос на вывод текста указанным шрифтом, а самостоятельно отрисовывает необходимые глифы шрифта используя функции FreeType и XFT, и передает на X-сервер уже сформированную картинку, которую X-сервер и отображает. Вполне логично, что при использовании этой методики X-клиент должен иметь доступ к оригиналу векторного шрифта (обычно это файл) – поэтому современные дистрибутивы также содержат еще и наборы векторных шрифтов PostScript и TrueType.

Новые версии тулkitов GTK и QT используют именно эту методику, но поскольку существуют еще и устаревшие тулkitы, такие как Motif или Xview (OpenLook) и значительное количество приложений на них основанных, да и с точки зрения необходимой полосы пропускания сети технология поддержки растровых шрифтов имеет явное преимущество перед XFT и FreeType, поддержка технологии

растровых шрифтов по-прежнему входит в X11 и будет оставаться в ней еще долго.

Внутри X11. Расширения

Протокол X11 не является замершим в своем развитии, но его развития ведется не путем изменения самого протокола, а путем внесения в протокол расширений – то есть дополнительных опциональных наборов команд и инструкций. Например, для поддержки непрямоугольных окон было введено расширение XShape, для поддержки проигрывания видеороликов введено расширение XVideo, для поддержки OpenGL было введено расширение GLX и так далее.

Внутри X11. Полезные утилиты

В штатную поставку X11 входит множество полезных программ, позволяющих получать служебную информацию об ваших настройках среды X11. Перечислим некоторые из них:

xterm – эмулятор терминала

xfontsel – интерактивный просмотр растровых шрифтов

xdpyinfo – вывод информации о ваших настройках среды X11 и задействованных расширениях

xwininfo – просмотр информации об указанном окне (положение, размер, класс окна и т.п.)

xwd – моментальный «снимок» окна

xwud – показ результатов работы xwd

xhost – управление контролем доступа X-клиентов к X-серверу

Внутри X11. Менеджер окон

Существует особая разновидность X-клиентов, называемых менеджерами окон. Их основная функция – обеспечивать управление другими окнами – перемещением, изменением размеров, сворачиванием и разворачиванием окон, отрисовкой обрамлений окон, управление передачей фокуса от окна к окну а также управление Z-порядком размещения окон. Соответственно, в один момент времени для одного экрана может быть использован только один менеджер окон. GNOME и KDE имеют свои собственные менеджеры окон – это **metacity** и **kwin**. Менеджеры окон являются взаимозаменяемыми – то есть вы можете выбрать и использовать тот из них, который вам больше нравится или кажется более удобным. Еще один распространенный (то есть наличествующий почти на всех UNIX-системах) менеджер окон – это **twm**. Крайне простой и примитивный, он тем не менее предоставляет пользователю базовые функции управления окнами и доступ к меню приложений.

Локализация

Широкое распространение UNIX-систем привело к необходимости введения системы локализации – поддержки предпочитаемых пользователем стандартов на запись даты и времени, формата чисел, языка сообщения программ, обозначений национальной валюты и так далее.

В Linux стандартом на методику локализации является спецификация i18n. Не вдаваясь в подробности, мы попытаемся получить некоторое представление о

том как это выглядит для пользователя.

Как конечные пользователи, мы можем ознакомиться с текущими настройками своей локали используя команду `locale`:

```
$ locale
LANG=ru_RU.UTF-8
LC_CTYPE="ru_RU.UTF-8"
LC_NUMERIC="ru_RU.UTF-8"
LC_TIME="ru_RU.UTF-8"
LC_COLLATE="ru_RU.UTF-8"
LC_MONETARY="ru_RU.UTF-8"
LC_MESSAGES=en_US
LC_PAPER="ru_RU.UTF-8"
LC_NAME="ru_RU.UTF-8"
LC_ADDRESS="ru_RU.UTF-8"
LC_TELEPHONE="ru_RU.UTF-8"
LC_MEASUREMENT="ru_RU.UTF-8"
LC_IDENTIFICATION="ru_RU.UTF-8"
```

На самом деле, программа `locale` в корректно настроенной системе просто выведет вам значение соответствующих переменных окружения. Каждая из этих переменных отвечает за свой собственный аспект локализации, перечислим наиболее важные:

LANG – локаль по умолчанию, ее значение для различных аспектов может быть перекрыто путем установки отдельных переменных в нужное вам значение

LC_CTYPE – отвечает за классификацию символов и различия в их регистре

LC_NUMERIC – отвечает за представление и форматирование чисел

LC_TIME – отвечает за формат даты

LC_COLLATE – определяет настройки сравнения строк и символов, влияет на сортировку

LC_MONETARY – отвечает за представление национальной валюты

LC_MESSAGES – определяет язык сообщений и интерфейса

В примере мы видим что локалью по умолчанию через переменную `LANG` выбрана локаль `ru_RU.UTF-8` (русский язык, соответствие российским стандартам, кодировка символов UTF-8), а в качестве языка интерфейса назначен английский язык в варианте используемом в США. Фактически это означает что все программы будут иметь английский интерфейс, но показывать дату, валюту, десятичную точку и сортировать строки они должны так как это принято в России. В отличие от мира Windows, где язык сообщений программы как правило жестко забит в ресурсах, а ресурсы включены в исполняемый файл, в UNIX и Linux сообщения как правило содержатся в отдельных файлах и программа загружает текстовые строки для сообщений интерфейса основываясь на значениях переменных `LANG` и `LC_MESSAGES`. Это дает возможность обеспечить работу пользователей из разных стран на привычных им языках, не прибегая к установке нескольких копий операционной системы или нескольких экземпляров программы, каждый для своего языка.

Системному администратору Linux достаточно обеспечить наличие файлов сообщений нужной локали для используемых пользователем программ, и правильно установить переменные окружения, после чего пользователь оказывается в привычной для него языковой обстановке и работает с привычными форматами даты, валюты, в привычных единицах измерения и т.д.

Более того – можно установить значение некоторых переменных окружения в нужные значения непосредственно перед запуском программы – и тогда эта

программа примет будет использовать указанные настройки локали только для этого запуска, например:

```
[viking@alpha ~]$
[viking@alpha ~]$ LC_MESSAGES=ru_RU.UTF-8 ls -l something
ls: невозможно получить доступ к something: Нет такого файла или каталога
[viking@alpha ~]$
[viking@alpha ~]$ LC_MESSAGES=en_US.UTF-8 ls -l something
ls: cannot access something: No such file or directory
[viking@alpha ~]$
[viking@alpha ~]$ LC_MESSAGES=fr_CA.UTF-8 ls -l something
[viking@alpha ~]$
ls: ne peut accéder something: Aucun fichier ou répertoire de ce type
[viking@alpha ~]$
```

Таким образом, в Linux-системе возможно обеспечить одновременную работу пользователей в разных языковых окружениях, изменяя только изменение значений переменных окружения. В системах RedHat/Fedora общесистемные настройки локали хранятся в файле **/etc/sysconfig/i18n**, а пользователь может самостоятельно перекрыть их путем записи соответствующих значений в файле **~/.i18n**:

```
[viking@alpha ~]$
[viking@alpha ~]$ cat /etc/sysconfig/i18n
LANG="ru_RU.UTF-8"
SYSFONT="latarcyrheb-sun16"
[viking@alpha ~]$ cat /homed/viking/.i18n
LC_MESSAGES="en_US"
[viking@alpha ~]$
```

В данном примере системная локаль **ru_RU.UTF-8**, но пользователь **viking** предпочитает работать в англоязычном интерфейсе, что и записал в своем персональном файле настроек локали.

Файловые системы Linux

Создание файловых систем

Ссылка на оригинал: <http://linuxshop.ru/unix4all/?cid=&id=160>

Автор: Алексей Федорчук (alv@linux-online.ru)

С версии: 1.3

Создание файловых систем на дисковых разделах (или, в терминах DOS/Windows, форматирование последних) - второй этап подготовки диска к инсталляции Linux. Само по себе это действие - не из самых сложных, однако осознанное его выполнение требует некоторой подготовки.

Необходимое введение

Термин "файловая система" - один из самых многозначных в компьютерной терминологии вообще, и в литературе, связанной с ОС Unix и ее клонами (а Linux именно к таковым и относится) - в особенности: больше значений имеет, пожалуй, только просто слово "система". Так сложилось исторически, и потому значение термина приходится определять по контексту. Мне таких значений известно как минимум пять, и нет уверенности, что это - все возможные значения.

Во-первых, файловой системой или, иногда, подсистемой, называют часть ядра, которая управляет организацией файлов и операциями над ними: в этом аспекте в одном ряду с файловой системой выступают система управления процессами и система ввода/вывода. Впрочем, именно это значение мы пока рассматривать не будем.

Во-вторых, и это специфично для Unix и Unix-подобных ОС, файловая система - универсальный интерфейс доступа ко всем ресурсам, как локальной машины, так и компьютеров, объединенных в сеть любого рода - от модемного подключения до спутникового канала. Собственно говоря, именно задействие такого интерфейса в некоей ОС и являет собой один из поводов для отнесения ее к семейству Unix-подобных.

Во-третьих, файловая система - это способ описания некоего физического устройства (обычно - дискового раздела). Именно это значение используется в термине "идентификатор типа файловой системы", о котором говорилось в статье [о дисковых разделах](#). Он обычно специфичен для конкретной ОС, и потому здесь уместны термины: DOS-раздел, расширенный (Extended) раздел DOS, раздел Linux native и так далее. Хотя большинство всамделишных операционных систем способны тем или иным образом опознавать "неродные" для них идентификаторы и обращаться к данным на них. А в Linux в качестве "родных" (native) рассматривает дисковые разделы, идентифицированные целым рядом файловых систем в третьем понимании термина - от истинно родного Linux native до Extended DOS, включая логические тома (LVM) о которых говорилось в [соответствующей статье](#), и программные RAID-массивы.

Здесь следует отметить, что помимо файловых систем физических (то есть надстраивающих реальные дисковые устройства, почему в этом контексте нередко фигурирует выражение - disk based file system), существуют и виртуальные файловые системы различных типов. К ним относятся и файловая система устройств - devfs, и временная файловая система в оперативной памяти - tmpfs, и procfs - система, ответственная за представление в виде файловой системы (уж простите за тавтологию) процессов. Иногда используются и файловые системы промежуточного типа, например, виртуальные диски (RAM-диски). Подобно tmpfs, они существуют только в оперативной памяти, однако в остальном ведут себя также, как и файловые системы disk-based.

В четвертых, под файловыми системами понимается внутренняя управляющая структура, позволяющая хранить, идентифицировать и отыскивать данные, ну и, конечно, манипулировать ими. Такие структуры, с одной стороны, специфичны для операционных систем, как FAT16 (со всеми ее вариациями типа VFAT или FAT32) для DOS, UFS для FreeBSD или Ext2fs - для Linux. С другой же - структуры управления файлами в ряде операционных строятся по близким принципам, чему ярким примером служат файловые системы Unix- и Unix-подобных ОС. И потому они могут быть объединены в одно семейство, противопоставляемое FAT-семейству, например.

Кроме того, Linux в настоящее время способен работать с управляющими структурами различных типов - от Ext3fs, являющей собой надстройку над традиционной Ext2fs, до XFS и JFS, разработанных первоначально для версий Unix от SGI и IBM, соответственно, а также ReiserFS. Нет запрета и на размещение Linux'a на файловой системе типа FAT (хотя и резонов к тому - нет также).

Добавлю, что в списке из предыдущего абзаца перечислены только файловые системы, способные нести базовые компоненты Linux, отвечающие за ее запуск и минимальную функциональность. Что же касается обмена данными - таковой возможен из Linux'a практически со всеми известными файловыми системами, хотя с некоторыми из них (например, NTFS или HPFS) - только в режиме чтения.

Наконец, в пятых, файловая система в Unix - это и логическая структура каталогов и файлов, которая объединяет и физические, и виртуальные файловые системы самых различных типов (например, дисковые разделы с файловыми системами Ext2fs и FAT16, виртуальные procfs, devfs и tmpfs), причем не только на локальной машине, но и на любой удаленной. Структура эта - иерархическая, или древовидная, начинающаяся с корневого каталога, родительского по отношению ко всем прочим, от которого ответвляются отдельные файлы и дочерние каталоги, которые, в свою очередь, могут выступать как родительские по отношению к подкаталогам более глубоких уровней вложенности.

Положение дел в настоящий момент таково, что в Linux структура файловой системы обычно специфична для конкретного дистрибутива или их группы, связанной единством происхождения. Поэтому нередко можно столкнуться с такими выражениями, как файловая система Red Hat или Debian. Собственно, именно исторически сложившиеся различия в иерархии каталогов являются одним из критериев обособления нескольких линий дистрибутивов Linux. Как, впрочем, и потенциальной причиной их несовместимости. Однако можно надеяться, что усилиями стандартизирующих организаций, таких, как [Linux Standard Base](#) и [Filesystem Hierarchy Standard](#), русский перевод стандарта - на

[сайте Виктора Костромина](#)), увенчаются успехом, и можно будет говорить о единой логической файловой системы Linux, подобно тому, как это имеет место в ОС линии BSD.

В контексте настоящей статьи нас интересует только четвертый аспект файловых систем, то есть создание управляющих файлами структур, базируемых на дисках (вернее, их разделах).

Устройство файловых систем Unix-семейства

В этом разделе будет говориться о предметах, общих для всех Unix. Все файлы в Unix физически состоят из двух частей, реально локализованных в различных блоках дискового накопителя, но обязательно находящихся в одном дисковом разделе, первичном или логическом.

Первая часть файла - его так называемые метаданные, которые содержат файловый дескриптор (это просто некое уникальное число), сведения о его атрибутах (принадлежности, правах доступа, времени модификации и т.д.), а также информацию о том, в каких блоках дискового раздела (которые так и называются - блоки данных) физически размещено содержимое файла - те самые последовательности байтов, которые образуют доступный пользователю ASCII-текст или исполняемый модуль программы.

Метаданные каждого файла записаны в специальной области диска, называемой суперблоком, где образуют т.н. inodes (от information nodes - информационные узлы). Каждому существующему файлу соответствует свой inode, и именно он однозначно идентифицируется файловым дескриптором. А сам по себе список inodes, соответствующих как существующим файлам, так и свободным блокам дискового раздела, и определяет границы файловой системы, то есть сколько файлов может быть в ней создано.

Так вот, суть процесса создания файловой системы на дисковом разделе (или, в понимании DOS/Windows, его форматирования) - в создании на нем суперблока (или, в некоторых файловых системах, нескольких его копий), списка inodes и отведении дискового пространства под блоки данных (а также загрузочного блока, о котором будет сказано ниже), а устройством этих дисковых областей определяются различия между файловыми системами различных типов. В результате на новом разделе образуется единственный файл - каталог корневого (для данной файловой системы) раздела (в некоторых случаях создается еще и каталог /lost+found, предназначенный для хранения нарушенных файлов).

Возникает вопрос, почему такой, казалось бы неотъемлемый, атрибут файла, как его имя, не обнаруживается ни в его метаданных, ни, тем более, среди его данных. Ответ прост: в Unix имя являет собой атрибут не файла, но файловой системы (в пятом, логическом, понимании термина). И для хранения имен файлов предназначены файлы особого типа - каталоги (в Unix есть и другие типы файлов, например, упомянутые выше файлы устройств). Они представляют собой просто списки файловых дескрипторов идентификаторов и соответствующих им имен файлов. Поэтому идущая от MacOS и активно используемая в Windows метафора каталога как папки с документами - в Unix только затемняет суть дела: здесь это скорее именно каталожный ящик в библиотеке.

Не смотря на столь простое устройство, роль каталогов в файловой системе Unix трудно переоценить: имена файлов, через которые они включаются в файловую

систему (и через которые пользователь получает доступ к их содержимому), фигурируют только в составе каталога, к которому файл приписан - и больше нигде в системе. Так что удаление имени файла (или подкаталога) из списка, представляющего собой данные его родительского каталога (который, конечно, также имеет свой inode и файловый дескриптор, приписанный к каталогу, расположенному уровнем выше в иерархии файловой системы, и так далее) равносильно тому, что метаданные файла становятся недоступными, а приписанные к его inode блоки данных помечаются как свободные. Именно так осуществляется удаление файла командой `rm` или файловым менеджером типа `Midnight Commander`.

Нас, однако, сейчас интересует прямо противоположное - сделать файловую систему доступной. Из сказанного понятно, что для этого она со всем ее содержимым (суперблоком, списком inode, блоками данных) должна быть включена в состав какого-либо из существующих каталогов, называемого точкой монтирования. Именно это и составляет суть процесса монтирования. Результат же для монтируемой файловой системы - в том, что ее корневой каталог (до сих пор безымянный) получает имя каталога - точки монтирования (`mount point`), содержимое которого отныне составляет список имен ее файлов и подкаталогов. Обратный процесс - размонтирование, следствием чего является отсоединение от точки монтирования дерева смонтированной файловой системы. Кроме того, в inode ее корневого каталога устанавливается т.н. бит чистого размонтирования (`clean bit`). Впрочем, вопросам монтирования и размонтирования файловых систем будет посвящена специальная статья. Пока же рассмотрим особенности файловых систем, используемых в Linux'e.

Файловые системы Linux

Ext2fs

До недавнего времени список истинно родных (`native`) файловых систем для Linux ограничивался единственной - `ext2fs` (правда, Linux способен загрузиться и работать с FAT-раздела, но об этом мне даже не хочется говорить). Название это расшифровывается как "вторая расширенная файловая система"; "расширенная" она - по сравнению с файловой системой ОС `minix`, послужившей прототипом Linux, "вторая" - потому что ранние версии Linux базировались на `Extfs` с более ограниченными возможностями.

О файловой системе `Ext2fs` написано немало (см. дополнительные источники). Поэтому замечу только, что по способу организации хранения данных она - типичная представительница файловых систем Unix. Отличительная ее особенность - наличие нескольких копий суперблока, что повышает надежность хранения данных. Кроме того, для характерен очень эффективный механизм кэширования дисковых операций, что обеспечивает замечательное их быстродействие - едва ли не рекордное среди известных мне файловых систем. Обратная сторона чего, однако, - относительно слабая устойчивость при аварийном завершении работы (вследствие мертвого зависания или отказа питания), поскольку отложенность записи изменений файлов делает весьма высокой вероятность нарушения связи между их inodes и блоками данных.

Конечно, времена, когда некорректный останов Linux-машины грозил полным разрушением файловой системы, остались в далеком прошлом. Однако в любом

случае останова системы без штатного размонтирования файловых систем приводит к тому, что в них не устанавливается упомянутый выше "бит чистого размонтирования". А без этого утилиты обслуживания диска (такие, как программа проверки fsck) при перезагрузке не воспринимают их как целостные и начинают проверку, которая при современных объемах дисков может занять немалое время.

О журналируемых файловых системах

Проблема нарушения целостности файловой системы при некорректном завершении работы в большей или меньшей мере характерна для всех ОС семейства Unix. И потому с давних пор в них разрабатываются т.н. журналируемые файловые системы. Журнал -- это нечто вроде log-файла дисковых операций, в котором фиксируются не выполненные, а только предстоящие манипуляции с файлами, вследствие чего оказывается возможным самовосстановление целостности файловой системы после сбоя.

Во избежание недоразумений следует подчеркнуть, что журналирование направленно на обеспечение целостности файловой системы, но ни в коем случае не гарантирует сохранность пользовательских данных как таковых. Так, не следует ожидать, что журналирование волшебным образом восстановит не сохраненные перед сбоем изменения документа, загруженного в текстовый редактор.

Более того, в большинстве журналируемых файловых систем фиксируются грядущие операции только над метаданными изменяемых файлов. Обычно этого достаточно для сохранения целостности файловой системы и, уж во всяком случае, предотвращения долговременных их проверок, однако не предотвращает потери данных в аварийных ситуациях. В некоторых из файловых систем возможно распространение журналирования и на область данных файла. Однако, как всегда, повышение надежности за счет этого оплачивается снижением быстродействия.

Текущие версии ядра Linux поддерживают в качестве нативных четыре журналируемые файловые системы: ReiserFS и Ext3fs, специфичные для этой ОС, XFS и JFS - результаты портирования в Linux файловых систем, разработанных первоначально для рабочих станций под ОС Irix (SGI) и AIX (IBM), соответственно. Правда, широкое признание получили только три первых, так что о JFS я пока говорить не буду.

ReiserFS

Файловая система ReiserFS оказалась для Linux исторически первой из журналируемых - она поддерживается каноническим ядром с <http://www.kernel.org>, начиная с первых версий ветви 2.4.x (в настоящее время существуют патчи, позволяющие использовать ее и с версиями ветви 2.2.xx). И была единственной, разработанной "с нуля" специально для этой ОС Хансом Райзером и его фирмой [Namesys](http://www.namesys.com). Как и в большинстве рассмотренных, в ReiserFS осуществляется журналирование только операций над метаданными файлов. Что, при определенном снижении надежности, обеспечивает высокую производительность: по моим наблюдениям, на большинстве типичных пользовательских задач она лишь незначительно уступает Ext2fs. А на такой, достаточно обычной, операции,

как копировании большого количества мелких файлов, существенно ее опережает.

Кроме этого, ReiserFS обладает уникальной (и по умолчанию задействованной) возможностью оптимизации дискового пространства, занимаемого мелкими, менее одного блока, файлами (а следует помнить, что в любой Unix-системе такие файлы присутствуют в изобилии): они целиком хранятся в своих inode, без выделения блоков в области данных - вместе с экономией места это способствует и росту производительности, так как и данные, и метаданные (в терминах ReiserFS - stat-data) файла хранятся в непосредственной близости и могут быть считаны одной операцией ввода/вывода.

Вторая особенность ReiserFS - то, что т.н. хвосты файлов, то есть их конечные части, меньшие по размеру, чем один блок, могут быть подвергнуты упаковке. Этот режим (tailing) также включается по умолчанию при создании ReiserFS, обеспечивая около 5% экономии дискового пространства. Что, правда, несколько снижает быстродействие, и потому режим тайлинга можно отменить при монтировании файловой системы. Однако упаковка хвостов автоматически восстанавливается после перекомпиляции ядра - что, как будет сказано чуть ниже, требует внимательного отношения.

ReiserFS не совместима с Ext2fs на уровне утилит обслуживания файловой системы. Однако соответствующий инструментарий, объединенный в пакет reiserfsprogs, уже давно включается в штатный комплект современных дистрибутивов (или, в крайнем случае, может быть получен с сайта Namesys).

Более серьезная проблема с совместимостью - в том, что распространенные загрузчики Linux (и Lilo, и GRUB - хотя и по разным причинам) часто не способны загрузить ядро Linux с раздела ReiserFS, оптимизированного в режиме тайлинга. А поскольку, будучи отключен, этот режим обладает свойством самовосстановления, пользователь может столкнуться с тем, что после пересборки ядра система просто откажется загружаться. Именно поэтому выше я упоминал, что создание раздела под каталог /boot может быть необходимым.

Ext3fs

В отличие от ReiserFS, Ext3fs - не более чем журналируемая надстройка над классической Ext2fs, разработанная Стивеном Твиди в компании Red Hat и поддерживаемая ядром Linux, начиная с версии 2.4.16. Как следствие такого происхождения, она сохраняет со своей прародительницей полную совместимость, в том числе и на уровне утилит обслуживания (начиная с версии 1.21 объединяющего их пакета e2fsprogs). И переход от ext2fs к ext3fs может быть осуществлен простым добавлением файла журнала к первой, не только без переформатирования раздела, но даже и без рестарта машины.

Из этого вытекает первое преимущество ext3fs, особенно весомое в случае большого парка компьютеров. Второе же - чуть ли не максимальная надежность: ext3fs является единственной системой из рассматриваемых, в которой возможно журналирование операций не только с метаданными, но и с данными файлов.

В Ext3fs предусмотрено три режима работы - полное журналирование (full data journaling), журналирование с обратной записью (writeback), а также задействуемое по умолчанию последовательное (ordered).

Режим полного журналирования, как легко догадаться из названия,

распространяется и на метаданные, и на данные файлов. Все их изменения сначала пишутся в файл журнала и только после этого фиксируются на диске. В случае аварийного отказа журнал можно повторно перечитать, приведя данные и метаданные в непротиворечивое состояние. Этот механизм практически гарантирует от потерь данных, однако является наиболее медленным.

В режиме отложенной записи, напротив, в файл журнала записываются только изменения метаданных файлов, подобно всем рассмотренным ниже файловым системам. То есть никакой гарантии сохранности данных он не предоставляет, однако обеспечивает наибольшее (в рамках Ext3fs) быстродействие.

В последовательном режиме также физически журналируются только метаданные файлов, однако связанные с ними блоки данных логически группируются в единый модуль, называемый транзакцией (transaction). И эти блоки записываются перед записью на диск новых метаданных на диск, что, хотя и не гарантирует сохранности данных, весьма последней способствует. Причем - при меньших накладных расходах по сравнению с полным журналированием, обеспечивая промежуточный уровень быстродействия.

Однако, по моим наблюдениям Ext3fs ни в одном из режимов не может похвастаться высокой производительностью при решении настольных пользовательских задач. А в режиме полного журналирования возможны ситуации, вызывающие на некоторое время полный паралич машины.

XFS

Файловая система XFS, в отличие от молодых ReiserFS и ext3fs, развивается для фирмой SGI на протяжении почти десяти лет - впервые она появилась в версии Irix 5.3, вышедшей в 1994 г. Но в Linux она была портирована лишь недавно (текущая ее версия - 1.1, свободно доступна с сайта SGI's XFS page - <http://oss.sgi.com/projects/xfs>) и по сию пору не поддерживается официальным ядром.

XFS - единственная 64-разрядная файловая система из рассмотренных. Однако уникальность ее - не только в этом. Особенности XFS являются:

- механизм allocation group, то есть деление единого дискового раздела на несколько равных областей, имеющих собственные списки inodes и свободных блоков, для распараллеливания дисковых операций;
- логическое журналирование только изменений метаданных, но - с частым сбросом их на диск для минимизации возможных потерь при сбоях;
- механизм delayed allocation - ассигнование дискового пространства при записи файлов не во время журналирования, а при фактическом сбросе их на диск, что, вместе с повышением производительности, предотвращает фрагментацию дискового раздела;
- списки контроля доступа (ACL, Access Control List) и расширенные атрибуты файлов (extended attributes), рассмотрение которых далеко выходит за рамки нынешней темы.

В результате XFS предстает как очень сбалансированная файловая система: она почти столь же надежна, как Ext3fs, и не очень уступает ReiserFS в быстродействии на большинстве файловых операций. А при манипуляциях с (очень) большими файлами XFS - просто вне конкуренции: как легко догадаться по имени фирмы-разработчика, она ориентирована на работу с мультимедийными приложениями с их огромными потоками данных. Не отмечалось для нее и

проблем с совместимостью.

Все сказанное позволяет сделать вывод, что XFS - оптимальная файловая система для Linux. Однако следует учесть: в отличие от ReiserFS и ext2fs, поддержка которых является штатными опциями ядра Linux, XFS по сию пору (текущая версия - 2.4.19) не поддерживается каноническим ядром Линуса Торвальдса (тем, которое можно получить с <http://www.kernel.org>). Хотя недавнее включение такой поддержки в разрабатываемую ветвь ядра (версии 2.5.X) позволяют надеяться, что скоро эта функция станет штатной.

Возможность работы с XFS обеспечивает специальный патч (xfs-2.4.1X-all-i386.bz2), который можно получить с сайта [SGI](http://www.sgi.com) вместе с соответствующими утилитами поддержки: традиционные средства e2fsprogs, для XFS не пригодны. Утилиты поддержки для XFS объединены в несколько пакетов, из которых абсолютно необходимым является xfsprogs. Обо всем этом следует помнить при предварительной разметке диска.

Критерии выбора

Таким образом, каждая из четырех рассмотренных файловых систем имеет свою уникальную положительную особенность (даже Ext2fs - как бы то ни было, лидером по суммарному быстродействию остается она), и как минимум один недостаток (который, тем не менее, не служит препятствием к ее использованию). Так что выбор файловой системы должен определяться задачами пользователя и характером преобладающих его данных.

Кроме этого, если, как рассказывалось в статье про [LVM](#), максимально разгрузить корневую файловую систему, выделив в логические тома отдельные ее ветви, возможно и комбинированное использование файловых систем. То есть подбор для каждого из дисковых разделов наиболее подходящей к его содержанию управляющей структуры.

Так, ранее неоднократно говорилось, что Ext2fs - наиболее подходящий выбор для загрузочного раздела (а при использовании в качестве загрузчика GRUB - это почти обязательное требование). Кроме того, Ext2fs вполне подходит для таких ветвей, как /tmp или /var. Для первого, по определению, устойчивость к сбоям не критична. Для второго же определяющим требованием является быстродействие (так, в Source Based дистрибутивах типа Gentoo /var задействуется для хранения временных продуктов компиляции, и быстродействие файловых операций в нем несколько способствует ускорению сборки пакетов). Наконец, на настольной машине Ext2fs можно применить и для корневой файловой системы - ведь при дробном разбиении диска в корне остается минимум редко изменяемых компонентов.

С другой стороны, корень - наиболее критичный в отношении устойчивости элемент файловой системы. И потому оптимальным для него представляется файловая система Ext3fs, как наиболее устоявшаяся. Кроме того, в экстремальных ситуациях она может быть без проблем смонтирована как Ext2fs. Для разделов типа /usr и /usr/local Ext3fs также видится вполне подходящим вариантом.

Наиболее важная часть файловой системы настольной машины с точки зрения пользователя - его, пользовательские, данные, то есть каталог /home (ибо систему можно переустановить, а вот потеря данных может быть невосполнимой). Однако это - и наиболее изменяемая ее часть, что предъявляет высокие требования к

быстродействию файловых операций. И поэтому Ext3fs - не лучшее (ИМХО) решение для каталога /home, более целесообразно разместить здесь какую-либо из "быстрых" журналируемых файловых систем, ReiserFS или XFS. Выбор между ними определяется личными предпочтениями и характером данных (пользуясь случаем, замечу, что быстродействие JFS, по моим наблюдениям над типичными пользовательскими манипуляциями, оставляет желать лучшего).

Очевидно, что быстродействие XFS при работе с файлами (очень) большого размера делает ее предпочтительной, если речь идет об обработке изображений, мультимедийном контенте, картографической информации и т.д. В то же время преимущества ReiserFS сказываются в основном при работе с файлами (очень) маленького размера (менее блока файловой системы), каковых среди пользовательских данных обычно бывает мало. И потому мое личное мнение однозначно - и высказывается в пользу XFS. К тому же собственный опыт общения с ReiserFS был неблагоприятен, особенно в сочетании с технологией LVM. XFS же, напротив, по моим впечатлениям идеально с ней гармонирует.

Подведем итог - оптимальной мне видится следующее сочетание файловых систем:

- Ext3fs - для корневого каталога (/) и каталога /usr (а также /usr/local и /usr/X11R6, если таковые обособляются в отдельные ветви);
- Ext2fs - для загрузочного /boot, каталогов /tmp и /var;
- XFS - для раздела под домашние каталоги (/home).

Повторяю, это лишь мое мнение, основанное на опыте настольного применения Linux - для серверов различного рода оно силы не имеет.

Практические следствия

Надеюсь, предыдущий раздел дал достаточно информации для выбора файловой системы, наиболее подходящей к конкретным условиям. И потому далее вкратце рассмотрим последовательность действий после того, как выбор осуществлен.

Обычно при установке Linux создание файловых систем - компетенция инсталлятора, который осуществляет его с некоторыми опциями по умолчанию. В контексте рассмотренного в предыдущем разделе ясно, что опции эти не всегда будут лучшими с точки зрения пользователя. Изменить же характеристики, определенные для файловой системы при ее создании, невозможно без повторного выполнения этого процесса (и, соответственно, потери данных). Однако все SB-дистрибутивы допускают ручное вмешательство в процедуру создания файловой системы, а в Gentoo, например, это - просто единственный способ ее выполнения.

Файловая система Ext2fs может быть создана любой из следующих команд - /sbin/mke2fs, /sbin/mkfs, /sbin/mkfs.ext2 с указанием файла устройства в качестве аргумента, например:

```
$ /sbin/mke2fs /dev/hd?#
```

Каждая из этих команд (а /sbin/mkfs.ext2 - обычно символическая ссылка на /sbin/mke2fs) имеет ряд опций, как то: -b для определения размера блока файловой системы (возможные значения - 1024, 2048 или 4096 байт, по умолчанию принято последнее), -c для проверки испорченных участков диска, -N и -i для задания числа inodes и количества байт на один узел, соответственно; с деталями можно

ознакомиться на соответствующей man-странице, например, man (8) mke2fs.

Для создания файловой системы ext3fs можно применить ту же команду mke2fs с опцией -j, при этом случае она получит некоторые "умолчальные" характеристики. Определить же их вручную позволяет следующая форма этой команды:

```
$ /sbin/mke2fs -J опции_журналирования /dev/hd?#
```

Возможные значения опций журналирования - size=размер, задающая объем журнального файла в мегабайтах, и device=внешний_журнал подключения новой файловой системы к журналу, ранее созданному на другом дисковом разделе.

Можно использовать и специальную команду /sbin/mkfs.ext3 - возможности ее идентичны таковым /sbin/mke2fs (ибо она ни что иное, как символическая на нее ссылка). Но самое интересное - возможность преобразования существующей ext2fs в ext3fs простым добавлением журнала, не только без потери данных, но и без перезапуска системы (и даже без размонтирования). Делается это командой

```
$ tune2fs -j /dev/hd?#
```

Она просто добавляет файл журнала /.journal в корневом каталоге модифицируемой файловой системы (если последняя не была размонтирована), или задействует для журнала скрытый inode (если перед модификацией файловая система была размонтирована). Добавлю, что обратное преобразование - еще проще, и осуществляется командой монтирования (о чем будет говориться в следующей статье).

Файловая система ReiserFS создается специально предназначенной для этого командой - /sbin/mkreiserfs из пакета reiserfsprogs. Для нее доступны многочисленные опции (-s для задания размера журнала, -f для принудительного переформатирования ранее существовавшей файловой системы иного типа, и т.д.), с которыми можно ознакомиться посредством man (8) mkreiserfs. И во избежание неожиданностей напомним: если корневой раздел форматируется как ReiserFS, не лишним будет предусмотреть небольшой раздел под каталог /boot для размещения на нем файловой системы ext2fs.

Для создания XFS также существует собственная команда mkfs.xfs (из пакета xfsprogs). В ней предусмотрено несколько опций, каждая из которых имеет ряд субопций, принимающих численные значения. Важнейшие из них:

- -b, которая посредством субопции size=## позволяет задать размер блока данных в байтах, который должен быть кратен размеру страницы оперативной памяти (для платформы i386 - 4 Кбайт) и может варьировать в диапазоне от 512 до 65536 (по умолчанию - 4096);
- -d, определяющая параметры области данных файловой системы, такие, как количество самостоятельных областей раздела (Allocation groups, субопция agcount), или, напротив, их размер (субопция agsize);
- -l, специфицирующая параметры журнального файла, например, его размер (субопция size).

При использовании mkfs.xfs для достижения максимальной производительности рекомендуется в явном виде задать количество allocation groups - иначе оно будет определяться автоматически, что ведет к непроизводительным расходам ресурсов. Это делается из расчета - одна allocation group на 4 Гбайт дискового пространства. Далее можно установить размер файла журнала - здесь рекомендованное значение составляет 32 Мбайт. То есть для дискового раздела

объемом в 20 Гбайт команда приобретет вид

```
$ mkfs.xfs -d agcount=5 -l size=32m /dev/hda1
```

Кроме всех перечисленных, команда `mkfs.xfs` имеет опцию `-f` (от `force`) - принудительное создание файловой системы XFS поверх любой существующей. Ее достаточно, если последняя была `ext2fs` (и, исходя из общих соображений, `ext3fs`, хотя я этого не проверял). Если же XFS создается поверх `ReiserFS` - после этого возможны ошибки при монтировании новой файловой системы. Впрочем, то же относится и к обратной процедуре (замене XFS на `ReiserFS`), а также, если любая из этих "продвинутых" файловых систем заменяется на разделе системой `ext2fs`. Они связаны с тем, что команда монтирования может распознать новосозданную XFS как дефектную `ReiserFS`, и наоборот.

Во избежание этого перед таким замещением приходится прибегать к несколько шаманскому приему - обнулению начальных областей раздела (хранящего метаданные файловой системы) командой

```
$ dd if=/dev/zero of=/dev/hd?#
```

Ждать заполнения нулями всего устройства не обязательно - достаточно дать этой команде поработать секунд 10-20, после чего прервать ее комбинацией клавиш `Control+D` и перейти к созданию новых файловых систем.

И последнее, о чем следует сказать - о `swar`-разделе, созданном на этапе разбиения диска. Хотя файловой системы как таковой он не несет, но нуждается в определении, что достигается командой

```
$ mkswap имя_устройства
```

к которой следует подходить со вниманием - применение ее к обычному разделу уничтожит на нем все данные.

Библиография вопроса

Многие сложные вопросы устройства файловых систем в этой статье были затронуты лишь вскользь. За более подробной информацией по ним следует обратиться к дополнительным источникам. Так, общая организация файловой системы Unix рассматривается во многих руководствах по этой операционной системе, например: С. [Д. Кузнецов. Операционная система UNIX](#).

Устройство файловой системы `ext2fs` подробно описано в статье Виктора Хименко "Файлы, файлы, файлы" (Мир ПК, 2000, [часть 1 - #2](#); [часть 2 - #3](#),).

Подробное описание современных журналируемых файловых систем, используемых в Linux, дано в цикле статей Дэниела Роббинса, русский перевод которого, выполненный Владимиром Холмановым, доступен на сайте [Ярославской группы пользователей Linux](#).

Linux и файловые системы: еще раз о проблеме выбора

Ссылка на оригинал: <http://citkit.ru/articles/381/>

Автор: Алексей Федорчук (alv@linux-online.ru)

Дата: 15.03.2007.

С версии: 1.3

Весна наступила... Индо взопрели озимые, солнышко рассупонилося. Старик Ромуальдыч начал к своей портянке прияхиваться - оттаяла, видать. Податтаили и пользователи - на форумы потянулись, вопросы задавать, да ответы получать...

Одним из популярных вопросов в нынешнем сезоне оказался такой: как разбить диск и какую файловую систему выбрать.

Общий ответ на первую половину вопроса дать невозможно, а варианты решения этой проблемы обсуждались столько раз, что повторять их было бы скучно. А потому перейдем сразу ко второй половине - о дисковой разметке я вскользь скажу в самом конце.

В качестве "родных" для Linux (то есть тех, на которые он может быть установлен и с которых способен стартовать) рассматриваются следующие файловые системы: ext2fs, ext3fs, ReiserFS, XFS, JFS. Именно они обычно и предлагаются на выбор при установке подавляющего большинства дистрибутивов. Конечно, существуют и способы установки Linux на файловые системы FAT/VFAT/FAT32, но это - только для тех медам и мсье, которые понимают толк в извращениях, и о них я говорить не буду.

Исключу из рассмотрения также JFS - по следующим причинам:

- малой распространенности среди пользователей Linux;
- недостаточности источников информации по ней (именно в Linux - для родной ОС AIX эта файловая система прекрасно документирована);
- моим лично слабым с ней знакомством.

Основными критериями при выборе файловой системы являются обычно надежность и быстродействие. В некоторых случаях приходится учитывать также фактор совместимости - в данном случае под ней понимается способность других операционок обращаться к той или иной файловой системе.

Начну рассмотрение с ReiserFS - потому что поводом к сочинению этой заметки послужил вопрос: а что следует считать маленькими файлами? Ведь общеизвестно, что именно эффективность работы с мелкими файлами является сильной стороной этой файловой системы.

Так вот, под мелкими файлами понимаются файлы размером меньше логического блока файловой системы, который в Linux в большинстве случаев равен четырем килобайтам, хотя и может задаваться при форматировании в некоторых пределах (зависящих от конкретной FS). Таких мелких файлов в любой Unix-подобной ОС - бесчисленное количество. Типичным примером являются файлы, составляющие дерево портов FreeBSD, портежей Gentoo и тому подобных портообразных систем.

В большинстве файловых систем для таких минифайлов существует как свой

inode (информационный узел, содержащий метаданные о файле), так и блок данных, что приводит как к расходу дискового пространства, так и снижению быстродействия файловых операций. В частности, именно в этом причина катастрофической задумчивости файловой системы FreeBSD (как старой, UFS, так и новой, UFS2) при работе с собственной же системой портов.

В файловой системе ReiserFS в таких случаях отдельные блоки под данные не выделяются - она умудряется записать данные файла непосредственно в область его же inode. За счет этого и дисковое пространство экономится, и быстродействие возрастает - буквально в несколько раз по сравнению со всеми прочими FS.

Такое обращение с мелкими файлами ReiserFS послужило причиной возникновения легенды о ее ненадежности. Действительно, при крахе файловой системы (то есть разрушении служебных областей) данные, размещенные совместно со своими inodes, вместе с ними же и пропадают - причем безвозвратно. Тогда как в тех файловых системах, где inodes и блоки данных всегда разобщены пространственно, последние теоретически можно восстановить. Так, для ext2/ext3 даже существуют средства, позволяющие это сделать.

Однако, как и всякая легенда, эта лишь производит впечатление достоверности. Во-первых, безвозвратная потеря данных относится лишь к очень маленьким файлам. Среди пользовательских таковых практически не бывает, а все прочие же легко восстанавливаются из дистрибутива.

Во-вторых, говоря о возможности восстановления данных из блоков, утративших привязку к своим inodes, я не случайно употребил слово "теоретическая". Потому что на практике это занятие чрезвычайно трудоемкое, не дающее гарантированного результата. Каждый, кому приходилось этим заниматься, согласится, что предаться ему можно только от полной безысходности. И это относится ко всем файловым системам Linux. Так что этим аспектом при выборе файловой системы можно пренебречь.

По суммарному быстродействию ReiserFS однозначно быстрее всех остальных журналируемых FS, а по некоторым показателям превосходит и ext2. С результатами сравнения скорости выполнения некоторых распространенных файловых операций можно ознакомиться [здесь](#).

А вот с совместимостью у ReiserFS дело обстоит несколько хуже. Доступ к ней из ОС семейства Windows, насколько мне известно, невозможен. В некоторых операционках семейства BSD (DragonFlyBSD, FreeBSD) реализована поддержка этой файловой системы, но в режиме только для чтения. Даже вероятность того, что произвольный Linux LiveCD прошлых лет не имеет поддержку ReiserFS, не нулевая.

И тут впору вспомнить об ext3fs. Преимущество ее вовсе не в большей надежности - это такая же легенда, как и неустойчивость ReiserFS. О случаях краха ext3fs я слышал не меньше, чем об аналогичных происшествиях с ReiserFS. Самому мне не удавалось порушить ни ту, ни другую. Разве что с ext2 получалось - но и то очень давно, во времена ядра 2.2 (или даже еще 2.0).

Нет, главное преимущество ext3fs в ее совместимости - она с гарантией будет прочитана любой Linux-системой. Например, при восстановлении с какого-нибудь древнего подручного LiveCD - ситуация, практически не столь уж невероятная, мне приходилось в нее попадать. Опять же, большинство BSD-систем легко

понимают ext3fs (хотя и без журналирования). Для Windows также имеются, насколько я знаю, всякого рода драйверы и plug-ins к распространенным файловым менеджерам (типа Total Commander), обеспечивающие доступ к разделам с ext2fs/ext3fs.

В отношении производительности ext3fs оставляет противоречивое впечатление. Во-первых, быстродействие ее очень зависит от режима журналирования, каковых предусмотрено три: с полным журналированием данных, частичным их журналированием и журналированием только метаданных. В каждом из режимов она показывает различную производительность на разных типах файловых операций. Впрочем, ни в одном случае быстродействие не является рекордным, в чем можно убедиться, пройдя по [указанной выше ссылке](#).

Впрочем, если требование быстродействия ставится на первое место, то тут вне конкуренции оказывается ext2fs - правда, в этом случае придется смириться с отсутствием журналирования вообще. И, следовательно, с длительными проверками файловой системы при любом некорректном завершении работы - а при объемах современных дисков это может затянуться ой как надолго...

Относительно XFS можно сказать следующее. В плане совместимости к ней относится все то же самое, что написано для ReiserFS - более того, до некоторого времени она не поддерживалась стандартным ядром Linux. С точки зрения быстродействия она XFS она также не блещет, выступая суммарно примерно на одном уровне с ext3fs. А на операции удаления файлов вообще демонстрирует удручающую медлительность.

По моим наблюдениям, использование XFS оправдывает себя при работе не просто с большими, а с очень большими файлами - каковыми являются фактически только образы DVD и видеофайлы.

Возвращаясь к вопросу о надежности. Банальное выключение питания в ходе обычной пользовательской работы, как правило, безболезненно переносят все журналируемые файловые системы (и ни одна из них не обеспечивает сохранности не записанных на диск пользовательских операций - спасение утопающих и тут остается делом рук самих утопающих). Правда, для любой файловой системы можно смоделировать ситуацию, в ходе которой выключение питания приведет к более или менее серьезным ее повреждениям. Однако в реальной жизни возникновение таких ситуаций маловероятно. А полностью исключить их можно приобретением источника бесперебойного питания - он придаст больше уверенности в сохранности данных, чем тип файловой системы. Ну а гарантией восстановления разрушенных данных в любом случае может быть только их регулярное резервное копирование...

Думаю, изложенной выше информации достаточно для осознанного выбора. Мой личный выбор в течении последних нескольких лет - ReiserFS. Изредка, на системах, где оправданно вынесение за пределы корневого раздела всего, чего только можно, целесообразно использование ext3fs для корневой файловой системы и ReiserFS - для всех остальных.

Если предусматривается отдельный раздел под каталог /boot (а это рекомендуется при использовании загрузчика GRUB его разработчиками) - для него никакая другая файловая система, кроме ext2fs, не оправдана, какое-либо журналирование тут смысла не имеет. Наконец, если создается отдельный раздел под всякого рода мультимедийные материалы - тут можно подумать и о XFS.

В заключение добавлю, что более подробные сведения обо всех рассмотренных здесь материях можно получить из [цикла статей Дэниеля Роббинса](#).

Windows-эквиваленты

Ссылка на оригинал: <http://ru.gentoo-wiki.com/Windows-эквиваленты>

С версии: 1.0

Обновлено: 1.4

Вы знаете, что хотите сделать, но не знаете, чем? Это список пакетов в портежах Gentoo и их эквиваленты в Windows. Он, конечно же, далеко не полный, но и не предназначен для этого. Он просто даёт возможность лёгкого старта в Gentoo.

Для расширения этой таблицы можно использовать статью: [Аналоги Windows-программ в Linux - таблица соответствий](#). Также, весьма полный лист можно найти здесь: [\[1\]](#)^(англ.).

Смотрите также [официальную страницу портежей](#)^(англ.) с портежами, классифицированными по [категориям](#)^(англ.)

Прикладные и системные программы

Категории ПО	Программы MS Windows	Программы Linux	
		Аналоги	Портежи
Антивирусы	много	ClamAV	clamav
Текстовые редакторы	Блокнот / Notepad, WordPad, TextPad, Vim, Xemacs,...	Kedit (KDE) ^(грф.) Gedit (Gnome) Kate (KDE) Nedit Vim ^(кнс.) Xemacs Nvi	kedit gedit kedit nedit vim xemacs nvi
Запись CD/DVD	BurnAtOnce, Nero	Graveman K3b	graveman k3b

Работа в сети

Категории ПО	Программы MS Windows	Программы Linux	
		Аналоги	Портежи
Web-браузеры	Internet Explorer, Mozilla , Firefox , Opera...	Epiphany ^(грф.) Galeon ^(грф.) Konqueror Links ^(кнс.) Mozilla ^(грф.) Firefox ^(грф.)	epiphany galeon konqueror links mozilla mozilla-firefox

		Nautilus ^(грф.) Opera ^(грф.)	nautilus opera
Почтовые программы	Outlook Express, Netscape/Mozilla, Thunderbird, The Bat, Eudora, Becky, Datula, Sylpheed/Sylpheed-claws, Opera	Evolution ^(грф.) Mozilla messenger Sylpheed Sylpheed-claws Kmail Gnus ^(кнс.) Balsa Gnumail Opera Thunderbird Mutt ^(кнс.) Pine ^(кнс.)	evolution mozilla sylpheed sylpheed-claws kmail gnus balsa gnumail opera thunderbird mutt pine
Клиенты быстрого обмена сообщениями (IM)	ICQ Lite, ICQ Corp, MSN, AIM, Yahoo, ... Trillian ICQ (бесплатный, может заменить все 3 распространённых клиента), Miranda, Gaim	Licq ^(ICQ) Centericq ^(Bce) Micq ^{(ICQ)(кнс.)} GnomeICU ^(ICQ) Gaim ^(Bce) Ayttn ^(Bce) Kopete Simple Instant Messenger ^(Bce) aMSN ^(MSN) YSM ^{(ICQ)(кнс.)} Yahoo Messenger for Unix Kmess ^(MSN) kmess	licq centericq micq gnomeicu gaim ayttm kopete sim amsn ysm ymessenger
Видеоредакторы и конвертеры	VirtualDub , VirtualDubMod	Avidemux	avidemux
Virtual CD	VirtualDrive, VirtualCD, Daemon Tools, ...	<code>mount -o loop "mycd.iso" /mnt/cdrom</code>	
Другое	Blue Screen of Death	XScreenSaver	xscreensaver

Офис/Бизнес

Категории ПО	Программы MS Windows	Программы Linux	
		Аналоги	Портежи
Офисный пакет	MS Office, StarOffice / OpenOffice, 602Software	OpenOffice Koffice	openoffice

			koffice
Текстовый процессор	Word, StarOffice / OpenOffice Writer, 602Text, Abiword	OpenOffice Writer KWord AbiWord	openoffice kword abiword
Электронные таблицы	Excel, StarOffice / OpenOffice Calc	OpenOffice Calc KSpread Gnumeric	openoffice kspread gnumeric

Chat

- *amsn*: alternative for MSN Messenger
- *gaim*: alternative for Aim, ICQ, IRC, Yahoo!, MSN, Jabber (Google Talk)
- *irssi*: (command line based) alternative for mIRC
- *kopete*: alternative for Aim, ICQ, IRC, Yahoo!, MSN, Jabber
- *licq*: alternative for ICQ
- *psi*: jabber client
- *xchat*: альтернатива mIRC
- [KVirc](#): альтернатива mIRC
- *konversation*: alternative for mIRC
- *Skype*: Skype has also version in Linux
- *WengoPhone*: Wengophone also has a Linux version and is open source
- *SIM-IM*: alternative for ICQ, Jabber, MSN, SMS, Yahoo!, AIM

File managers

- *krusader*: alternative for Total Commander
- [GNOME Commander](#)

FTP

- *gFTP*: alternative for FileZilla, SmartFTP

Graphics

- *digiKam*: alternative for the proprietary digital photo software bundled with your camera
- *gimp*: alternative for Adobe Photoshop
- [Inkscape](#): alternative for Adobe Illustrator
- *gqview*: alternative for Irfanview
- [gwenview](#): alternative for Irfanview
- *kuickshow*: alternative for Irfanview

Internet

- *Konqueror*: alternative for Internet Explorer
- *Mozilla Firefox*: there is no alternative!
- *Seamonkey* (previously called *Mozilla*): alternative for Internet Explorer
- *Opera*: free but proprietary alternative for Internet Explorer

Mail

- *Evolution*: alternative for Outlook
- *Mozilla Thunderbird*: alternative for Outlook Express
- *Kontact / kmail*: alternative for Outlook Express

Multimedia

- *AmaroK*: alternative for iTunes player/WMP
- *XMMS*: alternative for WinAMP <http://komnas.com/~ashamril/tips/files/win-vs-lin-soft-EN.shtml>
- *beep-media-player*: alternative for WinAMP
- *kaffeine*: alternative for Windows Media Player (also plays DVDs)
- *MPlayer* | *gmplayer* | *kmplayer* | *kplayer*: alternative for Windows Media Player
- *Xine*: alternative for Windows Media Player (also plays DVDs)
- *vlc*: alternative for Windows Media Player
- *RealPlayer*: runs natively on linux!
- *Totem* : alternative for WMP, Menusupport for DVDs
- *LMMS* : "Linux Music Maker Studio" Linux analog of "FL Studio"
[HomePage:http://www.lmms.sourceforge.net/](http://www.lmms.sourceforge.net/)
- *Rosegarden* : MIDI/Audio-sequencer
[HomePage:http://www.rosegardenmusic.com/](http://www.rosegardenmusic.com/)
- *Cheese-Tracker* : one of the Trackers, use QT

Office

- *OpenOffice.org*: alternative for MS Word, Excel, Powerpoint and Access, great support for MS formats
- *KOffice* : like above without MS format support. Includes more specialized apps than even the full MS Office Pro
- *AbiWord* : Only like MS Word. Decent MS Format support. Includes grammar check
- *StarOffice*: commercial version of OpenOffice.org, includes extras (grammar check)
- *kpdf*, *xpdf*, *epdfview*, *evince*: PDF readers, alternative for Adobe Acrobat Reader

P2P

- *Azureus*: java bittorrent client (or *bittorrent* which is written in python)
- *qtorrent*, *ktorrent*: bittorrent clients
- *Limewire* + *gtk-gnutella*: Gnutella client
- *MLDonkey* + *Sancho*: client and interface for bittorrent, Gnutella, FastTrack, etc.
- *aMule*: alternative for eMule
- *xmule*: alternative for eMule
- *nicotine*: soulseek client
- *Valknut*, [LinuxDC++](#): альтернативные клиенты для DC (Direct Connect)

О Gentoo Linux

Ссылка на оригинал: <http://gentoo.ru/about>

С версии: 1.0



Gentoo — английское название вида пингвинов **Pygoscelis papua** (русские варианты названия: субантарктический пингвин, папуанский пингвин, ослиный пингвин). Согласно Книге рекордов Гиннеса 98 этому виду пингвинов принадлежит рекорд скорости плавания (27км/ч).

Gentoo Linux — дистрибутив, который может быть автоматически оптимизирован и приспособлен для применения практически в любой области в соответствии с вашими требованиями. **Наивысшая производительность, настраиваемость и высококлассное сообщество пользователей и разработчиков** — вот отличительные черты Gentoo.

Благодаря технологии портежей, Gentoo Linux станет идеальным защищенным сервером, рабочей станцией разработчика, профессиональной настольной системой, системой для игр, встроенной системой или чем-либо другим — всем, чем вы пожелаете. Из-за практически неограниченной способности дистрибутива подстраиваться под нужды пользователя, мы называем Gentoo Linux "**метадистрибутивом**".

Что такое портежи?

Portage — сердце Gentoo Linux, выполняющее множество ключевых функций. Например, portage — это система распространения программного обеспечения Gentoo Linux. Для того, чтобы получить доступ к последним версиям программ, в Gentoo Linux достаточно набрать всего одну команду: **emerge --sync** (emerge; средство установки/удаления программ (дословно: **emerge** - *выяснять, всплывать, появляться*)). Эта команда заставляет Portage обновить локальное дерево портежей вашей системы через интернет. В локальном дереве портежей содержится полная коллекция сценариев, которые используются для сборки и установки последних версий пакетов Gentoo. Сейчас в нашем дереве Portage более 10000 пакетов, а обновления и новые пакеты добавляются каждый день.

Portage также является системой сборки и установки пакетов. Желая установить пакет, нужно просто набрать: **emerge имя_пакета**, при этом Portage автоматически соберет вариант пакета «на заказ» в точности по вашим указаниям, оптимизируя его под ваше оборудование и гарантируя, что нужные вам дополнительные возможности включены, ненужные — нет.

Portage также следит за тем, чтобы ваша система не устаревала.

Команда: **emerge -u world** — всего одна команда — позаботится о том, чтобы все

нужные именно вам пакеты в системе автоматически обновились.

Gentoo Linux

С помощью портежей можно обновлять систему Gentoo Linux при малейшем желании. Поэтому опытные пользователи Gentoo не обращают особого внимания на «новые версии» Gentoo Linux: в конце концов, свежайшую версию Gentoo Linux всегда можно получить, набрав команду: ***emerge --sync***. Нет необходимости ждать несколько месяцев выпуска «новой версии» дистрибутива, поскольку Gentoo Linux непрерывно обновляется и улучшается, а все изменения немедленно становятся доступны пользователям.

Безусловно, мы издаем официальные выпуски Gentoo Linux на компакт-дисках, чтобы вновь устанавливаемые системы с самого начала были как можно свежее. Вот краткий обзор того, что включено в самый свежий выпуск Gentoo Linux:

- ***поддержка процессоров x86, AMD64, PowerPC, UltraSparc, Alpha и MIPS***
- ***установка на x86, AMD64, PowerPC, UltraSparc и Alpha с диска LiveCD***
- ***новейшие стабильные KDE и GNOME***
- ***различные оптимизированные ядра Linux***
- ***весьма современная среда разработки GNU***
- ***отличная поддержка файловых систем: ReiserFS, XFS, ext3, EVMS, LVM***
- ***превосходная поддержка оборудования: NVIDIA, Creative Labs Live! и Audigy***
- ***модульная подсистема OpenGL и компилятора (с поддержкой сосуществования нескольких версий)***
- ***понятные сценарии запуска системы, основанные на зависимостях***
- ***новая инициатива системы безопасности «укрепленный Gentoo»***
- ***более 10000 пакетов лучших новейших программ***
- ***расширенные возможности системы портежей***

Если вас привлекли мощь, гибкость и скорость Gentoo Linux, обязательно испытайте его в деле. Мы уверены: вы не разочаруетесь!

История Gentoo

Все началось с Лишнего Времени. Времени — на изучение, Времени — на исследования, Времени — на эксперименты... Именно так создатель Gentoo Дэниел Роббинс шагнул в мир Linux. Он начал с Debian Linux, настроил пару приложений, изучил Linux от «А» до «Я», а затем, как поступает большинство пользователей Linux, перепробовал еще парочку дистрибутивов, принявшись помогать дистрибутиву под названием Stampede Linux. И вскоре, погрузившись в разработку Stampede, он уже работал над их системой управления пакетами.

Через некоторое время, по определенным причинам, он пошел дальше и решил создать свой собственный дистрибутив.

Так родился «Енох». Дэниел хотел, чтобы Енох стал чрезвычайно быстрым дистрибутивом, способным полностью автоматизировать процесс создания и обновления пакетов. Вскоре появился канал #enoch на irc.freenode.net и 10 разработчиков, участвующих в создании дистрибутива. По прошествии времени, когда Енох подрос, им показалось, что дистрибутиву нужно новое имя. Они нарекли его Gentoo Linux. И как раз в то время, когда выход Gentoo 1.0 был на носу, Дэниел купил новую быструю машину. В материнской плате оказался дефектный чип, из-за которого Linux при простое зависал, и из-за этого разработка Gentoo полностью встала.

С тех пор чего только не происходило с Gentoo... Дэниел переключился на FreeBSD. Увиденное ему понравилось. Особенно система портов. И он вернулся в мир Linux. При поддержке других разработчиков, таких как Аким Готтингер, Gentoo снова вышел в путь и стремительно рванул вперед. Система управления пакетами подверглась полной переработке и получила название Portage. С тех пор год за годом Gentoo пребывает в активной разработке, с постоянным добавлением массы новых возможностей. Команды добровольцев помогают удерживать Gentoo на самом переднем крае, одновременно обеспечивая максимальную безопасность и стабильность.

Модель разработки Gentoo дополнилась полностью проектно-ориентированным подходом, где каждый проект разрабатывается независимо, но в сотрудничестве с другими проектами Gentoo. Регулярные совещания ведущих («менеджеров проектов») поддерживают быстрый темп разработки. Чтобы позаботиться о финансовых вопросах, обеспечить юридическую защиту и удержать разработку Gentoo в рамках общественного договора, создан фонд Gentoo.

В апреле 2004 Дэниел решил *оставить* пост разработчика Gentoo. Мы все весьма благодарны Дэниелу за его вклад в Gentoo, и желаем ему всего наилучшего.

Сейчас Gentoo продолжает расти, развиваться и улучшаться: появляются новые проекты, присоединяются новые разработчики, каждый день добавляются свежие пакеты. Но, безусловно, наивысшая ценность Gentoo — это сообщество разработчиков и пользователей Gentoo!

Философия Gentoo

Ссылка на оригинал: <http://gentoo.ru/philosophy>

С версии: 1.0

Автор: *Daniel Robbins* - изначальный разработчик Gentoo

Перевод: *Логинов Игорь*

Я начал работу над Gentoo, потому как не мог найти дистрибутива, который бы меня устраивал. Во всех Linux-дистрибутивах я заметил одну неприятную деталь - инсталляционные утилиты, управляющие установкой ПО в системе, предназначались для упрощения работы, однако на практике лишь мешали администрированию системы.

Поэтому я и создал Gentoo с её собственной системой портэжей (т.н. Portages), более совершенной, чем все системы управления ПО, существовавшие до этого. Система портэжей отличается большой гибкостью и даёт возможность пользователю успешно использовать её для решения конкретных задач независимо от поставленных целей.

Например, если вы хотите посмотреть подробности сборки пакета, вы можете посмотреть файл **ebuild**.

ebuild - сборочный файл (вероятно, авторское сокращение от emerge build file)

и получить всю необходимую информацию оттуда. Чтобы настроить параметры установки, можно воспользоваться переменной **USE**. Чтобы добавить новый пакет, можно просто создать новый файл **ebuild** для дерева портэжей. Вы также можете установить требуемый пакет командой **emerge** - система автоматически проследит за установкой всех необходимых пакетов, от которых зависит нужное вам приложение.

Linux-сообщество оценило новую систему портэжей, и Gentoo начал стремительно развиваться. Дистрибутив получил известность как "**дистрибутив исходников**", однако сутью Gentoo является вовсе не принцип "from source" (из исходников). Конечно, это важный, ключевой принцип дистрибутива, но не основной. Основным здесь является технология, позволяющая пользователю делать с системой всё что угодно, без каких либо ограничений.

Работая над этой технологией, я представлял себе произвольного пользователя ОС Linux. Что он (или она) хотел бы сделать с системой?

Философия Gentoo - дистрибутив, дающий возможность делать с системой всё что угодно.

Камнем преткновения для появления Gentoo было отсутствие простого метода автоматической установки ПО из исходников, ориентированного на пользователей, с подробными инструкциями. На данный момент мы решили эту задачу практически полностью. Частично реализована поддержка прекомпилированных пакетов, хотя, с другой стороны, с момента своего появления система портэжей Gentoo **поддерживала** установку бинарников.

Реализация поддержки бинарных пакетов очень важна, т.к. большое количество ПО распространяется именно в бинарниках (например, в виде пакетов RPM - прим. переводчика). Если бы Gentoo не поддерживал бинарные пакеты, мы бы не смогли бы назвать его удобным дистрибутивом, позволяющим пользователю делать с системой всё, что ему придёт в голову. Впрочем, помимо "философских" причин для поддержки бинарников есть много причин практических.

Философию Gentoo можно сформулировать в нескольких предложениях:

- Каждый линуксоид использует компьютер для решения своих конкретных задач.
- Главной целью Gentoo является предоставление ПО, позволяющего пользователю выполнять свои задачи эффективно.
- Утилиты Gentoo должны быть удобны в использовании, должны позволить пользователю оценить по достоинству все преимущества открытого ПО вообще и Linux в частности. Такое возможно лишь при наличии огромной базы открытого ПО, отражающей все потребности современных

пользователей, содержащей тысячи разнообразных программ, по возможности удобных.

- Если некая утилита вынуждают пользователя выполнять какие-то нестандартные действия для совершения стандартных операций - она скорее мешает работе, чем помогает. Каждый встречал подобные утилиты, неудобные в работе, лишённые ряда очевидных функций. Такое ПО противоречит философии Gentoo.

Другими словами, философия Gentoo также заключается в разработке корректного ПО. Когда утилита выполняет свои и только свои обязанности, но выполняет их в полной мере, вы можете забыть о её присутствии (что, впрочем, соответствует философии Unix вообще как таковой - прим. переводчика). Она не помешает, не попытается взаимодействовать с вами, пока вы сами не захотите этого.

В настоящее время в перспективы Gentoo входит поддержка и создание ПО, близкого к идеалу - мощных и простых в использовании утилит, способных обеспечить выполнение множества различных задач. Разве не прекрасно иметь возможности найти ПО, делающее именно то, что вы от него хотите? Разве это не вызывает чувство эстетического наслаждения? Наша задача - донести это чувство до как можно большего числа людей.

Советы пользователям Gentoo

Ссылка на оригинал: <http://ylsoftware.com/?action=news&na=viewfull&news=130>

С версии: 1.0

Автор: [MooSE](#)

Дата: 22.11.2006

Gentoo - один из самых трудных для изучения дистрибутивов. Хотя многие старые пользователи скажут что огромное сообщество и обширная документация могут здорово помочь новичкам. В этой статье приведён ряд подсказок, которые помогут новым пользователям быстрее освоить Gentoo.

Для начала приготовьтесь прочитать огромное количество документации. Вы не сможете установить Gentoo не изучив внимательно [Handbook](#) ([Главное руководство Gentoo- см. ниже: Настольная книга пользователя](#)).

Gentoo имеет графический инсталлятор, но новичкам не стоит его использовать. Установка Gentoo "традиционным" способом вынуждает читать документацию, и способствует лучшему пониманию системы. Так же инсталлятор Gentoo не так хорош при установке и настройке драйверов как инсталляторы других дистрибутивов.

Handbook содержит основную информацию о Gentoo. Дополнительная информация (например о настройке X и ALSA) может быть найдена на странице документации или на Gentoo Wiki.

Вики (Wiki) (от гавайского: быстрый) — гипертекстовая среда (обычно Веб-сайт) для сбора и структуризации письменных сведений (в программировании: документации, советов, часто задаваемых вопросов(FAQ), мини-руководств(HOWTO)).

Характеризуется следующими признаками:

- Множество авторов. Некоторые вики могут править все посетители.
- Возможность многократно править текст посредством самой вики-среды (вебсайта), без применения особых приспособлений на стороне редактора.
- Проявление изменений сразу после их внесения.
- Разделение информации на страницы, где у каждой своё название.
- Особый язык разметки, позволяющий легко и быстро размечать в тексте структурные элементы, форматирование, гиперссылки и т. п.
- Учёт изменений (учёт версий) текста и возможность отката к ранней версии.

Для создания вики-среды необходимо особое ПО — движок вики. Это частный вид системы управления сайтом, довольно простой в своём устройстве и функциональности, ибо почти все действия по структуризации и обработке сведений делаются пользователями вручную. Главная особенность — посетители сайта могут сами редактировать содержимое этих страниц и создавать новые. Wiki сделаны так, чтобы редактировать страницы было просто; для редактирования ничего кроме программы веб-браузера не нужно.

ALSA, или Advanced Linux Sound Architecture — продвинутая Звуковая Архитектура Linux, обеспечивает аудио- и MIDI- (Musical Instrument Digital Interface — Цифровой Интерфейс Музыкальных Инструментов) функциональность в операционной системе Linux. ALSA является основной звуковой подсистемой в ядрах 2.6, заменившей собой OSS (Open Sound System -

Открытая Звуковая Система), которая использовалась в ядрах 2.4

Х-сервер (или просто Иксы)- обычного пользователя могут испугать мысли о необходимости печатать в командной строке. Почему бы ему не прокладывать свой путь на территории свободы выбора Gentoo (и Linux вообще), просто «водя мышкой»? Ну, конечно, вы сможете это сделать :-). В Linux предлагается широкое разнообразие пользовательских интерфейсов и сред, которые можно установить поверх установленной у вас системы.

Это один из самых больших сюрпризов, с которым сталкиваются новички: графический интерфейс пользователя — это всего лишь прикладная программа, выполняемая в системе. Он не является частью ядра Linux или каких-либо других компонентов системы. Это — мощный инструмент, позволяющий полностью задействовать графические возможности вашего компьютера.

Так как стандартизация очень важна, для организации прорисовки и перемещения окон по экрану, взаимодействия пользователя с мышью и клавиатурой, а также других важных базовых аспектов, был создан стандарт под названием X Window System, сокращённо X11 или просто X. Он используется в Unix, Linux и других Unix-подобных операционных системах во всем мире.

Приложение, которое дает пользователям Linux возможность запускать графические интерфейсы пользователя и использует стандарт X11, называется Xorg-X11, ответвление проекта XFree86. Проектом XFree86 было решено использовать лицензию, возможно, несовместимую с лицензией GPL, поэтому рекомендуется использовать Xorg. Пакет XFree86 больше не поддерживается в официальном дереве портежей.

Проект X.org

Проектом X.org создана и поддерживается бесплатно распространяемая реализация системы X11 с открытым исходным кодом. Это открытая инфраструктура рабочей станции, основанная на X11.

Xorg обеспечивает интерфейс между устройствами вашего компьютера и нужными вам графическими программами. Кроме того, Xorg является полностью сетевым, то есть вы можете запускать программу на одной системе, а отображать ее — на другой.

Если у вас возник специфический вопрос, не освещённый в документации - попробуйте спросить на форуме или в [IRC](#).

Основы Portage

Одно из фундаментальных отличий Gentoo от других дистрибутивов - его система управления пакетами, названная **Portage**, ориентированная в первую очередь на установку программ из исходных текстов. Это позволяет иметь множество настроек и точек контроля для одного пакета там, где пользователи бинарных дистрибутивов имеют несколько пакетов с разными опциями и зависимостями (фактически это тоже самое, только управление различными опциями сборки не такое гибкое). С Portage вы можете установить так называемые **USE-флаги** для настройки опций одного пакета или даже всей системы.

Например если вы хотите включить поддержку KDE и выключить поддержку GNOME вам нужно добавить к строке USE в файле /etc/make.conf флаги **"kde -gnome"**. Для получения более подробной информации о USE-флагах смотрите [Gentoo Handbook](#).

Для работы с Portage используется команда **emerge**. Например для установки пакета `moos` достаточно набрать команду: **emerge moo**. Portage проверит зависимости, скачает все необходимые архивы с исходниками, соберёт пакет и установит его. Вся необходимая информация сохранена в базе, состоящей из `ebuild`'ов, каждый из которых представляет собой специальный скрипт. Все **ebuild**'ы находятся в папке `/usr/portage`.

Управление настройками Portage

Первое правило Gentoo - нужно внимательно следить за флагами **USE**. Почему при установке Midnight Commander устанавливается X.Org - потому что у вас разрешён флаг X. Почему нет KPDF в KDE? - Потому что KDE был собран без флага pdf (не спрашивайте почему всё же собрался KGhostScript).

Всегда устанавливайте новые пакеты командой **emerge --ask --verbose** (или **emerge -av**) для просмотра USE-флагов перед началом компиляции. Установите пакет **gentoolkit** (**emerge gentoolkit**) - среди множества полезных вещей он содержит утилиту **euse**, которая существенно упрощает управление флагами USE*. Выполните **euse -i flag** для получения информации о флаге "flag". Так же просмотрите man-страницу `euse` (не волнуйтесь. она короткая :-)).

Если вы хотите изменить USE-флаги только для одного пакета не пытайтесь собирать его командой **USE="some flags" emerge moo**. Это конечно сработает, но при обновлении системы пакет будет пересобран со старыми флагами. Лучше прописывайте USE-флаги в `/etc/portage/package.use`, для этого выполните **echo "category/moo some flags" >> /etc/portage/package.use** перед установкой пакета.

Тоже относится и к **ACCEPT_KEYWORDS** - используйте `/etc/portage/package.keywords`. В Portage 2.1 вы можете использовать папки `package.use` и `package.keywords` вместо файлов для более простого управления.

Когда нужно использовать оверлей?

Оверлей - набор `ebuild`'ов, которые вы хотите добавить к официальной базе данных (дереву) Portage

Иногда возникает необходимость установить пакет, отсутствующий в Portage. Например вы можете захотеть установить пакет, удалённый из дерева Portage. Тогда вам нужно взять его `ebuild` из архива Portage CVS. Или например вы захотите установить новый пакет, взяв **ebuild** в bugzilla. Возможна ситуация, когда вы просто захотите немного подправить официальный **ebuild**.

В любом случае вам нужно в такой ситуации поместить ваш **ebuild** в локальный оверлей. Оверлей - набор **ebuild**'ов, которые вы хотите добавить к официальной базе данных (дереву) Portage.

* Также имеется утилита: **ufed** (установка: **emerge ufed**) для более дружелюбного просмотра USE флагов

Не храните свои ebuild'ы внутри официального дерева (/usr/portage), поскольку при обновлении они будут удалены.

Создайте каталог для оверлея, и создайте в нём структуру подкаталогов, аналогичную `/usr/portage`. Поместите туда ваш **ebuild**. Добавьте **PORTDIR_OVERLAY="/path/to/local/overlay"** в ваш файл `/etc/make.conf`.

Обычное место для оверлеев - `/usr/local/overlays`. Таким образом местом для вашего **ebuild**'а будет `/usr/local/overlays/local`. Что же ещё может находиться в `/usr/local/overlays`.

Во многих случаях, когда вы ищите пакеты, не включённые в официальное дерево, вы находите не только **ebuild**'ы, но и оверлеи. Например один из самых популярных оверлеев **xgl-coffee** содержит всё, что нужно для работы XGL. Вы можете установить его в `/usr/local/overlays/xgl-coffee` что не смешивать с другими оверлеями и обновлять его через SVN.

Вы можете найти подробную информацию относительно использования оверлеев на [Gentoo Wiki](#). Там же вы найдёте список неофициальных оверлеев и описание процесса синхронизации с ними.

Как часто нужно обновляться?

Вам нет смысла обновляться чаще чем раз в сутки. Можно даже реже. Вы можете следить за вашими любимыми пакетами периодически посещая сайты [packages.gentoo.org](#) или [gentoo-portage.com](#) (или подписавшись на RSS).

Сохраните мир чистым !

Каждый раз, когда вы выполняете **emerge some-package**, имя пакета записывается в файл `/var/lib/portage/world`. Когда вы обновляете систему командой **emerge --update --deep world** - используется список из этого файла. Поэтому этот файл должен быть по возможности короче. В нём должны быть только необходимые пакеты.

Если какой-то пакет является всего лишь зависимостью другого - не стоит держать его в файле world.

Например, если вы устанавливаете пакет, использующий библиотеку wxWidgets, то эта библиотека устанавливается как зависимость и не добавляется в файл world. Если вы деинсталлируете все пакеты использующие эту библиотеку - она перестанет обновляться при выполнении **emerge --update (...) world**, и будет удалены после вызова **emerge --depclean**. Единственный случай когда стоит добавить wxWidgets в world - если вы разрабатываете приложения на основе этой библиотеки.

Существует скрипт, помогающий очистить world от всего лишнего. Для установки пакета без занесения его в world используйте опцию **--oneshot**. Эта опция так же будет хорошим решением при временной установке какого-либо пакета.

Деинсталляция пакетов

Все ваши действия с emerge записываются в файл **/var/log/emerge.log**. Если вы хотите знать какой пакет собирается в настоящее время - выполните **tail /var/log/emerge.log**. Утилита **genlop (emerge genlop)** позволяет извлечь множество различной информации из этого файла. Например genlop -с покажет вам какой пакет собирается в данный момент и попытается предсказать сколько времени осталось до конца сборки.

Удаление временных файлов

Когда сборка пакета завершается с ошибкой - остаётся каталог с временными файлами в **/var/tmp/portage**. Если вы устраняете проблему и пакет нормально собирается - файлы удаляются. Если же нет - вы можете очистить каталог **/var/tmp/portage** вручную.

После обновления вы можете удалить устаревшие файлы из **/usr/portage/distfiles**. Для этого воспользуйтесь утилитой **eclean** из пакета **gentoolkit**. Она поможет удалить ненужные файлы. Эта утилита может работать как с исходниками, так и с бинарными пакетами. (просто вызовите её как **eclean-dist** или **eclean-pkg** соответственно). Прочитайте страницу man для получения полного списка опций.

Gentoo предоставляет вам огромные возможности по настройке вашей системы. Автор статьи надеется что после прочтения этих строк вы сможете воспользоваться этими возможностями и испытать чувство глубокого удовлетворения от возможности управления любыми аспектами системы. Вам понравится это ощущение.

Gentoo Linux: установка и настройка

Ссылка на оригинал: http://posix.ru/distro/gentoo_old/

Автор: Алексей Федорчук (alv@linux-online.ru)

С версии: 1.0

О Gentoo Linux

Gentoo Linux - один из типичных представителей т.н. **Source Based** дистрибутивов. То есть систем, основанных не на прекомпилированных бинарных пакетах, а собираемых непосредственно из исходных текстов. **Gentoo** относительно молод, но известностью сравнялся с такими ветеранами дистрибутирования, как **Red Hat** или **Debian**.

Gentoo создан в 2001 г. Дэниелом Роббинсом. Официальный сайт проекта – <http://www.gentoo.org>. (Русский: <http://gentoo.ru>)

С самого начала Gentoo являл собой синтез наиболее передовых технологий Linux с традиционными достижениями BSD-систем (в первую очередь - **FreeBSD**). В частности, в нем приняты сценарии начальной загрузки системы в BSD-стиле - большинство популярных дистрибутивов **Linux** наследуют схему инициализации системы, унаследованную от **Unix System V** (именно она была принята за основу Линусом Торвальдсом при создании своего ядра).

Главная особенность **Gentoo** также идейно унаследована от **FreeBSD**. В последней, как известно, только базовые компоненты (т.н. **Distribution**) устанавливаются из прекомпилированных пакетов. Для установки же пользовательских приложений предназначена т.н. система портов. Это - набор средств для автоматизированного получения из Сети (с мастер-сайтов разработчиков или их официальных зеркал) исходных текстов программ, их конфигурирования, компиляции, установки и регистрации в специальной базе данных. Конечно, и **FreeBSD** поставляется с наборами прекомпилированных пакетов (**packages**), однако роль их сугубо вспомогательная.

В **Gentoo** концепция портов (именуемых здесь портежами - **Portages**) проводится еще более последовательно, чем во **FreeBSD**. Поскольку здесь этим путем можно установить (или - обновить) не только прикладные программы, но и все базовые компоненты системы, вплоть до ядра, компилятора **gcc** и главной системной библиотеки **glibc**.

Система портежей **Gentoo** позволяет:

1. очень гибкое конфигурирование системы, с подбором только необходимых (и осознанно выбираемых) приложений;
2. открыть путь для оптимизации системы не только под различные архитектуры (а на данный момент дистрибутивом поддерживаются, кроме i86, также Sparc, PowerPC и Alpha), но и под конкретные типы процессоров;
3. средства генерации бинарных пакетов (аналог **packages FreeBSD**), которые

- могут быть установлены на целевую платформу, отличную от host-машины;
4. оптимизировать пакеты с учетом особенностей целевой (например, более слабой) машины.

Дополнительным плюсом дистрибутива **Gentoo** является его прекрасная документированность. На сайте проекта доступны [Gentoo Handbook](#) и множество более частных источников информации, ряд из которых переведен на русский язык.

Название дистрибутива (произносится как "джен-ту") может показаться несколько странным. Но объясняется просто - это один из видов пингуина (научное название *Rygoscelis parrua*), обитающий на Циркум-Антарктических островах (Фолклендских, Кергелен, Южная Георгия).

В специальной литературе можно прочитать, что джен-ту отличается небольшим размером и стремительностью. Именно эти его качества и воплощены в дистрибутиве **Gentoo**.

Получение дистрибутива

Все версии дистрибутива Gentoo доступны на основном сервере проекта и на ряде его зеркал. Скачивание - по протоколам http и ftp, разумеется, свободно и бесплатно. Конечно, за некоторую сумму можно заказать и CD-вариант, но в наших условиях это не актуально.

Процесс получения **Gentoo** имеет некоторые особенности, отличающие его от получения (приобретения) любого пакетного дистрибутива. Если для последнего достаточно скачать iso-образы CD-дисков и записать их на соответствующие носители, продуктивное использование Gentoo требует выхода в Интернет. Правда, не обязательно с той же машины, на которую он устанавливается - вполне допустима ситуация, когда необходимые компоненты получаются, например, по рабочему каналу, а сама система устанавливается дома.

Итак, первое, что необходимо для установки - скачать образ инсталляционного диска. Объемы образов - достаточно скромны, обычно около 200 Мбайт (в зависимости от версии). Образы эти являют собой **Live CD** - то есть фактически полноценную Linux-систему, запускаемую с компакт-диска. И включающую набор не только системных утилит, но и пользовательских приложений.

Образ инсталляционного диска обычно содержит все необходимое для установки. Однако в некоторых версиях его оказывается недостаточно - потребуется еще три архива (*.tar.bz2), содержащие (в прекомпилированном виде) компоненты базовой системы. Первый из них (файл **stage1-***, объемом около 10 Мбайт) специфичен для архитектуры (i86, PowerPC, Sparc и т.д.). Два других тарбалла (**stage2-*** и **stage3-***, около 50 и 80 Мбайт, соответственно), кроме этого, для PC-архитектуры представлены в нескольких вариантах, оптимизированных под конкретные процессоры: от i486 до Pentium-4 (включая и Athlon). Ясно, что скачиванию подлежат только тарбаллы, соответствующие реалиям пользователя.

Далее, для установки пре-релиза потребуется скачать архив системы портежей - один из файлов вида **portage-200XXXXX.tar.bz2**, расположенных в каталоге **~/gentoo/snapshots** (в стабильной версии текущий архив портежей размещен на установочном диске). Как легко понять, цифры в имени файла соответствуют дате создания архива (а обновляется система портежей чуть ли не ежедневно), и лучше выбирать последний по времени.

И наконец, при установке на машину без выхода в сеть неплохо заблаговременно скачать архивы исходных текстов базовых компонентов системы - об их составе я скажу несколько позже. И не обязательно с сайта разработчика: исходники всех портированных в **Gentoo** программ можно найти на сайте проекта в каталоге **~/gentoo/distfiles** (или **~/gentoo/gentoo-sources**, содержание их идентично). Наличие их очень упростит жизнь, однако, как будет показано ниже, обойтись без них можно.

А вот без чего никак не обойтись - это без одного из дополнительных носителей, на котором будут размещены скачанные компоненты. При установке на "чистую" машину ими могут быть: второй CD (при наличии двух приводов, например, пишущего и читающего) или Zip (как нетрудно подсчитать - желательно 250-мегабайтного объема, иначе их придется менять в процессе установки). Можно также временно подцепить дополнительный винчестер с уже существующим разделом и какой-либо файловой системой (из числа поддерживаемых) на нем. Ну и заблаговременно переписать на эти носители все требуемые файлы - тарбаллы **stage2-***, **stage3-*** и **portage-*** - обязательно, прочие - по возможности и желанию.

К слову сказать, на тот же носитель хорошо поместить и инструкцию по установке с сайта **Gentoo** (добавлю, и все прочие файлы документации - тоже, отнюдь не повредят). Или, по крайней мере, держать ее распечатку под рукой - зело способствует. И можно приступать к установке.

Помещать тарбаллы базовой системы и системы портежей, а также документацию, можно где угодно - хоть в корне файловой системы дополнительного накопителя. А вот под архивы исходников лучше создать на нем отдельный каталог, например, **src** - ниже будет ясно, почему.

Загрузка системы и первые шаги

Редкий инсталляционный диск Linux-дистрибутива ныне не является загрузочным. И **Gentoo** тут - не исключение: чтобы загрузиться с любого из вышепоименованных образов, достаточно поместить диск в привод, нажать на **Reset**, выставить соответствующие опции в **Setup BIOS** и подождать некоторое время, по возможности вникая в появляющиеся сообщения об имеющемся оборудовании (впрочем, все эти сведения можно будет получить и потом - командой **dmesg**).

Ожидание завершается выводом приглашения на ввод пользовательского имени (**login**), ответом на что служит волшебное слово **root**, а затем и пароля - на это отвечаем просто нажатием клавиши **Enter**. За этим появляется приглашение командной строки - и можно приступить к дальнейшим действиям.

Они не вполне тривиальны. Еще одна отличительная особенность дистрибутива **Gentoo**, о которой я не упомянул ранее - полное отсутствие какого-либо инсталлятора, все манипуляции по установке осуществляются прямыми директивами из командной строки и прямым редактированием конфигурационных файлов в текстовом редакторе. Что начинающего (да и не очень начинающего) пользователя может просто обескуражить.

Однако все не так страшно. Во-первых, в нашем распоряжении - вышеупомянутая подробная, буквально пошаговая, инструкция по установке. А во-вторых (и об этом также говорилось ранее), установочный диск **Gentoo** - не просто загрузочный, а так называемый Live CD. То есть просто-напросто загружающий полноценную рабочую Linux-среду с корневой файловой системой в оперативной памяти (способной монтировать в себя файловые системы любых устройств). И потому в процессе установки в руках пользователя - вся мощь ОС Linux, включая командную оболочку **bash** и 6 виртуальных консолей (авторизоваться в любой из них можно точно так же, как и в первой - вводом имени root без пароля). Ну а в третьих - загружаемое с CD Linux-ядро собрано с поддержкой абсолютно всех опций, которые могут потребоваться при установке (встроенной в ядро или модульной).

И потому первое действие после старта системы - загрузка всех необходимых модулей. Это будут модули поддержки:

- устройств хранения данных - SCSI-адаптеров, контроллеров IDE-RAID, Zip-приводов (atapi-floppy, в терминологии Linux), короче, всего, на что мы собираемся устанавливать систему или считывать устанавливаемые компоненты;
- файловых систем: это потребуется, только если устанавливаемые тарбаллы и архивы исходников размещены на разделе с файловой системой FATxx - поддержка всех родных (native) файловых систем Linux, включая XFS, встроена в установочное ядро);
- сетевого адаптера - если таковой, вместе с постоянным подключением к Сети, имеется;
- соединения rrr - это даст возможность использовать при установке модемное подключение к Сети.

При этом возможно, что многие из перечисленных устройств (в частности, сетевые карты) будут корректно определены при загрузке системы и соответствующие модули загрузятся автоматически, что проверяется командой **\$ modprobe имя_модуля**

В остальных случаях это придется сделать вручную, командой **\$ insmod имя_модуля**

Имена поддерживаемых модулей можно посмотреть (авторизовавшись в следующей виртуальной консоли) в соответствующих подкаталогах каталога **/lib/modules/2.6.XX.X/kernel/** - ведь наша корневая система пока находится в оперативной памяти.

Следующий шаг - настройка сетевого соединения (при постоянном подключении).

Если машина использует динамически выделяемый IP-адрес, назначаемый через DHCP-сервер (а в большинстве случаев это так и есть), делается это элементарно просто, командой

```
$ dhcpcd eth0
```

Все остальное происходит почти автоматически. При статическом IP-адресе потребуется ручная настройка, но на этом я останавливаться не буду (все вопросы конфигурирования сети очень подробно рассмотрены в руководстве).

При отсутствии постоянного подключения необходимо смонтировать носитель со скачанными ранее архивными файлами. Для этого в каталоге */mnt* создаем соответствующую ситуации точку монтирования, например,

```
$ mkdir /mnt/zip
```

для Zip-диска, или

```
$ mkdir /mnt/cd2
```

для второго CD (каталог для установочного CD - */mnt/cdrom*, - там уже имеется, и диск этот в него уже смонтирован). А теперь даем команду монтирования. Для zip-диска "из коробки" она будет выглядеть так:

```
$ mount /dev/hdd4 -t msdos /mnt/zip
```

где, кроме явного указания файловой системы, обращаю внимание, что файл устройства - именно */dev/hdd4*: при фабричной разметке на zip-дисках создается один, но четвертый, первичный раздел. И еще одна тонкость: вследствие особенностей реализации файловой системы *devfs* в дистрибутиве Gentoo, zip-диск должен находиться в приводе в момент старта системы - иначе соответствующий файл устройства просто не будет создан.

Аналогичны и действия по монтированию второго CD или дискового раздела. Единственно, что в этом случае может не потребоваться явного указания на тип файловой системы (хотя и вреда от него, конечно, не будет).

Завершив монтирование носителя установочных файлов, следует перво-наперво активизировать еще одну виртуальную консоль и вывести на нее [инструкцию по инсталляции](#) - дабы сверять с ней все дальнейшие шаги, как со Священным писанием:

```
$ lynx /mnt/mount_point/build.html
```

При наличии модемного подключения к Сети можно настроить и его - даже если все требуемые файлы, включая исходники базовой системы, размещены на смонтированном носителе, не исключено, что в процессе их синхронизации может потребоваться какой-либо специфичный для Gentoo патч, скачать который по модему - дело пары минут, а то и секунд. Хотя должен предупредить, что установить базовую систему чисто по модему (что вполне реально, скажем, для **FreeBSD**) - не то что невозможно, но, скажем так, весьма долго и накладно для кармана.

Для настройки модемного соединения проще всего использовать программу **wvdial**, находящуюся на Live CD (в каталоге */usr/bin*). Подгрузив модули

поддержки протокола **ppp** (общий, **ppp_generic**, и асинхронного соединения, **ppp_async**), запускаем программу
\$ wvdialconf /etc/wvdial.conf

Она автоматически определит порт, к которому подключен модем, определит его скорость и создаст строку инициализации, занеся все эти сведения в файл, указанный в качестве аргумента. В результате содержание его примет вид, подобный следующему:

```
[Dialer Defaults]  
Modem = /dev/ttyS0  
Baud = 115200  
Init1 = ATZ  
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0  
ISDN = 0  
Modem Type = Analog Modem
```

Пользователю остается только добавить туда строку, определяющую способ дозвона. По умолчанию он - тоновый, тогда как в наших условиях, скорее всего, потребуется импульсный дозвон:

Dial Command = ATDP

Остальное же - обычная информация: телефон провайдера

Phone =

имя пользователя и его пароль

Username =
Password =

Теперь, проследив, чтобы в файле /etc/hostname наличествовала единственная строка

localhost

подкорректируем содержимое файла /etc/resolv.conf: значение строки nameserver должно соответствовать реальному IP-адресу DNS-сервера провайдера. И теперь при необходимости для дозвона можно использовать команду

\$ wvdial

А больше на этапе установки нам ничего и не требуется - все остальное сделает система портежей.

Возникает вопрос: а каким образом редактировать все вышепоименованные конфигурационные файлы (и те файлы, которые потребуется редактировать в дальнейшем)? Этой цели в дистрибутиве Gentoo штатно служит текстовый редактор **nano** - очень простой в освоении и обращении, снабженный встроенной

справкой и обеспечивающий достаточные базовые функции редактирования. Нужно только не забывать запускать его в форме

\$ nano -w имя_файла

где опция -w отключает режим переноса строк - конфигурационные файлы на это, как правило, реагируют болезненно. Поскольку редактор понадобится нам за время установки неоднократно, резонно было бы сразу определить псевдоним:

\$ alias nano='nano -w'

благо, как уже говорилось, в нашем распоряжении полнофункциональная оболочка **bash**.

Установка - первый этап

Вот теперь подготовительные мероприятия закончены - можно начинать собственно установку, начальный этап которой происходит внутри корневой файловой системы Live CD. Первый шаг на этом этапе - создание дисковых разделов и файловых систем на них. Эти вопросы были предметом детального рассмотрения в моей статье, подготовленной к печати (Открытые системы, #11, 2002) и потому здесь я на них остановлюсь лишь вкратце.

Для первой операции можно использовать одну из программ - **fdisk**, **cfdisk**, **parted**, все они имеются на Live CD. А созданию файловых систем служат специфичные для них утилиты: **mkefs** для **ext2fs**, **mkfs.xfs** - для XFS, **mkswap** - для раздела подкачки, и так далее. Рекомендуемая в руководстве схема разбиения для одного диска (первого на первом IDE-канале) приведена в табл. 1.

Таблица 1. Рекомендуемая схема разбиения диска для установки Gentoo

Раздел	Размер	Тип	Устройство
Загрузочный	100 Мбайт	ext2fs или ext3fs	/dev/hda1
Раздел подкачки	128 Мбайт=<2*RAM=<2 Гбайт	Linux swap	/dev/hda2
Корневой раздел	1,5 Гбайт минимум	XFS, ext3	/dev/hda3

Разумеется, эта рекомендация - приближительна. Так, 1,5 Гбайт для корневого раздела - лишь абсолютно необходимый минимум. При интенсивном использовании системы портежей цифру эту нужно увеличить в два-три раза, а то и более. Или выделить файловую систему **/usr** в отдельный раздел. Кроме того, на мой взгляд, абсолютно необходимо создать отдельный раздел под файловую систему **/home** - для домашних каталогов пользователей. А в некоторых случаях целесообразно выделение и каталогов **/var** и **/tmp**.

Если под установку **Gentoo** предполагается задействовать более одного физического диска, можно прибегнуть к таким современным технологиям, как программные RAID-массивы или система управления логическими томами (LVM - Logical Volumes Manager). Рассмотрение их далеко выходит за рамки настоящей

статьи. Скажу только, что первая технология способствует повышению быстродействия дисковых операций или (иногда - и) сохранности данных. А система LVM позволяет достичь непревзойденной гибкости в управлении уже созданными и смонтированными файловыми системами (в частности, динамического перераспределения занимаемого ими дискового пространства). И, разумеется, на установочном Live CD содержится весь необходимый инструментарий для работы с программными RAID-массивами и системой LVM. Впрочем, нет препятствий и к использованию дисков, подключенных к аппаратному контроллеру IDE-RAID, хотя это потребует некоторых дополнительных действий (детали - в руководстве по установке).

Следующий шаг - монтирование созданных файловых систем в структуру корневой системы (напомню, она - в оперативной памяти). Но сначала - активизируем раздел подкачки:

```
$ swapon /dev/hda2
```

что будет способствовать быстродействию при всех дальнейших манипуляциях, особенно при не очень большом объеме ОЗУ.

Теперь необходимо создать точку монтирования для той файловой системы, которая в дальнейшем станет корневой:

```
$ mkdir /mnt/gentoo
```

и смонтировать в нее соответствующий раздел:

```
$ mount /dev/hda3 /mnt/gentoo
```

Если создавались отдельные разделы под каталоги /boot, /usr, /home - они тоже потребуют своих точек монтирования, теперь уже относительно каталога /mnt/gentoo:

```
$ mkdir /mnt/gentoo{/boot,/usr,/home}
```

и выполнить для них команды монтирования:

```
$ mount /dev/hda1 /mnt/gentoo/boot
```

и так далее. Теперь командой

```
$ cd /mnt/gentoo/
```

переходим в будущий корневой каталог Gentoo и вспоминаем наконец о наших архивных файлах. Разворачиваем тарбалл первой очереди, находящийся на инсталляционном CD:

```
$ tar -xvjp /mnt/cdrom/nocompress/stage1-x86-*.tar.bz2
```

Здесь и далее следует обратить внимание на опцию -p в командной строке: она необходима для сохранения оригинальных атрибутов доступа к файлам, извлекаемым из архива. По самой же команде разворачивается иерархия каталогов в дереве /mnt/gentoo (/mnt/gentoo/etc, /mnt/gentoo/proc и т.д.) и устанавливается некий минимум прекомпилированных программ, необходимых для дальнейшей инсталляции, в том числе и ftp-клиент wget. И действительно, при постоянном подключении к Сети более ничего не потребуется - можно переходить к подготовке смены корневого каталога. Однако тем, кто не может им

похвастаться, предварительно придется распаковать два оставшихся тарбалла, stage2* и stage3*.

Делается это точно так же, как и развертывание первого, только пути к соответствующим файлам нужно указать те, что соответствуют их реальному расположению, например:

```
$ tar -xvlpf /mnt/zip/stage2*.tar.bz2
```

и так далее. После этого мы имеем в каталоге **/mnt/gentoo** тот самый набор прекомпилированных библиотек, утилит и приложений, который объединяется понятием base Linux. И, соответственно, морально готовы к тому, чтобы сделать каталог **/mnt/gentoo** корневым.

Этап второй - подготовка к смене корня

Однако этот ответственный шаг требует некоторых подготовительных мероприятий - ведь после него все остальные ветви нынешнего корневого каталога станут недоступными. Правда, только с текущей, первой, консоли - во всех прочих сеансах корневым останется каталог в оперативной памяти.

Для начала, чтобы не делать двойную работу, скопируем настроенные нами конфигурационные файлы - ведь и они пока существуют только виртуально:

```
$ cp /etc/resolv.conf /mnt/gentoo/etc/resolv.conf  
$ cp /etc/hostname /mnt/gentoo/etc/hostname  
$ cp /etc/wvdial.conf /mnt/gentoo/etc/wvdial.conf
```

Теперь воспользуемся замечательным свойством современных ядер Linux - возможностью доступа к одной и той же файловой системе через разные точки монтирования. И смонтируем таким образом файловую систему процессов (она, как ясно из названия, представляет протекающие в системе процессы в виде файлов):

```
$ mount -o bind /proc /mnt/gentoo/proc
```

К слову сказать, таким же образом можно поступить и с тем носителем, на котором у нас размещены архивы системы портежей и исходников (если последние мы туда помещали). А можно, просто размонтировав его (доступ к ним из виртуального корня Live CD больше не обязателен), смонтировать заново - но уже относительно будущего нового корня:

```
$ mount /dev/hdd4 /mnt/gentoo/mnt
```

Ведь в нем развернута собственная корневая система, имеющая все положенные компоненты. Правда, в этом случае мы потеряем доступ к инструкции по установке - а она нам еще потребуется. Вообще говоря, обойтись с этим носителем можно самыми разными способами - например, не перемонтируя его, просто обращаться с других виртуальных консолей. Я опишу самый, на мой взгляд, удобный.

Не предпринимая никаких размонтировочных действий, переходим в каталог **/mnt/gentoo/usr** (таковой ведь у нас тоже имеется). И из него разворачиваем архив

портежей, точно так же, как мы это делали для основных тарбаллов:

```
$ tar -xvjp /mnt/zip/portage*.tar.bz2
```

Если исходников на вспомогательном носителе нет - можно оставить его в покое, нужды в нем больше не возникнет. Если же исходники имеются - создаем в новообразованном каталоге **/mnt/gentoo/usr/portage** подкаталог **distfiles** и копируем в него все наши архивы исходных текстов:

```
$ cp /mnt/zip/src/* /mnt/gentoo/usr/portage/distfiles
```

Теперь они находятся именно там, где их ожидает встретить система портежей, прежде чем обращаться к Сети (и выдавать сообщение об ошибке при отсутствии подключения). А дополнительный носитель оставляем смонтированным - чтобы подсматривать в документацию через другой виртуальный терминал.

Теперь все готово к волнующему моменту - смене корневого каталога. Итак, командой

```
$ chroot /mnt/gentoo /bin/bash
```

объявляем каталог **/mnt/gentoo** корнем, одновременно запуская новый экземпляр командной оболочки (уже из нового источника). Теперь пересчитываем пути к системным библиотекам

```
$ env-update
```

ответом на что будет сообщение

```
Regenerating /etc/ld.so.cache...
```

И устанавливаем для нового экземпляра оболочки переменные окружения:

```
$ source /etc/profile
```

Все, мы - внутри системы Gentoo Linux, причем уже очень даже реальной, не имеющей никакого отношения к виртуальной системе, загруженной с Live CD. И далее все пути к файлам в абсолютной форме (типа **/usr/portage**) будут указываться с учетом того, что отныне наш / - это не что иное, как **/mnt/gentoo**.

Третий этап - внутри Gentoo

Итак, мы уже как бы установили операционную систему **Gentoo** Linux (без перезагрузки, прошу обратить внимание). И вот тут-то и начинается самый охмурежь - рядом планомерных и последовательных действий мы можем после перезагрузки (ну куда же без нее) получить не просто полнофункциональную и аккуратно настроенную, но, не побоюсь громких слов, - идеально оптимизированную систему.

Требуемые к тому действия описываются в предположении: а) установки всех трех тарбаллов и системы портежей, б) помещения в каталог **/usr/portage/distfiles** (уже относительно нового корня - **/mnt/gentoo**) некоторого набора минимально необходимых исходников, и в) хоть какого-то соединения с Интернетом (на крайний случай). Без последнего можно, как я неоднократно говорил, и обойтись - но тут уж потребуются ювелирная точность в выборе

предварительно скачанных (и ныне помещенных в **/usr/portage/distfiles**) архивов исходников.

В чем смысл этих действий? Да ни много, ни мало, как в полной пересборке базовой системы (те, кто имел дело с FreeBSD, неизбежно вспомнят тут **make world**) и доустановки необходимых компонентов. Поскольку это - не **apt-get**, и не установка прекомпилированных пакетов, все требуемое извлекается непосредственно из исходников. И потому первое, что нам потребуется - настройка условий компиляции.

Глобальные параметры сборки посредством системы портежей описываются в файле **/etc/make.conf**. Вызываем его в редакторе **nano** (не забыв про опцию **-w**, если не решили этот вопрос разом - определением псевдонима) и начинаем действовать.

Первое, что бросается в глаза после открытия вышеупомянутого файла - переменная **USE**. Смысл ее - в определении условий предкомпиляционного конфигурирования исходников. То есть, она эквивалента опциям **--enable/disable** или **--with/without** скрипта **./configure**, с исполнения которого начинается ручная сборка любой программы. И, задав значения этой переменной (несложно догадаться, что указав

USE="gpm"

мы включаем поддержку службы консольной мыши во всех собираемых через портежи программах (разумеется, тех, где это имеет физический смысл). А задав значение

USE="-gpm"

таковую отключаем раз и навсегда (на самом деле - не совсем навсегда, но это тема отдельной беседы).

Список включенных по умолчанию значений переменной **USE** можно подсмотреть в файле **/usr/portage/profiles/default-x86-XX/make.defaults**. А полный (и логично классифицированный) список доступных значений можно найти в файле документации - **use-howto.html**. Мы ведь списали его на дополнительный носитель? Да? - тогда можем вызвать с очередной виртуальной консоли...

Следующая переменная файла **/etc/make.conf** - **CHOST='i686-pc-linux-gnu'**

Она указывает на архитектуру машины, на которой будут собираться программы. И ее, при необходимости, нужно будет привести в соответствие с реалиями оной. Правда, ныне такая необходимость возникнет редко - разве что поменять первую шестерку на пятерку в случае **Pentium**-просто...

А вот следующая строка,
CFLAGS=""

очень важна. Ею задаются именно те флаги оптимизации, передаваемые компилятору для оптимизации под конкретный процессор. Вдаваться в детали

опять же не буду - и это тема особого, и очень подробного, разговора. Достаточно помнить, что флаг **-O3** (а именно он указан здесь по умолчанию) устанавливает самый высокий уровень оптимизации, а оба флага **-mcpu=*** и **-march=*** указывают, под какой именно процессор оптимизировать следует. Различие между ними - в том, что программа, собранная с флагом **-mcpu=i686**, будучи оптимизирована под класс процессоров **Pentium Pro** и выше (а с формальной точки зрения, и **P-II**, и **P-III**, и даже **P-4** принадлежат к нему), теоретически способна запускаться и на **Pentium-100**, тогда как флаг **-march=athlon-xp** потребует именно того процессора, который указан - даже на просто **Athlon'e** скомпилированная с ним программа работать не будет.

Смысл переменной

CXXFLAGS=

аналогичен - только для программ, написанных не на Си, а на Си++. И ее имеет смысл определить просто, через переменную **CFLAGS=**

CXXFLAGS="\${CFLAGS}"

как это и сделано в **example**-файле. Остальные строки которого определяют условия скачивания (в частности, **ftp**-клиента по умолчанию). Менять их на данном этапе - оснований не вижу, кроме разве что идиосинкразии к **wget**, используемому по умолчанию.

Определившись с условиями, можно начинать перестройку. Для этого меняем каталог

\$ cd /usr/portage

и запускаем сценарий

\$ scripts/bootstrap.sh

Он производит сборку пакетов **binutils**, **gcc**, **gettext** и **glibc** (замещая установленные из тарбаллов прекомпилированные их варианты), а после сборки **glibc** - пересборку **binutils**, **gcc** и **gettext**. Достигая, таким образом, их идеальной оптимизации... Хотя и ценой временных затрат - процесс даже на мощной машине вполне может занять часок-другой.

Теперь можно полюбоваться на список пакетов, установленных в нашей системе - ибо это и есть тот самый base Linux с очень небольшими коррективами. Для этого даем команду

\$ emerge -p system

Если есть желание, время, а главное - полный набор исходников в **/usr/portage/distfiles**, можно завершить пересборку базовой системы командой

\$ emerge system

Здесь без постоянного коннекта с очень большой вероятностью рано или поздно последует сообщение об ошибке - о невозможности получить пакет имя_рек с такого-то адреса. Если имеется модем, к подбору исходников мы подошли ответственно, - не беда: активизируем еще одну виртуальную консоль (а на ней будет жить наша первозданная Live-система с CD) и дозваниваемся до

провайдера через программу **wvdial** (а она у нас пока существует только в той самой виртуальной Live-системе), а потом повторяем последнюю команду. И, скорее всего, дело было в каком-нибудь исконно Gentoo'ском патчике объемом в пару килобайт, так что процедура эта нас не разорит. Если же обуюет приступ жадности, можно просто продолжить, а к emerge system вернуться в другой раз - после полного зеркалирования каталога **distfiles** с сайта **Gentoo**, и за казенный счет, разумеется :-)

Этап четвертый: в недра ядра

На предыдущем этапе были перестроены все компоненты базовой системы, кроме одного - зато самого главного: ядра системы. Перекомпиляция ядра - процедура, рано или поздно неизбежная для пользователя любого дистрибутива Linux, даже самого user-ориентированного. Перед пользователем же Source Based дистрибутива она встает с неотвратимостью рока уже при установке системы.

Gentoo - опять не составляет исключения. Хотя в заключительных разделах я расскажу, как при крайней необходимости можно обойти этот этап, однако по хорошему к тому нет никаких причин. И потому, хотя тема пересборки ядра - совершенно отдельная, и нам придется еще не раз к ней возвращаться, применительно случаю постараюсь осветить ее здесь в минимально возможном объеме.

Чтобы пересобрать ядро, для начала нужно иметь его исходники. Ни на инсталляционном диске, ни в составе тарбаллов **stage2-3** мы их не найдем. Что делать?

При подключении к Сети - все просто: к нашим услугам система портежей. Отправляемся в каталог **/usr/portage**, просматриваем содержимое подкаталога **sys-kernel** - и остается только дивиться свалившемуся на голову богатству: даже отметая классово чуждые портежи ядер для Sparc, PPC, Alpha, на выбор предоставляется чуть не дюжина вариантов. Подчеркну, речь идет не о прекомпилированных ядрах, а именно о вариантах исходников, снабженных патчами, обеспечивающими дополнительные опции относительно ядра перевозданного.

Разумеется, и последнее, каноническое ядро от Линуса Торвальдса - тоже доступно (под известным именем **vanilla-sources**) для тех, кто хочет быть святее папы римского. Далее - фирменная коронка, **gentoo-sources**, это - лучший выбор с точки зрения функциональности). Ну и ac-sources - ядро от Алана Кокса, в патчах которого обычно воплощены новейшие достижения ядреной науки и техники.

Есть и более специфичные варианты: **usermode-sources** - ядро для запуска Linux в Linux, в т.н. пользовательской моде (предназначается обычно для тестирования); **openmosix-sources** - ядро с поддержкой одноименной (**openMosix**) кластерной технологии, и т.д..

Так что в этих условиях главное - определиться с выбором. А потом - простая

команда

\$ emerge sys-kernel/name-sources

некоторый период ожидания, пока скачиваются сами исходники ядра и все необходимые патчи, еще чуть-чуть времени на развертывание архивов - и вперед, в каталог **/usr/src/linux** для конфигурирования.

При наличии предусмотрительно скачанных исходников ядра, наиболее явно необходимых патчей и модемного подключения - сложнее, но не намного. Выполняем ту же команду. Если повезет - все по прежней схеме, но предупреждаю, шансов на это мало: почти наверняка при предварительном скачивании какой-то Gentoo-специфичный патч будет пропущен (учтем также наше пока еще очень ограниченное знакомство с системой портежей вообще - в дальнейшем такой ситуации можно и избежать). Но не беда: получив сообщение об ошибке, соединяемся с Сетью и либо скачиваем недостающий патч вручную (хоть через браузер **lynx** - полный адрес к недостающему компоненту будет выведен), либо (и это, товарищи, правильнее) запускаем процедуру по новой. Разорвав соединение по завершении установки исходников ядра.

И не нужно бояться, что это больно ударит по карману: при соединении по линии, способной вынести протокол V90, скачивание большинства патчей занимает минуты. И **wget** при обрыве коннекта способен к докачке - так что непроизводительные расходы сведены к минимуму.

Хуже, если нет ни полного комплекта исходников, ни модема. И это не смертельно, но придется кое-чем поступиться (надеюсь, не принципами). То есть: просто тупо распаковываем архив исходников ядра в каталог **/usr/src**, распаковываем и накладываем минимально необходимые патчи, далее - обычным порядком, как описано ниже. Можно и просто ограничиться каноническим ядром Линуса. Недостаток - кроме возможных потерь некоторых функций, эти действия не будут зафиксированы в базе данных портежей (имеющей место пребывания в каталоге **/var/db/pkg**), то есть система портежей о них как-бы и не узнает (на самом деле - не совсем, но все равно - этот способ может рассматриваться как временное решение).

И еще одно временное решение - вообще отказаться от пересборки ядра. А как при этом получить возможность загрузить систему - я расскажу под занавес (если вы к тому времени сами не догадаетесь).

В любом случае, кроме последнего, исходники ядра мы получили, так что можно отправляться в вышеупомянутый каталог **/usr/src/linux** (или **/usr/src/linux-номер_версии-имя_варианта**, без разницы, - первый лишь символическая на него ссылка). И давать команду

\$ make menuconfig

которая сгенерирует меню настройки опций ядра. Любители отвечать черному экрану могут выбрать и **make config** - но я к ним не принадлежу.

Сам процесс конфигурирования описывать не буду - это совсем другая история. Отмечу только обязательные опции. Первая из них - поддержка

экспериментальных и разрабатываемых опций: большинство продвинутых возможностей ядра Linux описываются именно этими терминами (что не мешает им функционировать вполне справно). В секции файловых систем необходимо отметить поддержку файловых систем процессов (**procfs**), файловой системы виртуальной памяти (Virtual Memory Filesystem) и файловой системы устройств (**devfs**). И, конечно же, тех файловых систем, которые были размещены на созданных разделах (например, XFS или **ext3fs** - про поддержку **ext2fs**, надеюсь, не забудет ни один линуксоид). Ну и учесть поддержку требуемых устройств - сами знаете, что у вас вкручено в машину.

Закончив с конфигурированием, выходим из системы меню (не забыв сохранить изменения) и даем последовательность команд:

```
$ make dep
$ make clean
$ make bzImage
$ make modules
$ make modules_install
```

Разумеется, не обязательно последовательно, можно и одной строкой, разделяя символом ; (или, правильнее, &&). А то и вообще в форме, рекомендованной руководством:

```
$ make dep && make clean bzImage modules modules_install
```

Теперь остается только скопировать образовавшийся образ ядра в загрузочный каталог

```
$ cp /usr/src/linux/arch/i386/boot/bzImage /boot
```

И ядерный вопрос можно считать закрытым. Хотя остается еще вопрос, косвенным образом с ним связанный - программы поддержки файловых систем. Их, в соответствии с выбором и предпочтениями, устанавливаем через систему портежей, например, команда

```
$ emerge sys-apps/xfspgms
```

установит все необходимое для работы с файловой системой **XFS**. Напомню, что архив исходников пакета **xfspgms** должен иметься в каталоге **distfiles**.

Попутно через систему портежей можно установить и другие программы, однако я пока ограничился бы только тем, что облегчит завершение конфигурирования, а именно - службой консольной мыши:

```
$ emerge sys-libs/gpm
```

А теперь активизируем ее. Если мы имеем наиболее типичного ныне грызуна с колесиком и под разъем PS/2, это будет выглядеть так:

```
$ gpm -t imps2 -m /dev/psaux
```

И на следующих этапах гораздо проще будет выделять нужные фрагменты и вставлять их щелчком средней клавиши, нежели набрать вручную.

Финал: постинсталляционное конфигурирование

Собственно установка на этом закончена. Осталось довести систему до идеала не

только изнутри, но и снаружи. И первый шаг здесь - настройка часового пояса: для Source Based дистрибутивов это не последнее дело (утилита управления компиляцией **make** обращает внимание на даты создания файлов). Реализуем просто: созданием символической ссылки

```
$ ln -sf /usr/share/zoneinfo/path/timezonefile /etc/localtime
```

Рискну предположить, что компонент пути path/timezonefile для большинства моих читателей окажется в диапазоне от Europe/Kaliningrad до Asia/Kamchatka (наличие Анадыря в нашей стране почему-то упорно игнорируется - но, слава богу, хоть о Петропавловске-Камчатском знают). Если системное время в BIOS установлено на не время по Гринвичу, дополнительно потребуется внести коррективы в файл /etc/rc.conf.

Теперь увековечим монтирование новых файловых систем. Как? - очень просто, лобовым редактированием ответственного за сие файла **/etc/fstab**. Вызываем его:

```
$ nano -w /etc/fstab
```

И видим, если исключить комментарии, следующую картину:

/dev/BOOT	/boot	ext2	noauto,noatime	1 1
/dev/ROOT	/	xfs	noatime	0 0
/dev/SWAP	none	swap	sw	0 0
/dev/cdroms/cdrom0	/mnt/cdrom	iso9660	noauto,ro	0 0
proc	/proc	proc	defaults	0 0
#tmpfs	/dev/shm	tmpfs	defaults	0 0

Если мы следовали рекомендациям инструкции, остается только заменить символы верхнего регистра конкретными реалиями, как то: **/dev/hda1**, **/dev/hda3**, **/dev/hda2**. Или, что будет более корректно, **/dev/discs/disc0/part1** и т.д., сняв при этом комментарий со строки **#tmpfs**. Если же использовалась какая-либо иная схема разбиения - можно в одной из ранее активизированных консолей с помощью команды **mount** без аргументов подсмотреть, какой раздел с какой файловой системой куда монтировался. И просто перетащить мышью (не зря же мы ее запускали) соответствующие значения в наш **/etc/fstab**. Ну и всякого рода коррективы (типа монтирования FAT-разделов, флоппи-дисков, zip-накопителей и еще чего требуется) также никто не запрещает.

Внимательный читатель обратил внимание - в строке **/dev/BOOT** по умолчанию стоит значение **noauto**, предписывающее не монтировать соответствующий раздел автоматически при старте системы. Именно так рекомендуется делать при использовании **GRUB** - он принят в **Gentoo** в качестве загрузчика по умолчанию. Так вот, настало время обеспечить загрузку системы. Для этого даем команду **grub** и после появления одноименного приглашения последовательно вводим:

```
grub> root (hd0,0)  
grub> setup (hd0)
```

вслед за чем выходим из программы:

```
grub> quit
```

Начальный загрузчик установлен в первый раздел первого диска, тот самый,

который в текущий момент смонтирован у нас как **/boot**. И который в дальнейшем не будет монтироваться автоматически - дополнительная защита, в первую очередь от человеческого фактора. Так что в дальнейшем не следует удивляться, если, собрав новое ядро системы, вы не сможете его сразу скопировать в каталог **/boot** - предварительно он придется смонтировать вручную.

Теперь остается **GRUB** настроить. Для этого посредством того же редактора создаем файл его меню:

```
$ nano -w /boot/grub/menu.lst
```

и записываем в него две секции. Строки первой относятся к **GRUB** в целом:

```
default 0
timeout [значение_в_секундах]
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
```

Она указывает на ОС, загружаемую по умолчанию, время ожидания выбора в секундах и описывает путь к файлу графической заставки. Последняя штатно - нечто вроде аметиста с морионовым оттенком. Не нравится - заставку можно убрать (стерев или закомментировав третью строку). Или - изготовить собственную соответствующем формате.

Вторая секция описывает непосредственно загрузку **Gentoo**.

```
title=My Love Gentoo Linux
root (hd0,0)
kernel /boot/умя_образа root=/dev/hda3
```

Первая строка - произвольное название пункта меню. Вторая определяет имя устройства, с которого будет загружаться система. Имя это дается в нотации **GRUB**, отличной от принятой в Linux: дело в том, что **GRUB** способен загрузить ядро с не смонтированной файловой системы, относящейся к незагруженной операционке. И это распространяется не только на Linux, но на практически любую ОС из клана свободных Unix'ов (например, на FreeBSD или OpenBSD).

В третьей строке определяется имя образа ядра, подлежащего загрузке, и устройство, на котором после этого будет располагаться корневая файловая система. Если мы собирали собственное ядро, в качестве образа будет выступать тот самый файл **/boot/bzImage**, который был скопирован в каталог **/boot** после компиляции. Если же ядро не собиралось, значение **kernel**, как будет показано ниже, проще всего определить как **/boot/linux**. А имя устройства с корневым каталогом на этот раз дается в нотации Linux - ведь соответствующее ядро уже загружено, и оно знает, что такое **/dev/hda**?

Напомню еще раз, что все действия по редактированию меню GRUB можно выполнить, просто перетаскивая мышью соответствующие фрагменты руководства (там приведены все необходимые примеры) в наш **/boot/grub/menu.lst** и редактируя их в соответствии с собственными реалиями. И все же - супер финал

Настройкой загрузчика можно считать завершенной официальную часть инсталляции. Однако для полного счастья не хватает еще некоторых штрихов.

Так, руководство рекомендует (и это - более чем резонно) оптимизировать режим работы жесткого диска на предмет использования DMA - что весьма способствует его быстродействию. Для такого рода штук предназначен файл **/etc/conf.d/local.start**. Открываем его в редакторе и после комментария (по умолчанию там больше ничего нет) приписываем строку

```
hdparm -d 1 /dev/hda
```

Если физических дисков больше одного - повторяем ее для каждого устройства. А затем, воспользовавшись случаем, обеспечим здесь же и загрузку консольной мыши при старте системы:

```
gpm -t imps2 -m /dev/psaux
```

Теперь обратимся к локализации. Не смотря на свое исходно американское происхождение, **Gentoo** - дистрибутив интернациональный (это ясно из состава команды его разработчиков). И потому в нем есть все базовые средства для работы с языками, отличными от американского. Нужно только немного потрудиться для их активизации.

Достигается это можно редактированием файла **/etc/rc.conf**. Что может показаться необычным для пользователей Linux. Но мы помним, что в Gentoo принят порядок начальной загрузки в стиле BSD (а не в стиле System V, как в большинстве распространенных дистрибутивов Linux, за исключением Slackware и его сородичей). Впрочем, вдаваться в подробности пока не буду - это тема для отдельной беседы. Пока же - голая практика: все, что мы хотим видеть загруженным при старте машины, должно быть явным образом указано в файле **/etc/rc.conf**. И именно в том порядке, в каком требуется.

Так что открываем этот универсальный файл

```
$ nano -w /etc/rc.conf
```

отыскиваем в начале его строку

```
KEYMAP="us"
```

и заменяем чуждую нам Америку на что-нибудь родное, от братчиков Кирилла и Мефодия. Что именно - определяем в какой-либо из иных виртуальных консолей командой

```
$ ls /usr/share/keymaps/i386/qwerty/ru*
```

В ответ она выведет на экран такой список:

```
/usr/share/keymaps/i386/qwerty/ru1.map.gz  
/usr/share/keymaps/i386/qwerty/ru2.map.gz  
/usr/share/keymaps/i386/qwerty/ru3.map.gz  
/usr/share/keymaps/i386/qwerty/ru4.map.gz  
/usr/share/keymaps/i386/qwerty/ru-cp1251.map.gz  
/usr/share/keymaps/i386/qwerty/ru.map.gz  
/usr/share/keymaps/i386/qwerty/ru-ms.map.gz  
/usr/share/keymaps/i386/qwerty/ru_win.map.gz  
/usr/share/keymaps/i386/qwerty/ru-yawerty.map.gz
```

Можно видеть, что в нем присутствуют кириллические раскладки на все случаи

жизни (а пути к ним, хотя и взяты с Live CD, соответствуют тем, что будут в инсталлированной системе). Дело, конечно, сугубо личное (я бы даже сказал, интимное), но по мне предпочтительна ru4: это раскладка для кодировки **KOI8-R** с Windows-маркировкой клавиш (не путать с собственно раскладкой для Windows-кодировки - ее, как видно из списка, тоже можно использовать) и переключением с латиницы на кириллицу по нажатию CapsLock. Так что вписываем **KEYMAP="ru4"**

и переходим к следующей строке, определяющей экранный шрифт консоли - **CONSOLEFONT**. Очевидно, что умолчальное ее значение описывает шрифт без символов кириллицы. Но - их у нас есть, в чем легко убедиться, просмотрев каталог **/usr/share/consolefonts**. Шрифтов там немерянно, и среди них легко отыскиваются семейства **alt-*.psf.gz** и **cp866-*.psf.gz**. И то, и другое нам (почти) подходит - это экранные шрифты с символами кириллицы в кодировке **CP866** (она же - альтернативная, кодировка DOS или IBM). Почему так - ведь мы только что установили раскладку **KOI8-R** для клавиатуры, да и шрифты (семейства **koi8r-*.gz**), имеются? Распространяться не буду, ограничившись словами: это лучше, Ленья.

А слово "почти" я добавил потому, что все имеющиеся стандартно кириллические шрифты далеки от совершенства как эстетически, так и медицински. Так что в дальнейшем лучше будет воспользоваться шрифтами из пакета Виктора Вагнера **console-tools-cyrillic** - там есть очень симпатичные экземпляры, но об этом - как-нибудь в другой раз. Пока же просто вписываем, например, **CONSOLEFONT="cp866-8x16"**

И начинаем рассуждать логически: как же так, раскладка клавиатуры **KOI8**, а кодировка экранного шрифта - **CP866**. Ясно, что без загрузки карты трансформации одного в другое не обойтись. И такая карта у нас есть тоже - в каталоге **/usr/share/consoletrans** обнаруживаем файл **koi2alt**. Имя которого и вписываем в строку **CONSOLETRANSLATION="koi2alt"**

сняв с нее предварительно стоящий по умолчанию комментарий. Откладываем в уме, что таким образом карта соответствия будет загружена только на одной, первой, из виртуальных консолей (она же - консоль системная) - к этому мы еще вернемся. А пока, дабы покончить с файлом **/etc/rc.conf**, вспоминаем, что системные часы нашей машины установлены по местному часовому поясу (скорее всего - хотя в последнее время я осознал преимущества времени гринвичского) - тому самому, который мы определили через файл-ссылку **/etc/localtime**. И обращаем внимание, что строка **CLOCK="UTC"**

указывает у нас, напротив, именно на время по Гринвичу (UTC=GMT). Заменяем UTC на localtime - и файл **/etc/rc.conf** можно закрывать (прочие его строки можно оставить без изменений, если вам лично не известны веские причины их сделать).

А вот теперь пора довести до конца дело с картами соответствия. Открываем файл **/etc/init.d/local**: как явствует из названия, он предназначен для всякой всячины местного значения (и исполняется после всех остальных

конфигурационных файлов). И вставляем в его конец с помощью редактора (или перетаскиваем мышью из данного руководства) простенький скрипт (оператор цикла):

```
for i in 1 2 3 4 5 6
do
echo -ne "> /dev/vc/$i
done
```

В первой его строке определяется значение переменной *i* (ее можно обозвать любым другим образом) как диапазона цифр от 1 до 6 (по умолчанию у нас именно шесть виртуальных консолей), вторая - указывает, что следующей строкой будет тело цикла. А тело цикла - вывод "магической последовательности" ESC-(K (именно она и обеспечивает активизацию карты соответствия) и ее перенаправление на виртуальные консоли, номера которых заданы переменной *i*, то есть **/dev/vc/1** - **/dev/vc/6**. Это - если используется файловая система устройств (**devfs**), в противном случае (если не был снят комментарий со строки **tmpfs** в файле **/etc/fstab**) имена консолей будут обычными - **/dev/tty0** - **/dev/tty5**.

Отступление для любителей (или невольников) кодировки **CP1251**. Все сказанное о русификации применимо и к ней. Нужно только определить в файле **/etc/fstab** загрузку соответствующих раскладки клавиатуры (они, как уже говорилось, имеются) и карты соответствия **CP1251->CP866**. А вот ее придется поискать (где - за ненадобностью не помню) или просто слямзить из любого отечественного дистрибутива - Altlinux или ASPLinux.

Все же открою секрет - почему в консоли в любом случае предпочтительны экранные шрифты в кодировке DOS. Так уж исторически сложилось: вследствие аппаратных особенностей VGA-адаптеров (а в консоли мы имеем дело именно с режимом VGA, сколь бы крутой GeForce не стоял в машине) только они обеспечивают нормальный вывод на экран символов псевдографики. Кроме того, их просто больше, и некоторые среди них, как я говорил, лучше. Так что есть резон загружать именно их, даже если программы с псевдографикой не используются совсем. В том числе и **Midnight Commander** - но какой же русский не любит детей командира Нортон!

Мы же пока завершим русификацию. Для чего установим собственно локаль последней строкой в файле **/etc/init.d/local** (опять прекрасный каламбур, господа!):

```
export LANG="ru_RU.KOI8-R"
```

Она обеспечит установку всех локально-зависимых переменных (не только страны, языка, набора символов, но и формата даты, времени, денежной единицы, и так далее) в значения, принятые для нашей многострадальной Отчизны. Тем не менее, для представления чисел я сделал бы исключение, добавив строку

```
export LC_NUMERIC="POSIX"
```

Ведь в любых программах, связанных со счетом, удобнее оперировать с десятичной точкой, а не запятой (а говорят, что некоторые такие программы этого просто требуют).

В итоге, выполнив несложную последовательность действий, мы получили корректно русифицированную систему - русифицированную ничуть не хуже, чем те дистрибутивы, кириллической поддержкой в которых гордятся (и - справедливо гордятся) именитые отечественные разработчики - Altlinux и ASPLinux. А поскольку мы сделали все это своими руками и с полным пониманием сути действий, у нас тоже есть основание для гордости. Правда, "лобовым" способом, без учета специфики дистрибутива - но зато работающим всегда и везде. А к "идеологически правильному" (для **Gentoo**) методу локализации можно будет вернуться позднее.

Можно, конечно, активизировав внесенные в конфигурацию изменения командой `etc-update`, продолжать устанавливать программы (Иксы, например). Или даже начинать обычную работу - установленная система, внутри которой мы находимся, это вполне уже позволяет. Однако для полной проверки лучше все-таки сначала перезагрузиться.

Итак, временно расстаемся с нашим новым Gentoo командой `exit`, после чего вновь оказываемся в системе Live CD, той же командой зарываем все виртуальные консоли, кроме первой (а у нас их могло расплодиться немало) и готовимся к выходу.

Хотя - стоп. А собирали ли мы при установке ядро системы. Если да - все в порядке, тотальный **`umount`**:

```
$ umount /mnt/gentoo/boot  
$ umount /mnt/gentoo
```

и что было смонтировано руками, а затем - **`reboot`** (или **`halt`**, если есть желание прерваться - при первой загрузке хорошо бы внимательно следить за ее ходом по сообщениям на экране).

Если же ядро не собиралось (за отсутствием каких-то исходников - другого оправдания такому разгильдяйству нет), то совершаем последнее установочное действие: просто тупо копируем ядро с Live CD (оно, как мы уже убедились, не только работоспособно, но и поддерживает большинство жизненно необходимых функций):

```
$ cp /cdroot/boot/linux /mnt/gentoo/boot/
```

Правда, у нас не собиралось не только ядро, но и модули. И это не беда, их тоже копируем с CD:

```
cp -R /cdroot/lib/modules /mnt/gentoo/lib/modules
```

А далее - все то же, размонтирование и перезагрузка. Конечно, в ходе ее мы рискуем получить кое-какие сообщения об ошибках, но, по моему опыту - ни одна из них не критична. И загрузившаяся система будет вполне пригодной к использования. По крайней мере, до тех пор, пока не разживемся всеми необходимыми архивами и все-таки ядро не пересоберем - от этого все равно не уйти.

Заключительный аккорд

У терпеливого читателя, добравшегося до этих строк, может возникнуть три совершенно законных вопроса: не слишком ли все это сложно, на кой все это нужно, и кому все это нужно. Постараюсь ответить последовательно.

Относительно сложности - могу заверить, что это из той самой оперы, когда глаза боятся, а руки делают. Проверено на собственном опыте - причем первый из них проводился в подмосковной деревне, где не то что постоянного подключения к Сети - и телефона-то с модемом не было. И все исходники скачивались на службе (в полутора часах езды) и таскались домой на CD и съемном винчестере. И если праведно выполнять все заветы инструкции по установке, то сложностей возникнет не много (надеюсь, что и настоящий мемуар поспособствует их преодолению).

Второй вопрос - а зачем их было преодолевать, когда можно легко и быстро развернуть любой пакетный дистрибутив, разработчики которого уже решили все мои проблемы. Отвечаю: решить-то они решили, но - не скажу плохо, просто в меру своих представлений, как это нужно народу (и конкретно мне, как его представителю). **Gentoo** же позволяет не просто поставить систему, идеально оптимизированную под железо, но и настроить ее тоже в соответствие именно со своим представлением об идеале. Причем - с полным осознанием, что и зачем делается, немало способствующим общему образованию.

Ну а третий ответ очевиден из второго. Дистрибутив Gentoo нужен пользователю, которому требуется система, настроенная на максимальную производительность для конкретной тяжелой задачи - счетной, графической, GIS или CAD. Разработчику, которому надоело проводить время компиляции тысяч и тысяч строк кода за игрой в тетрис. Сисадмину, у которого в системе не должно быть ничего лишнего, но все необходимое должно быть. Изучающему Linux, которого не устраивают готовые рецепты, предлагаемые руководствами пакетных дистрибутивов. И наконец, тому, кто просто стремится к идеалу - пусть идеал этот и недостижим...

Краткое руководство по установке Gentoo Linux x86

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/gentoo-x86-quickinstall.xml>

С версии: 1.0

Установочный диск

Скачайте компакт-диск с одного из наших зеркал. ISO-образы находятся в каталогах: **releases**/**<архитектура>**/**<выпуск>**/установочный-диск. Минимальный установочный диск подходит только для установки из интернета; с универсального диска также возможна установка без обращения к сети.

Запишите компакт-диск, затем загрузитесь с него. В загрузочном окне нажмите F2, чтобы выяснить, какие варианты загрузки предлагаются. Если вам нужна поддержка PCMCIA, сразу после загрузки потребуется запустить сценарий инициализации `pcmcia`.

Установочные диски позволяют запускать сервер `sshd`, добавлять новых пользователей, запускать `irssi` (консольный чат-клиент) и просматривать веб-страницы с помощью `lynx` или `links2`.

Настройка сети

Если ваша сеть еще не работает, для ее настройки можно использовать **net-setup**. Перед настройкой вам может потребоваться загрузить поддержку своей сетевой платы с помощью `modprobe`. Если у вас ADSL, используйте **adsl-setup** и **adsl-start**. Для поддержки PPTP, сначала отредактируйте `/etc/ppp/chap-secrets` и `/etc/ppp/options.pptp`, а затем запустите **pptp** `<ip сервера>`.

При беспроводном доступе для установки параметров соединения используйте **iwconfig**, а затем либо опять **net-setup**, либо запустите вручную **ifconfig**, **dhcpcd** и/или **route**.

Если вы находитесь за прокси-сервером, не забудьте инициализировать систему с помощью `export http_proxy, ftp_proxy` и `RSYNC_PROXY`.

Подготовка дисков

Для организации разделов запустите **fdisk** или **cfdisk**. Вам нужен хотя бы раздел подкачки (тип 82) и один раздел Linux (тип 83).

Для создания файловых систем на своих Linux-разделах используйте **mke2fs**, **mke2fs -j**, **mkreiserfs**, **mkfs.xfs** **mkfs.jfs**. Подготовьте свой раздел подкачки командами **mkswap** и **swapon**.

Смонтируйте свежесозданные файловые системы в `/mnt/gentoo`. Создайте каталоги для других точек монтирования (например, `/mnt/gentoo/boot`), если они вам нужны.

Установка стадии

Прежде всего, установите верную дату командой **date** MMDDhhmmYYYY. Далее скачайте файл стадии с одного из наших зеркал или используйте находящийся на установочном диске (в **/mnt/cdrom/stages**). Перейдите в **/mnt/gentoo** и распакуйте стадию командой **tar -xvjpf** <архив стадии>.

В случае установки без сети установите снимок портежей: перейдите в **/mnt/gentoo/usr** и запустите **tar -xvjf /mnt/cdrom/snapshots/<файл снимка>**. Другие пользователи таким же образом могут скачать снимок портежей и установить его.

При установке без сети, скопируйте файлы исходников из **/mnt/cdrom/distfiles/** в **/mnt/gentoo/usr/portage/distfiles/**.

Отредактируйте **/mnt/gentoo/etc/make.conf** под ваши нужды (флаги **USE**, **CFLAGS** и **CXXFLAGS**). Для этого можно использовать редактор **nano**.

Установка базовой системы Gentoo

Сначала смонтируйте файловую систему **/proc**, скопируйте файл **/etc/resolv.conf** поверх существующего, а затем смените корневой каталог (**chroot**) на свою среду **Gentoo**.

Листинг 1.1: Подготовка и смена корневого каталога

```
# mount -t proc none /mnt/gentoo/proc
# cp /etc/resolv.conf /mnt/gentoo/etc/
# chroot /mnt/gentoo /bin/bash
# env-update && source /etc/profile
```

Если вы не выполняете бессетевую установку, командой **emerge --sync** обновите свое дерево портежей.

Далее убедитесь, что **/etc/make.profile** указывает на верный профиль. Профиль по умолчанию должен подойти большинству пользователей; имеются подпрофили для различных ядер (например, 2.4/ для профилей, основанных на ядре 2.4). Изменяйте профиль командой **ln -sfn**.

Самогенерация системы (не выполняется при бессетевой установке) запускается сценарием **scripts/bootstrap.sh** в каталоге **/usr/portage**.

Установка системы (не выполняется при бессетевой установке) запускается командой **emerge -e system** (или **-N**, если вы не меняли значения **CFLAGS/CXXFLAGS** по умолчанию).

Настройка ядра

Установите сведения о своем часовом поясе, сменив символическую ссылку **/etc/localtime** на указывающую на верное место в **/usr/share/zoneinfo** командой **ln -sf**.

Установите исходный код ядра (при установке без сети доступны **gentoo-sources** и **vanilla-sources**) и настройте его с помощью **make menuconfig**, затем **make && make modules_install** внутри каталога **/usr/src/linux**. Скопируйте файл **arch/i386/boot/bzImage** в **/boot**, заменяя существующий. Также можно выполнить **emerge genkernel** и **genkernel all**.

Пользователям, использующим **Genkernel**, потребуется выполнить **emerge coldplug** и **rc-update add coldplug default**.

Настройка системы

Отредактируйте свой **/etc/fstab**; пример показан ниже.

Листинг 1.2: Пример файла **fstab**

/dev/hda1	/boot	ext2	defaults,noatime	1 2
/dev/hda2	none	swap	sw	0 0
/dev/hda3	/	ext3	noatime	0 1
none	/proc	proc	defaults	0 0
none	/dev/shm	tmpfs	nodev,nosuid,noexec	0 0
/dev/cdroms/cdrom0	/mnt/cdrom	auto	noauto,user	0 0

Для настройки своей сети отредактируйте **/etc/conf.d/hostname** и **/etc/conf.d/domainname**, запустите **rc-update add domainname default**, затем отредактируйте **/etc/conf.d/net**. Добавьте сценарий инициализации **net.eth0** к уровню запуска по умолчанию (**default run level**). Если у вас несколько сетевых карт (NIC), поместите символические ссылки на них в сценарий инициализации **net.eth0**, и также добавьте их в уровень запуска по умолчанию.

Отредактируйте **/etc/hosts**; примеры приведены ниже:

Листинг 1.3: Примеры файла **/etc/hosts**

(Для статических IP-адресов)

```
127.0.0.1    localhost
192.168.0.5  jenny.homenetwork jenny
192.168.0.6  benny.homenetwork benny
192.168.0.7  tux.homenetwork tux
```

(Для динамических IP-адресов)

```
127.0.0.1    localhost.homenetwork tux localhost
```

При необходимости установите (**emerge**) **pcmcia-cs** и добавьте его в уровень

запуска по умолчанию.

Установите пароль **root** с помощью **passwd**.

Выполните необходимую настройку в файлах: **/etc/rc.conf**, **/etc/conf.d/rc**, **/etc/conf.d/keymaps**, **/etc/conf.d/clock**.

Установка системных утилит

Пользователям ядра 2.4 нужно запустить **emerge --unmerge udev** и **emerge devfsd**.

Установите системный журнал, например **syslog-ng**, и добавьте его в уровень запуска по умолчанию. Сделайте то же самое для демона **cron**, например, **viXie-cron** (необязательный шаг).

Установите необходимые средства для файловых систем (**xfsp****rogs**, **reiserfsprogs** или **jfsutils**), а также для сети (**dhcpcd** или **rp-pppoe**).

Настройка загрузчика

Установите (**emerge**) **grub** или **lilo**. Отредактируйте **/boot/grub/grub.conf** или **/etc/lilo.conf**, соответственно. Ниже приведены примеры обоих.

Grub

Листинг 1.4: Пример **grub.conf**

```
default 0
timeout 30
splashimage=(hd0,0)/grub/splash.xpm.gz
# для пользователей genkernel
title=Gentoo Linux 2.6.11-r3
root (hd0,0)
kernel /kernel-genkernel-x86-2.6.11-gentoo-r3 root=/dev/ram0 init=/linuxrc ramdisk=8192
real_root=/dev/hda3 udev
initrd /initramfs-genkernel-x86-2.6.11-gentoo-r3
# для остальных пользователей (без initrd)
title=Gentoo Linux 2.6.11 r3
root (hd0,0)
kernel /kernel-2.6.11-gentoo-r3 root=/dev/hda3
# только если вам нужна двойная загрузка
title=Windows XP
root (hd0,5)
makeactive
chainloader +1
```

Lilo

Листинг 1.5: Пример файла lilo.conf

```
boot=/dev/hda
prompt
timeout=50
default=gentoo

# для пользователей, не использующих genkernel
image=/boot/kernel-2.6.11-gentoo-r3
label=gentoo
read-only
root=/dev/hda3

# для пользователей genkernel
image=/boot/kernel-genkernel-x86-2.6.11-gentoo-r3
label=gentoo
read-only
root=/dev/ram0
append="init=/linuxrc ramdisk=8192 real_root=/dev/hda3 udev"
initrd=/boot/initramfs-genkernel-x86-2.6.11-gentoo-r3

# Для двойной загрузки
other=/dev/hda6
label=windows
```

Пользователям **GRUB** необходимо установить **GRUB** в главную загрузочную запись (**MBR**) командой **grub-install /dev/hda** после копирования **/proc/mounts** в **/etc/mtab**. Пользователям **LILO** нужно запустить **/sbin/lilo**.

Выйдите из изолированной среды, размонтируйте все файловые системы и перезагрузитесь.

Окончание установки

Зайдите как **root**, затем добавьте одного или нескольких пользователей для повседневного использования командой **useradd -m -G <группа> <имя пользователя>**.

Если вы выполняли установку без сети, смонтируйте компакт-диск с пакетами в **/mnt/cdrom** и выполните **export PKGDIR="/mnt/cdrom"**, после чего вы сможете устанавливать дополнительное ПО, например, **kde**, командой **emerge -k <имя пакета>**. для .

Спасибо за установку Gentoo!

Описание альтернативных способов установки Gentoo Linux

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/altinstall.xml>

С версии: 1.0

1. Об этом документе

Если у вас не работает стандартный метод установки **boot-from-CD** (или он вас не устраивает), вам сюда. Этот документ описывает альтернативные приемы установки Gentoo Linux для тех, кто в них нуждается. Или, если хотите, это место сбора всех "оригинальных" способов установки. Если у вас один такой завалялся, или вы откопали новую прикольную методу, не стесняйтесь, написать запрос в Bugzilla.

2. Загрузка с установочного диска при помощи Smart BootManager

Скачайте **Smart BootManager** с <http://btmgr.sourceforge.net/download.html>. Доступны как исходники, так и бинарники для Linux, не говоря уже про виндовский .exe и множество языковых пакетов. Все-таки, в настоящее время, бинарный формат предпочтительнее, поскольку исходники могут не откомпилироваться современными версиями NASM.

Скомпилируйте пакет или возьмите уже готовый. Ниже я расскажу о некоторых полезных опциях, доступных при создании загрузочной дискеты.

Листинг 2.1: Опции Smart BootManager

sbminst [-t theme] [-d drv] [-b backup_file] [-u backup_file]

-t тема выбор используемой темы, где тема должна быть:
us = English тема de = German тема
hu = Hungarian тема zh = Chinese тема
ru = Russian тема cz = Czech тема
es = Spanish тема fr = French тема
pt = Portuguese тема

-d диск выбор диска, куда вы собираетесь инсталлировать Smart BootManager;
для Linux:
/dev/fd0 первый floppy-диск,
/dev/hda первый IDE-диск.
/dev/sda первый SCSI-диск.
для DOS:
0 первый floppy-диск
128 первый жесткий диск;


```
-c      отключить загрузку с CD-ROM;

-b backup_file сохранить (для будущей деинсталляции) данные, которые будут
    перезаписаны;

-u backup_file деинсталлировать Smart BootManager, нельзя использовать
    с другими ключами;

-y      никаких вопросов и предупреждений.
```

Листинг 2.2: Использование sbminst для создания загрузочной дискеты

```
# sbminst -t us -d /dev/fd0
```

Примечание: Замените fd0 на имя вашего floppy-дисковода (если у вас он называется по-другому).

Теперь просто вставьте дискету в дисковод компьютера, на котором вы хотите запустить установочный диск, и загрузитесь с нее.

Вас поприветствует Smart BootManager. Выберите CD-ROM и нажмите ENTER для загрузки установочного диска. Если все прошло удачно, следуйте стандартной процедуре установки.

Больше информации по Smart BootManager вы найдете на <http://btmgr.sourceforge.net/>

3. Установка при помощи Knoppix

Примечание: Knoppix доступен только для пользователей x86.

Загрузившись с Knoppix LiveCD, вы получаете полностью работоспособную систему для компиляции Gentoo. Во время сборки Tux Racer скрасит ваше ожидание, а для работы вы можете пользоваться OpenOffice.

Предупреждение: Помните, что если вы что-либо сохраните в домашний каталог Knoppix, ожидая установки своей системы Gentoo, вы потеряете это после перезагрузки в Gentoo. Убедитесь, что сохраняете важные файлы на жесткий диск или на другой компьютер!

Загрузитесь с Knoppix CD. Если у вас Knoppix 3.6-3.8.2, потребуется указать knoppix26 в качестве параметра ядра, чтобы загрузить ядро 2.6. Если вы пропустите этот шаг, то при chroot вы получите сообщение об ошибке, что ваше ядро слишком старое. Однако, если у вас Knoppix 3.9+, этот шаг не потребуется, так как ядро 2.6 выбрано по умолчанию.

По умолчанию вы попадаете на рабочий стол KDE. Откройте консоль и введите su -, чтобы была возможность изменить пароль. Это позволяет задать пароль для

root в Knoppix. Теперь, при желании, можете настроить sshd для удаленного входа в систему.

Листинг 3.1: Создание точки монтирования /mnt/gentoo

```
# mkdir /mnt/gentoo
```

Далее вы можете придерживаться стандартного руководства по установке, начиная с части 4. Кроме монтирования /proc. Тут вам надо поступить по-другому:

Листинг 3.2: Монтирование с привязкой псевдо-ФС proc

```
# mount -o bind /proc /mnt/gentoo/proc
```

Также знайте, что некоторые возможности (FEATURES) портежей не будут работать в knoppix. Будьте особенно осторожны с userpriv и usersandbox. Если вы обнаружили, что начали получать ошибки, будет разумным отключить все или некоторые необязательные возможности.

4. Установка со Stage 1 без доступа в сеть

Создайте установочный компакт-диск из iso-образа.

Возьмите последний snapshot портежей из <http://distro.ibiblio.org/pub/linux/distributions/gentoo/snapshots/> (или с любимого зеркала). Запишите этот tarball на существующий раздел в компьютере, где будет проходить установка, или прожгите его на CD.

Следуйте инструкции из Gentoo Install Doc вплоть до chroot /mnt/gentoo в главе 6. Если у вас в наличии только один привод CD-ROM, не забудьте при загрузке ввести опцию docache чтобы иметь возможность размонтировать установочный компакт-диск и смонтировать CD со snapshot-ом портежей.

Открыв новую консоль (Alt-F2), а в ней руководство по установке, мы можем следовать ему вплоть до запуска скрипта bootstrap.sh.

Предупреждение: В старых выпусках установочного компакт-диска требуется изменить пароль командой passwd прежде чем логиниться с другой консоли.

Возвращаемся к первой консоли (Alt-F1, without chroot) и монтируем второй CD в /mnt/gentoo/mnt/cdrom2. Копируем portage tarball с cdrom2 и распаковываем в /mnt/gentoo/usr.

Листинг 4.1: Монтирование snapshot cd

```
# umount /mnt/cdrom
# mkdir /mnt/gentoo/mnt/cdrom2
# mount /dev/cdroms/cdrom0 /mnt/gentoo/mnt/cdrom2
# cp /mnt/gentoo/mnt/cdrom2/portage-$date.tar.bz2 /mnt/gentoo/usr
# cd /mnt/gentoo/usr
# tar -xvjp portage-$date.tar.bz2
```

Снова переключаемся на вторую консоль. Если сейчас попытаться запустить bootstrap.sh, ничего не выйдет, так как он не сможет закачать ни одного файла.

Мы получим эти файлы из другого места и разместим их в /usr/portage/distfiles (в консоли F2).

Нам необходим список пакетов Stage1: glibc, baselayout, texinfo, gettext, zlib, binutils, gcc, ncurses плюс их зависимости.

Примечание: Не забывайте, что вам нужны версии пакетов, синхронизированные с вашим деревом портежей.

Листинг 4.2: Получение списка для скачивания

(Не забывайте 2 перед >)

```
# emerge -fp glibc baselayout texinfo gettext zlib binutils gcc ncurses 2> stage1.list
# mount -t vfat /dev/fd0 /mnt/floppy
# cp /mnt/gentoo/stage1.list /mnt/floppy
# umount /mnt/floppy
```

Вставьте дискету в компьютер с хорошим каналом. Если заглянуть в файл stage1.list, можно увидеть, что в нем перечисляются несколько ссылок на каждую загрузку. К сожалению, это не совсем то, что нам надо. Сначала список надо почистить от лишнего:

Листинг 4.3: Stripping URLs

(Этот скрипт привязан к формату вывода emerge, который может измениться без предварительного предупреждения - используйте с осторожностью!)

```
# cut -f 1 -d ' ' stage1.list > stage1.download
```

Теперь используйте wget для загрузки списка исходников:

Листинг 4.4: Загружаем пакеты с помощью wget

```
# wget -N -i stage1.download
```

Получив все файлы, переносим их в наш компьютер в /mnt/gentoo/usr/portage/distfiles. Вот теперь можно запускать bootstrap.sh. Повторяем ту же операцию с wget для stage2 и 3.

5. Бездисковая установка с загрузкой через PXE

Что необходимо

Нужна сетевая карта на бездисковом клиенте, которая способна загружаться по протоколу PXE (многие карточки 3com). BIOS тоже должен поддерживать такой способ загрузки.

Настройки на стороне сервера

Создание папок: Во-первых, следует создать папки, в которых будет размещена ваша бездисковая система. Начнем с /diskless, в ней будет находиться по отдельной папке для каждого бездискового клиента. Далее мы рассмотрим работу с клиентом 'eta'.

Листинг 5.1: создание папок

```
# mkdir /diskless
# mkdir /diskless/eta
# mkdir /diskless/eta/boot
```

Настройка DHCP и TFTP: Клиент получает загрузочную информацию, используя DHCP, затем загружает необходимые файлы с помощью TFTP. Собираем DHCP командой `emerge` и настраиваем его для наших нужд. Добавляем следующее в `/etc/dhcp/dhcpd.conf`:

Примечание: Мы получим статический IP для клиента и путь к загрузочному образу PXE, здесь `pxegrub`. Вам необходимо подставить соответствующий MAC-адрес сетевой карты вашего клиента и определиться с директорией, где будут храниться клиентские файлы.

Для DHCPd, запустите `emerge dhcp` (или любой другой сервер DHCP на свое усмотрение). Убедитесь, что в `/etc/conf.d/dhcp` выбран верный интерфейс.

Листинг 5.2: dhcp.conf

```
option option-150 code 150 = text ;
ddns-update-style none ;
host eta {
hardware ethernet 00:00:00:00:00:00;
fixed-address ip.add.re.ss;
option option-150 "/eta/boot/grub.lst";
filename "/eta/boot/pxegrub";
}
```

Для установки TFTP, соберите `app-admin/tftp-hpa`. Впишите следующее в `/etc/conf.d/in.tftpd`:

Листинг 5.3: in.tftpd

```
INTFTPD_PATH="/diskless"
INTFTPD_USER="nobody"
INTFTPD_OPTS="-u ${INTFTPD_USER} -l -vvvvv -p -c -s ${INTFTPD_PATH}"
```

Настройка GRUB: Я использую GRUB для PXE-загрузки. Чтобы можно было использовать PXE, вы должны самостоятельно скомпилировать GRUB ... ведь это просто. Во-первых, скачайте самую свежую версию исходников GRUB (`emerge -f grub` поместит tarball в `/usr/portage/distfiles`). Скопируйте tarball в `/diskless` и скомпилируйте GRUB, чтобы получить доступный для pxe бинарник. Готовый бинарник копируем в `boot`-папку бездискового клиента. И редактируем файл `grub.lst`:

Листинг 5.4: установка grub

```
# tar zxvf grub-0.92.tar.gz
# cd grub-0.92
# ./configure --help
(Среди опций вы увидите список поддерживаемых драйверов сетевых интерфейсов.)
(Выберите драйвер, совместимый с вашей картой. Здесь он фигурирует как $nic)
# ./configure --enable-diskless --enable-$nic
# make
# cd stage2
# cp pxegrub /diskless/eta/boot/pxegrub
```

```
# nano -w /diskless/eta/boot/grub.lst
```

Листинг 5.5: grub.lst

```
default 0
timeout 30

title=Diskless Gentoo
root (nd)
kernel /eta/bzImage ip=dhcp root=/dev/nfs nfsroot=ip.add.re.ss:/diskless/eta
```

(В опции nfsroot указывается IP адрес сервера и путь к директории,)
(в которой на сервере лежат файлы нашего бездискового клиента.)

Настройка NFS: NFS легко сконфигурировать. Все что нам надо - это добавить одну строку в конфигурационный файл /etc/exports:

Листинг 5.6: /etc/exports

```
# nano -w /etc/exports
# /etc/exports: NFS file systems being exported. See exports(5).
/diskless/eta eta(rw,sync,no_root_squash)
```

Обновление hosts: Еще одно важное дело - изменение файла /etc/hosts.

Листинг 5.7: /etc/hosts

```
127.0.0.1 localhost

192.168.1.10 eta.example.com eta
192.168.1.20 sigma.example.com sigma
```

Создание системы на сервере

Вы можете захотеть перезагрузить сервер с установочного диска Gentoo, хотя точно так же можете немедленно продолжить, если знать, как выполнять инструкции по установке Gentoo на установленной системе. Следуйте стандартной процедуре установки, как описано в Gentoo Install Howto, НО с учетом следующих отличий: Для монтирования файловой системы сделайте следующее (здесь hdaX - раздел, в котором создана папка /diskless). Вам не надо монтировать другие разделы, так как все файлы располагаются в /diskless/eta.

Листинг 5.8: монтирование файловой системы

```
# mount /dev/hdaX /mnt/gentoo
```

Stage tarballs и chroot: Этот пример использует tarball stage3. Смонтируйте /proc в вашу бездисковую директорию и выполните chroot в нее(директорию) для продолжения установки. Дальше - по инструкции вплоть до конфигурации ядра. Предупреждение: Будьте очень внимательны при разархивировании stage tarball. Не распакуйте его в существующую систему.

Листинг 5.9: извлечение stage tarball

```
# cd /mnt/gentoo/diskless/eta/
# tar -xvjp /mnt/cdrom/gentoo/stage3-*.tar.bz2
# mount -t proc /proc /mnt/gentoo/diskless/eta/proc
```

```
# cp /etc/resolv.conf /mnt/gentoo/diskless/eta/etc/resolv.conf
# chroot /mnt/gentoo/diskless/eta/ /bin/bash
# env-update
# source /etc/profile
```

Конфигурация ядра: При выполнении **make menuconfig** во время конфигурации ядра, не забудьте сделать доступными следующие опции (помимо рекомендуемых в руководстве по установке):

Листинг 5.10: опции menuconfig

```
- Your network card device support
(в ядре, *не* как модуль!)

- Under "Networking options" :

[*] TCP/IP networking
[*] IP: kernel level autoconfiguration
[*] IP: DHCP support
[*] IP: BOOTP support

- Under "File systems --> Network File Systems" :

<*> NFS file system support
[*] Provide NFSv3 client support
[*] Root file system on NFS
```

Сохраните ядро в / своей изолированной среды (не в /boot), в соответствии с настройкой rhexgrub, определенной ранее. Затем настройте /etc/fstab своего бездискового клиента.

Листинг 5.11: /etc/fstab

```
# nano -w /etc/fstab

/dev/cdroms/cdrom0 /mnt/cdrom iso9660 noauto,ro 0 0
proc /proc proc defaults 0 0
tmpfs /dev/shm tmpfs nodev,nosuid,noexec 0 0
```

Вам также потребуется предотвратить запуск на клиенте проверки файловой системы:

Листинг 5.12: Исключение запуска проверки файловой системы на клиенте

```
# touch /fastboot
# echo "touch /fastboot" >> /etc/conf.d/local.start
```

Установите nfs-utils, так как ваш клиент будет сильно зависеть от них:

Листинг 5.13: Установка nfs-utils

```
# emerge nfs-utils
```

Загрузчик. Не устанавливайте других загрузчиков, так как один у нас уже есть - rhexgrub. Просто завершаете установку и перезагружаете сервер. Запустите службы, необходимые для загрузки новоиспеченного клиента: DHCP, TFTP и NFS.

Листинг 5.14: Запуск служб

```
# /etc/init.d/dhcp start
# /etc/init.d/in.tftpd start
# /etc/init.d/nfs start
```

Загрузка нового клиента

Чтобы клиент загрузился, надо настроить bios и сетевую карту для использования PXE как первого способа загрузки - до CD-ROM или floppy. Помощником вам будет документация к оборудованию или website производителя. Сетевая карта должна получить IP адрес от DHCP и загрузить PXE-образ GRUB при помощи TFTP. Дальше вы увидите симпатичное черно-белое меню GRUB, в котором можно выбрать ядро для загрузки, а нажав Enter, и загрузить его. Если все в порядке, ядро примонтирует корневую файловую систему, используя NFS, и выведет на терминал приглашение для входа в систему. Вот и все.

6. Установка Gentoo из существующей Linux-системы

Что необходимо

Для установки Gentoo из существующей системы Linux вам должна быть доступна команда chroot, а также копия установочного tarball-а Gentoo, либо его ISO-образ. Не мешает доступ в сеть, если вы не хотите ограничиваться тем, что есть в вашем tarball-е. (к слову, tarball - это просто файл, заканчивающийся на .tbz или .tar.gz). Автор (ориг.текста) использовал RedHat Linux 7.3 как "базовую" ОС, хотя это не так уж важно. Итак, приступим!

Обзор

Сначала надо отвести раздел для Gentoo, изменив размер существующего раздела Linux, смонтировать его, разархивировать tarball, chroot внутрь нашей псевдо-системы, и начинаем строительство. После настройки загрузчика немного поработаем напильником, просто для того, чтобы система была способна загрузиться самостоятельно, перезагружаемся и пользуемся Gentoo.

Как отвести место для Gentoo?

Корневой раздел - это файловая система, примонтированная к /. Запущенная команда mount показывает, о чем я веду речь. Еще мы можем выполнить df (disk free), чтобы посмотреть, сколько свободного места у нас есть. Прошу заметить - совсем не обязательно кромсать ваш root раздел! Вы можете отрезать в любом другом месте, доступном вашей программе редактирования разделов жесткого диска, но мы поговорим об этом чуть позже.

Листинг 6.1: Информация о файловых системах

```
# mount
/dev/hdb2 on / type ext3 (rw)
```



```
none on /proc type proc (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
none on /dev/shm type tmpfs (rw,nodev,nosuid,noexec)
# df -h
Filesystem      Size Used Avail Use% Mounted on
/dev/hdb2       4.0G 1.9G 2.4G 82% /
none            38M  0  38M  0% /dev/shm
```

Как видите, раздел, смонтированный как /, носит имя /dev/hdb2 и на нем еще свободно 2.4 гигабайта. В моем случае, думаю, 400 МБ свободного пространства хватит с головой, следовательно, под Gentoo отводим 2 ГБ. Совсем неплохо, учитывая, что большинству пользователей хватит и одного гигабайта. Приступим!

Сборка parted для переразбивки разделов

Parted - утилита, поддерживаемая GNU foundation, старым, уважаемым, безразмерным проектом, софтом которого мы в данный момент и собираемся воспользоваться. Всего одним инструментом, который вполне нам подходит. Называется он parted, partition editor, загрузить его можно из <http://www.gnu.org/software/parted/>

Примечание: Конечно, существуют и другие инструменты для работы с разделами, но автор не уверен/не интересуется, сможет ли PartitionMagic(tm) или любая другая программа сделать то, что нам нужно. Проверить - это забота читателя.

Взгляните на ту страницу руководства parted, где речь идет о типе файловой системы, размер которой вы собираетесь изменять, и узнайте, сможет ли parted это сделать. Если нет, что ж, вам не повезло. Придется убить какой-нибудь раздел и создать по-новому, чтобы было место для Gentoo. Вперед, качаем программу, устанавливаем. Возникает проблема. Мы хотим изменить размер корневого раздела, следовательно нам нужно загрузить с дискеты минимальную linux-систему и использовать предварительно скомпилированный parted для редактирования /. Если в вашем случае можно отмонтировать выбранный раздел, оставаясь в Linux, вам везет и нет необходимости делать то, что описано ниже. Просто соберите parted и запустите его на разделе, который хотите урезать. А вот то, что я сделал в своей системе.

Важно: Убедитесь, что действия, которые вы собираетесь проделать с разделом, поддерживаются parted!

Скачайте загрузочный/корневой диск mininux (бесплатный мини-дистрибутив Linux с ядром 2.4 на дискете) с <http://mininux.free.fr/uk/>, создайте дискету, как описано в прилагающейся документации, вставьте еще одну чистую дискету в дисковод и перейдите к следующему шагу.

Примечание: Опять замечу, что Linux - синоним понятия "Этот путь не единственный". Наша цель - запустить parted на отмонтированном разделе жесткого диска и это можно сделать не только описанным способом. Вы можете использовать любой другой набор boot/root дискет, не только mininux. Вы можете

вообще не выполнять данный шаг, главное, что нужно - отмонтировать файловую систему, которую вы собираетесь использовать под Gentoo, и перекроить ее, используя parted.

Листинг 6.2: Создание вспомогательной дискеты

```
# mkfs.minix /dev/fd0
480 inodes
1440 blocks
Firstdatazone=19 (19)
Zonesize=1024
Maxsize=268966912
```

Продолжим со сборкой parted. Если он еще не закачан и не распакован, сделайте это. А затем перейдите в каталог, куда вы его распаковали. И здесь выполните следующий набор команд для сборки утилиты и записи ее на дискету.

Листинг 6.3: Подготовка вспомогательной дискеты

```
# mkdir /floppy; mount -t minix /dev/fd0 /floppy &&
export CFLAGS="-O3 -pipe -fomit-frame-pointer -static" && ./configure
&& make && cp parted/parted /floppy && umount /floppy
```

Поздравляю, вы готовы перезагрузиться и изменять выбранный раздел. Только делайте это после хотя бы беглого просмотра документации к parted на GNU website. Наберитесь терпения - изменение размера может занять больше получаса на больших дисках. Загрузитесь с дискеты mininux, а после этого замените загрузочную дискету на вспомогательную, которую мы только что создали. Введите mount /dev/fd0 /floppy и parted будет доступен в папке /floppy. Нам туда. Запускаете parted и перекраиваете нужный раздел жесткого диска. Когда этот бесконечный процесс наконец завершится, все будет готово для того, чтобы получить истинное наслаждение, устанавливая Gentoo. Диск, с которым мы работаем - это тот, на котором находится раздел, размер которого мы меняем. К примеру, если мы хотим изменить /dev/hda3, то диск, соответственно, будет называться /dev/hda.

Листинг 6.4: Команды, которые следует выполнить, загрузившись с mininux

```
# mount /dev/fd0 /floppy
# cd /floppy; ./parted [диск, с которым мы работаем]
(parted) print
Disk geometry for /dev/hdb: 0.000-9787.148 megabytes
Disk label type: msdos
Minor  Start   End     Type    Filesystem  Flags
1       0.031 2953.125 primary ntfs
3    2953.125 3133.265 primary linux-swap
2    3133.266 5633.085 primary ext3
4    5633.086 9787.148 extended
5    5633.117 6633.210 logical
6    6633.242 9787.148 logical ext3
(parted) help resize
resize MINOR START END      resize filesystem on partition MINOR

MINOR is the partition number used by Linux. On msdos disk labels, the
primary partitions number from 1-4, and logical partitions are 5
onwards.
START and END are in megabytes
```

```
(parted) resize 2 3133.266 4000.000
```

Важно: Наберитесь терпения! Компьютер работает! Посмотрите на индикатор обращения к жесткому диску (находится на корпусе), чтобы убедиться, что это так. Процесс может занять от 2 до 30 минут.

Когда процесс завершится, перезагрузитесь в вашу старую linux-систему. После этого откройте документ Настольная книга Gentoo: подготовка дисков и следуйте инструкциям. Дойдя до chroot, используйте следующую команду чтобы очистить переменные окружения:

Листинг 6.5: Обновление среды при chroot

```
# env -i HOME=$HOME TERM=$TERM chroot /mnt/gentoo /bin/bash
# /usr/sbin/env-update
# source /etc/profile
```

Удачи!

Полезные советы по установке Gentoo/x86

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/gentoo-x86-tipsntricks.xml>
С версии: 1.0

1. Вступление

Для начала

Этот документ содержит различные полезные советы по установке Gentoo/x86. Большинство из них описаны кратко — подразумевается, что они послужат дополнением к руководству по установке, а не заменой ему.

2. Расширенная установка

Программный RAID

Примечание: Если вы не знакомы с программным RAID, пожалуйста, прочтите Software-RAID-HOWTO (англ.).

Примечание: Более подробное описание установки приведено в руководстве по быстрой установке программного RAID и LVM2 для x86 (англ.).

После загрузки с установочного CD, загрузите соответствующие модули RAID. Например, если вы собираетесь использовать RAID-1:

Листинг 2.1: Загрузка модуля RAID-1
`# modprobe raid1`

Разбивая свои диски, убедитесь, что используете тип раздела fd (Linux raid autodetect), а не 83 (Linux native). Тип раздела можно изменить, используя команду `t` программы fdisk.

Теперь, до начала создания массивов RAID, нам потребуется создать узлы метаустройств:

Листинг 2.2: Создание узлов метаустройств
`# mknod`
`# mknod /dev/md2 b 9 2`
`# mknod /dev/md3 b 9 3d /dev/md1 b 9 1`

После разбивки на разделы, создайте файл `/etc/mdadm.conf` (да, именно так, в среде установочного CD), с помощью `mdadm`, расширенного средства управления RAID. Например, чтобы зеркалировать (RAID-1) разделы `boot`, `swap` и `root`, охватывая `/dev/sda` и `/dev/sdb`, можете использовать:

Листинг 2.3: Создание устройств raid командой mdadm

```
# mdadm --create --verbose /dev/md1 --level=1 --raid-devices=2 /dev/sda1 /dev/sdb1
# mdadm --create --verbose /dev/md2 --level=1 --raid-devices=2 /dev/sda2 /dev/sdb2
# mdadm --create --verbose /dev/md3 --level=1 --raid-devices=2 /dev/sda3 /dev/sdb3
```

Важно: На загрузочном разделе не следует использовать никаких разновидностей чередования (striping), таких как RAID-0 or RAID-5.

Драйвер Linux Software RAID начнет создавать метаустройства. Вы можете проследить за этим в /proc/mdstat. Перед продолжением дождитесь, пока создание метаустройств окончательно завершится..

Листинг 2.4: Сохранение сведений о созданных устройствах

```
# mdadm --detail --scan > /etc/mdadm.conf
```

Теперь и далее используйте /dev/md1 для загрузочного раздела, /dev/md2 для раздела подкачки и /dev/md3 для корневого раздела.

Прямо перед изменением корня (chroot), не забудьте скопировать /etc/mdadm.conf в /mnt/gentoo/etc.

При конфигурации ядра, обязательно включите соответствующую поддержку RAID в состав ядра, а не модулем.

При установке дополнительных утилит, также установите mdadm. Заметьте, что она есть не на всех установочных CD, поэтому у вас может не получиться бессетевая установка Gentoo на программный raid.

При настройке загрузчика не забудьте установить его в MBR обоих дисков, если используется зеркалирование.

ATA RAID с ядрами 2.4

Удостоверьтесь, что вы загрузились с установочного CD с параметром doataraid. После загрузки, проверьте содержимое /dev/ataraid. Там должны находиться различные каталоги disc* для каждого жесткого диска, доступного в ATA RAID. Целый диск показывается как disc, а разделы — как part*.

Выпишите различные файлы устройств /dev/ataraid/disc*/*, на которые будете устанавливать Gentoo. При установке вам потребуется указывать этот путь вместо /dev/hda, указанного в примерах.

Перед изменением корня, свяжите структуру /dev с новой средой:

Листинг 2.5: Связывание /dev

```
# mount -o bind /dev /mnt/gentoo/dev
```

При настройке ядра не забудьте включить поддержку вашего ATA RAID чипсета с нужными параметрами. Например, для популярной системы ATA RAID Promise FastTrack built-in RAID требуется включение в ядро Promise FastTrack Options.

При настройке GRUB сначала потребуется создать загрузочный диск GRUB. Это не так сложно, как кажется. Сначала установите GRUB как обычно, а дойдя до пункта, в котором GRUB устанавливается в MBR, следуйте этим инструкциям:

Листинг 2.6: Создание загрузочного диска GRUB

```
# cd /boot/grub
# dd if=stage1 of=/dev/fd0 bs=512 count=1
# dd if=stage2 of=/dev/fd0 bs=512 seek=1
```

Еще вам потребуется записать файл grub.conf. Здесь нет никаких отличий от установочных инструкций, просто убедитесь, что запись root= указывает на ваше устройство ATA RAID.

После окончания установки, загрузитесь со своего загрузочного диска GRUB. Вы должны увидеть приглашение командной строки GRUB. Теперь настройте GRUB для загрузки с устройства ATA RAID:

Листинг 2.7: Установка GRUB на ATA RAID

```
grub> root (hd0,x)
grub> setup (hd0)
grub> quit
```

Теперь перезагрузитесь (вытащив загрузочную дискету GRUB из дисковода).

Пользователи LILO могут просто следовать указаниям руководства по установке.

Использование ядра с установочного CD

Если вы не хотите компилировать ядро сами, можно взять ядро с установочного компакт-диска и скопировать его в свою систему. Дойдя в процессе инсталляции Gentoo до стадии компиляции ядра, перейдите на другую виртуальную консоль (Alt-F2) и войдите в систему как ROOT, используя пароль, установленный вами в начале установки (passwd root).

Скопируйте ядро и модули в свою систему:

Листинг 2.8: Копирование ядра с установочного CD

```
(${KN}) это название ядра, обычно это что-то вроде 'gentoo' или 'smp'
cdimage ~# cp /mnt/cdrom/isolinux/${KN} /mnt/cdrom/isolinux/${KN}.gz /mnt/gentoo/boot
cdimage ~# mkdir -p /mnt/gentoo/lib/modules
cdimage ~# cp -Rp /lib/modules/`uname -r` /mnt/gentoo/lib/modules
```

Удостоверьтесь в том, что вы установили hotplug (emerge hotplug) и уже добавили его в загрузку. Чтобы все запущенные сейчас модули (с установочного CD) загружались на вашей машине, запустите следующие команды из среды с измененным корнем (chroot):

Листинг 2.9: Добавление всех запущенных модулей в файл modules.conf

```
# cat /proc/modules | cut -d ' ' -f 1 >> \
/etc/modules.autoload.d/kernel-`uname -r` | cut -d . -f -2`
# modules-update
```

3. Упрощение установки

Как оставить терминал без присмотра

Многим хочется отойти от своей системы, пока она компилируется. Иногда это довольно сложно, поскольку установка производится в месте, где много народу и нельзя доверять всем подряд. На этот случай пригодится возможность проводить компиляцию в фоновом режиме, выйдя из всех терминалов.

Есть несколько возможных путей. Первый — использовать screen. После загрузки с LiveCD, установите пароль для root и запустите сеанс screen:

Примечание: screen есть не на всех LiveCD. Если у вас его нет, придется использовать один из других способов, описанных в этом разделе.

Листинг 3.1: Запуск сеанса screen
screen -S gentoo

Из сеанса screen можно хоть провести полную установку. Захотев уйти от терминала, нажмите Ctrl-a, d (то есть control и a одновременно, затем d), чтобы открепить свой сеанс screen. Теперь можно с уверенностью выйти из системы.

Чтобы восстановить доступ к терминалу, опять войдите как root и прикрепитесь к запущенному сеансу screen:

Листинг 3.2: Прикрепление к сеансу screen
screen -x gentoo

Если вы не можете использовать screen, есть другой путь отойти от терминала. Следуйте инструкциям по установке, а дойдя до пункта, в котором запускается длительная компиляция (например, шаг с запуском ./scripts/bootstrap.sh), используйте команду nohup, которая позволит процессу продолжиться, даже если вы выйдете. Не забудьте добавить в конце "&", иначе процесс не будет выполняться в фоновом режиме! Запомните, в каком каталоге вы находитесь (команда pwd покажет ее), так как это вам позже понадобится.

Листинг 3.3: Использование nohup
pwd
/usr/portage
nohup ./scripts/bootstrap.sh &

Теперь выйдите из среды измененного корня (exit) и из сеанса загрузочного CD. Компиляция продолжится в фоновом режиме.

Захотев проверить компиляцию, войдите как root (на установочный CD) и сделайте chroot обратно в свою среду, затем перейдите в оставленный каталог:

Листинг 3.4: Chroot обратно

```
# chroot /mnt/gentoo /bin/bash
# env-update && source /etc/profile
# cd /usr/portage
```

Теперь запустите команду less на файле po4up.out, расположенном внутри каталога. Компиляция добавляет свои сообщения в этот файл, так что при желании следить за ней запустите less po4up.out и нажмите F, чтобы наблюдать за ее ходом. Когда компиляция закончится, можно приступить к следующему пункту указаний по установке.

Если вам надоело следить за изменениями, нажмите Ctrl-C, затем q. Это остановит только процесс less, не затрагивая процесс компиляции.

4. Решение ошибок/проблем

Тщательное тестирование дисков

Если вы считаете, что необходимо тщательно проверить ваш диск на предмет целостности (неисправные секторы и т.д.), можете включить параметр -c при создании на нем файловой системы ext2 или ext3 (используя mke2fs). Это запустит проверку на чтение, которая пометит все неисправные блоки. Если вы настоящий параноик, можете включить -c -c, чтобы провести детальный тест на чтение/запись.

Листинг 4.1: Проверка целостности диска

```
# mke2fs -j -c /dev/hda3
```

Восстановление сбойной установки

Если по какой-то причине ваша установка Gentoo дает сбой, вам не придется повторять ее раз за разом с самого начала. Вместо этого можно спокойно вернуться к моменту, в который вы, как вам кажется, ошиблись (или где, как вы считаете, есть ошибка в инструкции), и попробовать другой подход.

Прежде всего, вам потребуется перейти обратно в свою среду Gentoo Linux командой chroot. Снова следуйте указаниям, пропуская шаги по разбивке диска, так как ваши разделы уже созданы и даже заполнены. Таким образом, вы можете сразу монтировать эти разделы в /mnt/gentoo. Следует также пропустить шаги, связанные с извлечением файла стадии и изменением make.conf — вы же не

хотите перезаписывать существующие файлы, не так ли?

Изменив корень на свою среду Gentoo Linux, сразу переходите к шагу, где, как вам кажется, следует попробовать действовать по-другому. Не повторяйте все шаги, такие как самогенерация, если не считаете, что именно там что-то пошло не так.

Например, если вы считаете, что неверно настроили `grub.conf`, можно сразу запустить свой редактор, чтобы изменить `/boot/grub/grub.conf`.

Попробовав другой подход в своей ситуации, вы, скорее всего, сможете представить, сколько последующих шагов потребуется выполнить снова. Если последующие действия зависели от вашего изменения, их потребуется повторить.

Например:

изменив переменную в `make.conf`, вам потребуется выполнить всю последующую компиляцию, поскольку ее результаты зависят от настройки `make.conf`

изменив `/boot/grub/grub.conf`, можно сразу выходить из среды измененного корня и перезагружаться, так как никакие последующие шаги не зависят от `grub.conf`

перекомпилировав свое ядро, вам нужно лишь убедиться, что конфигурация вашего начального загрузчика указывает на верный образ ядра (убедитесь, что вы смонтировали свой `/boot!`), затем можно выйти из среды измененного корня и перезагрузиться

изменив `/etc/fstab`, можно выходить из среды измененного корня и перезагружаться

Как видите, после большинства действий по восстановлению можно сразу перезагружаться. Лишь изредка вам потребуется повторять последующие шаги установки.

Шпаргалка по установке Gentoo

Ссылка на оригинал: <http://ru.gentoo-wiki.com>

Автор: Molin

С версии: 1.4

Описание

Данное руководство является шпаргалкой по настройке Gentoo для тех, кого утомляет чтение [Настольной книги](#).

Подготовка к установке

Перед установкой необходимо иметь:

- не слишком старый компьютер;
- широкополосный доступ в интернет;
- много времени и терпения.

Прежде чем приступить к установке, следует загрузить на компьютер какую-либо готовую систему...

...при единственном жёстком диске

Взять любой LiveCD с дистрибутивом Linux, поддерживающий настройку сети:

- [Gentoo 2007.0 InstallCD](#)
- [Mandriva Linux One 2008](#)
- [Cool Linux CD](#)
- [Debian Live](#)

Сделать загрузочный CD (DVD, USB-drive), загрузиться, поднять сеть согласно документации к выбранному дистрибутиву.

...при установке на второй жёсткий диск с работающей системой

Включить компьютер и убедиться в работающем интернете.

Условимся, что Gentoo будет устанавливаться на устройство /dev/sda.

Порядок основных действий и команд

Разбиение диска

1. Разметка

```
# fdisk /dev/sda
```

ВАЖНО: Требует дополнительных действий (См. "Логику разметки диска")

2. Форматирование разделов

```
# mkfs -t <тип_ФС> /dev/sda<номер_раздела>  
и т.д.
```

Установка базовой системы

3. Монтирование разделов

```
# mount /dev/sda<номер_раздела> /mnt/gentoo/<путь>
и т.д.

4. Переход в каталог gentoo
# cd /mnt/gentoo

5. Скачивание архива стадии
# wget <сервер>/gentoo/releases/<архитектура>/current/stages/stage3-
<архитектура>-2007.0.tar.bz2

6. Скачивание снимка дерева портежей
# wget <сервер>/gentoo/snapshots/portage-latest.tar.bz2

7. Распаковка архива стадии
# tar xvjpf stage3-i686-2007.0.tar.bz2

8. Распаковка снимка дерева портежей
# tar xvjf /mnt/gentoo/portage-latest.tar.bz2 -C /mnt/gentoo/usr
```

Файлы настройки

9. Правка одного из основных файлов Gentoo

Файл: /mnt/gentoo/etc/make.conf

```
# За подробностями см. make.conf.example
CFLAGS="-O<уровень_оптимизации> -march=<тип_ЦПУ> -pipe"
CXXFLAGS="${CFLAGS}"

# Не изменять
CHOST=<архитектура>

MAKEOPTS="-j<кол-во_процессоров+1>"
GENTOO_MIRRORS="http://distro.ibiblio.org/pub/linux/distributions/gentoo/ "
SYNC="rsync://rsync.gentoo.org/gentoo-portage"

# Для поддержки мыши и клавиатуры в Xorg
INPUT_DEVICES="keyboard mouse"
# Для поддержки видеокарты в Xorg
VIDEO_CARDS=<название_драйвера>

# См. /mnt/gentoo/usr/portage/profiles/use.desc
# См. /mnt/gentoo/usr/portage/profiles/use.local.desc
USE=<флаги>
```

10. Настройка DNS

Файл: /mnt/gentoo/etc/resolv.conf

```
nameserver <IP_сервера_DNS>
nameserver <IP_ещё_одного_сервера_DNS>
```

Или просто скопировать текущую информацию о DNS-серверах
cp -L /etc/resolv.conf /mnt/gentoo/etc/resolv.conf

11. Информация о файловых системах

Файл: /mnt/gentoo/etc/fstab

```
/dev/sda1      /          <файловая_система> defaults    0 1
/dev/sda<номер_раздела> none        swap        sw          0 2
/dev/sda<номер_раздела> <каталог>    <файловая_система> defaults    0 2
proc           /proc      proc        defaults     0 0

# Если /tmp нужен в ОЗУ
none          /tmp       tmpfs       defaults     0 0

# Если CD/DVD монтируется вручную
# Указанное для mod значение снимает флаг исполнения с файлов на CD/DVD
/dev/hd<X> /mnt/cdrom auto user,ro,noauto,uid=1000,gid=1000,mode=100644 0
0

# Если планируется использование DVD-RAM
/dev/hd<X> /mnt/dvdram auto user,rw,noauto 0 0

# Если флоппи кому-то ещё нужен
/dev/fd0      /mnt/floppy auto user,ioccharset=cp1251,noauto,rw,noexec,showexec 0 0
```

12. Имя машины

Файл: /mnt/gentoo/etc/conf.d/hostname

```
HOSTNAME="<название_хоста>"
```

13. Настройка сетевых параметров

Файл: /mnt/gentoo/etc/conf.d/net

```
# Задать параметры вручную...
config_eth0=( " 192.168.0.1 netmask 255.255.255.0 broadcast 192.168.0.255" )
routes_eth0=( "default gw 192.168.0.1" )

# ...или получить их автоматически
config_eth0=( "dhcp" )
dhcp_eth0="nodns nontp nonis"
```

Оживление Gentoo

14. Монтирование /proc

mount -t proc none /mnt/gentoo/proc

15. Монтирование /dev

mount -o bind /dev /mnt/gentoo/dev

16. Смена корневого каталога

```
# chroot /mnt/gentoo /bin/bash
```

ВАЖНО: Дальнейшие команды будут выполняться в среде Gentoo

17. Обновление переменных среды

```
# env-update
```

```
# source /etc/profile
```

18. Смена приглашения командной строки

```
# export PS1="(chroot) $PS1"
```

Сборка ядра

19. Обновление дерева портежей

```
# emerge --sync
```

или

```
# emerge-webrsync
```

20. Установка исходных кодов ядра Gentoo

```
# USE="-doc symlink" emerge gentoo-sources
```

21. Настройка ядра Gentoo

```
# cd /usr/src/linux
```

```
# make menuconfig
```

ВАЖНО: Требуется дополнительных действий (См. "Логика настройки ядра")

22. Компиляция ядра

```
# make && make modules_install
```

23. Установка ядра

```
# cp arch/i386/boot/bzImage /boot/linux-<архитектура>-gentoo
```

Прочие системные установки

24. Сетевое подключение при загрузке системы

```
# rc-update add net.eth0 default
```

25. Пароль суперпользователя

```
# passwd
```

26. Система журналирования

```
# emerge syslog-ng
```

```
# rc-update add syslog-ng default
```

27. Периодические задания (по необходимости)

```
# emerge vixie-cron
```

```
# rc-update add vixie-cron default
```

28. Установка клиента DHCP (по необходимости)

```
# emerge dhcpcd
```

29. Установка клиента PPPoE (по необходимости)

```
# USE="-X" emerge rp-pppoe
```

Настройка загрузчика

30. Установка загрузчика LILO

```
# emerge lilo
```

31. Настройка LILO

Файл: /etc/lilo.conf

```
boot=/dev/sda
prompt
timeout=200
default=gentoo

image=/boot/linux-<архитектура>-gentoo
label=gentoo
read-only
root=/dev/sda<X>
```

32. Записать LILO

```
# /sbin/lilo
```

Локализация в UNICODE

33. Установка шрифтов

```
# emerge terminus-font intlfonts freefonts cronyx-fonts corefonts artwiz-fonts kbd
```

34. Включение нужных локалей

Файл: /etc/locale.gen

```
en_US ISO-8859-1
en_US.UTF-8 UTF-8
ru_RU.CP1251 CP1251
ru_RU.KOI8-R KOI8-R
```

```
ru_RU.UTF-8 UTF-8
```

35. Пересборка glibc
emerge -av glibc

36. Сборка локали
locale-gen

37. Переменные окружения

Файл: /etc/env.d/02locale

```
LC_ALL=""  
LANG="ru_RU.UTF-8"  
LC_NUMERIC="C"
```

38. Файл rc.conf

Файл: /etc/rc.conf

```
UNICODE="yes"
```

39. Шрифт консоли

Файл: /etc/conf.d/consolefont

```
CONSOLEFONT="Cyr_a8x16"
```

40. Карта

Файл: /etc/conf.d/keymaps

```
KEYMAP="-u ru4"  
SET_WINDOWKEYS="yes"  
EXTENDED_KEYMAPS=""  
DUMPKEYS_CHARSET="koi8-r"
```

Загрузка Gentoo

41. Выход из окружения Gentoo
exit

42. Размонтирование разделов
cd /
umount /mnt/gentoo/<путь>
и т.д.

43. Перезагрузка
reboot

Базовая установка Gentoo завершена!
За более тонкой настройкой системы см. документацию.

Вспомогательная информация

Логика разметки диска

Раздел	Каталог	Размер	Размещение	Файловая система
Загрузчик	/boot	~30Мб	в начале диска	ext2, ext3, ReiserFS
Своп	нет	100-200% от объёма ОЗУ	начало диска, но при достаточном объёме ОЗУ и/или работе с не ресурсоёмкими приложениями может не создаваться	Без ФС (swap-область)
Основные системные файлы	/bin, /etc, /lib, /sbin и /sys	0,1-0,5Гб	ближе к началу диска	ext2, ext3, ReiserFS, XFS
Изменяемые данные	/var	Обычно от 0,5Гб, но для Gentoo -- 2-4Гб, т.к. сборка из portage требует много места под временные файлы на /var/tmp	ближе к началу диска	ext3, ReiserFS, XFS
Временные файлы	/tmp	0,5-2Гб (некоторые программы требуют много места)	ближе к началу диска, но раздел может быть создан и в ОЗУ	ext2, JFS, XFS tmpfs
Прикладное ПО	/usr	1-10Гб (зависит от задач)	середина или конец диска	ext2, ReiserFS, XFS, JFS
Несвободное ПО	/opt	от 0,1Гб (в основном для игр и узкоспециализированного ПО)	середина или конец диска	ext2, ReiserFS, XFS, JFS
Данные пользователей	/home	зависит от задач	конец диска	ext3, ReiserFS, XFS, JFS
Данные для чтения из ОС Windows	/mnt/fat, /mnt/win или т.п.	зависит от задач	конец диска	FAT32
ОС Windows	нет	зависит от задач	зависит от задач	NTFS

Логика настройки ядра

По умолчанию многие необходимые опции уже включены.

Выбор семейства ЦПУ

```
Processor type and features --->
Processor family (выбрать ЦПУ) --->
```

Поддержка паравиртуализации

```
Processor type and features --->
[*] Paravirtualization support (EXPERIMENTAL)
```

Поддержка виртуальной машины ядра (KVM)

```
Device Drivers --->
Virtualization --->
<*> Kernel-based Virtual Machine (KVM) support
```

Поддержка дисков SATA

```
Device Drivers --->
<*> Serial ATA (prod) and Parallel ATA (experimental) drivers --->
<*> AHCI SATA support
выбрать нужный SATA драйвер
```

Поддержка PPP

```
Device Drivers --->
Networking device support --->
<*> PPP (point-to-point protocol) support
<*> PPP support for async serial ports
<*> PPP support for sync tty ports
```

Поддержка интерфейса PCI-Express

```
Bus options (PCI, PCMCIA, EISA, MCA, ISA) --->
[*] PCI Express support
```

Поддержка DRM

```
Device Drivers --->
Character devices --->
<*> Direct Rendering Manager (XFree86 4.1.0 and higher DRI support)
```

Поддержка фреймбуфера

```
Device Drivers --->
Graphics support --->
<*> Support for frame buffer devices
выбрать необходимый драйвер
```

Поддержка звука (ALSA)

```
Device Drivers --->
Sound --->
Advanced Linux Sound Architecture --->
<*> Advanced Linux Sound Architecture
<*> OSS Mixer API
<*> OSS PCM (digital audio) API
[*] Verbose procfs contents
```


Поддержка файловых систем

File systems --->

выбрать необходимое

Настольная книга Gentoo Linux x86

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/handbook/handbook-x86.xml>
С версии: 1.0

Содержание:

Установка Gentoo

Из этой части вы узнаете, как устанавливать Gentoo на свою систему.

Об установке Gentoo Linux

В этой главе дается общее представление о предлагаемом порядке установки.

Выбор подходящего источника для установки

Gentoo можно устанавливать по-разному. Здесь описывается, как установить Gentoo с минимального установочного диска. Способ подходит и для установки с универсального установочного диска.

Настройка сети

Чтобы загрузить новейший исходный код, потребуется настройка сети. Здесь описывается порядок ее выполнения.

Подготовка дисков

Чтобы установить Gentoo, нужно создать подходящие дисковые разделы. В этой главе описывается, как разбить диск для будущего использования.

Установка установочных файлов Gentoo

Установка Gentoo выполняется с помощью архива третьей стадии (stage3). В этой главе мы расскажем, как извлечь файл третьей стадии и настроить Portage.

Установка базовой системы Gentoo

После установки и настройки третьей стадии в вашем распоряжении оказывается базовая система Gentoo. Здесь описывается, как этого достичь.

Настройка ядра

Ядро Linux — сердце каждого дистрибутива. В этой главе описывается, как его настроить.

Настройка параметров системы

Вам потребуется отредактировать несколько важных конфигурационных файлов. Из этой главы вы получите представление об этих файлах и поймете, как с ними обращаться.

Установка нужных системных средств

Как отмечено ранее, Gentoo — это богатство выбора. В этой главе мы поможем вам выбрать и установить некоторые важные инструменты.

Настройка начального загрузчика

Для архитектуры x86 существует несколько загрузчиков. Каждый из них настраивается по-своему. В этой главе мы шаг за шагом объясним вам порядок настройки начального загрузчика для ваших нужд.

Завершение установки Gentoo

Установка почти закончена. Осталось создать одну (или несколько) учетных записей для пользователей вашей системы. Как это делается, описано в данной главе.

Чем заняться дальше?

Теперь у вас появилась собственная система Gentoo. Но чем же заняться дальше?

Работа с Gentoo

Здесь вы научитесь работать с Gentoo: устанавливать программное обеспечение, изменять значения переменных, управлять поведением Portage и т.д.

Введение в Portage

В этой главе описываются «простые» шаги, которые вам точно потребуется знать для поддержания в порядке и обслуживания программного обеспечения.

USE-флаги

USE-флаги (признаки использования) — очень важный аспект Gentoo. Прочитав эту главу, вы научитесь обращаться с ними и разберетесь, как USE-флаги влияют на систему.

Возможности Portage

Откройте для себя дополнительные возможности Portage: поддержку распределенной компиляции, кэш компилятора и др.

Сценарии инициализации

В Gentoo используется специальный формат сценариев инициализации (initscript), в котором, например, предусмотрены решения, управляемые зависимостями, и виртуальные сценарии. Здесь описываются эти аспекты, и объясняется, как обращаться со сценариями.

Переменные среды

В Gentoo можно довольно легко управлять системными переменными среды. В этой главе объясняется, как это делать, и описываются часто используемые переменные.

Работа с Portage

В этой части подробно описывается Portage, средство управления программным обеспечением Gentoo.

Файлы и каталоги

Чтобы поближе познакомиться с Portage, вам потребуется узнать, где же хранятся его файлы и данные.

Настройка с помощью переменных

Portage полностью настраивается с помощью различных переменных, которые устанавливаются в конфигурационном файле или как переменные среды.

Смешение ветвей программного обеспечения

Программное обеспечение в составе Gentoo подразделяется на ветви в зависимости от стабильности и поддержки различных архитектур. В этой главе рассказывается, как настраивать использование ветвей, а также как при необходимости преодолевать такое разделение.

Дополнительные средства Portage

В состав Portage входит несколько дополнительных инструментов, которые могут значительно улучшить ваше впечатление от Gentoo. В этой главе раскрываются секреты использования dispatch-conf и других инструментов.

Отступление от официального дерева

Здесь даются советы и показываются приемы «выращивания» собственного дерева Portage, рассказывается о том, как синхронизировать только нужные категории, добавлять свои пакеты и т.д.

Использование ebuild

В этой главе рассказывается о шагах, выполняемых Portage при установке программного обеспечения, и об их самостоятельном выполнении с помощью утилиты ebuild.

Настройка сети в Gentoo

Полное руководство по сетям в Gentoo.

Начальная настройка

Руководство по быстрому запуску и подключению сетевого интерфейса в наиболее распространенных случаях.

Расширенная настройка

Здесь вы изучите, как работает конфигурация — это нужно сделать перед изучением модульного построения сети.

Модульное построение сети

В Gentoo предусмотрены гибкие сетевые средства: здесь рассказывается о выборе различных клиентов DHCP, настройке объединения, образования мостов, виртуальных сетей (VLAN) и т.п.

Беспроводная сеть

Настроить беспроводную сеть не совсем просто. Надеемся, мы поможем вам заставить ее работать.

Дополнительные возможности

Если вы ищете приключений, можете подключить к сети свои собственные функции.

Управление сетью

Для пользователей ноутбуков и тех, кто перемещает свои компьютеры из сети в сеть.

А. Установка Gentoo***1. Об установке Gentoo Linux******1.а. Введение******Добро пожаловать!***

Прежде всего, добро пожаловать в Gentoo! Вы — на пороге мира больших возможностей и высокой производительности! Gentoo весь пропитан идеей свободы выбора. При установке Gentoo вы не раз убедитесь в этом: вам предстоит выбрать объем самостоятельной компиляции, способ установки Gentoo, службу журналирования системы и т.д.

Gentoo — быстрый современный метадистрибутив, обладающий большой чистотой и гибкостью. Gentoo основан на свободном программном обеспечении, и не скрывает от пользователя, «что под капотом». Portage, система управления пакетами Gentoo, написана на языке Python, что позволяет легко просматривать и изменять ее исходный код. Сборка Gentoo также выполняется из исходных текстов (хотя есть и поддержка бинарных пакетов), а настройка Gentoo выполняется с помощью обычных текстовых файлов. Другими словами — сплошная открытость и свобода!

Важно понимать, что Gentoo развивается именно благодаря свободе выбора. Мы стараемся ничего вам не навязывать. А если вам покажется обратное — пожалуйста, сообщите нам об этой оплошности.

Как организована установка?

Установка Gentoo рассматривается как последовательность из 10 шагов, которым соответствуют главы 2—11. Каждый шаг приводит к определенному состоянию:

- **результат шага 1:** вы — в рабочей среде, готовой к установке Gentoo;
- **результат шага 2:** ваше подключение к интернету готово для установки Gentoo;
- **результат шага 3:** ваши жесткие диски готовы стать родным домом для Gentoo;
- **результат шага 4:** подготовлена установочная среда, и вы готовы переключиться (chroot) в новую среду;

- **результат шага 5:** развернуты пакеты ядра, общие для всех систем Gentoo;
- **результат шага 6:** вы скомпилировали собственное ядро Linux;
- **результат шага 7:** вы написали большую часть конфигурационных файлов системы Gentoo;
- **результат шага 8:** установлены необходимые системные средства (которые вы сами выбрали из славного списка);
- **результат шага 9:** установлен и настроен избранный загрузчик ОС, и вы вошли в новую систему Gentoo;
- **результат шага 10:** вы можете начинать изучение своей собственной среды Gentoo Linux!

Мы приложили все усилия, чтобы объяснить вам все «за» и «против» каждого варианта, когда вам предоставляется возможность выбора. При этом один из вариантов помечен как «По умолчанию: ». Другие варианты помечены как «Альтернатива: ». Не подумайте, что вариант по умолчанию — наша рекомендация. Нам просто кажется, что именно его выбирает большинство пользователей.

Иногда есть возможность выполнить необязательный шаг. Такие шаги помечены как «Дополнительно: », и не требуются для установки Gentoo. Однако, некоторые из них будут обусловлены вашими предшествующими решениями. Мы будем сообщать об этом, как в момент выбора, так и непосредственно перед описанием необязательных шагов.

Какие варианты установки существуют?

Gentoo можно установить разными способами. Можно скачать и запустить один из установочных компакт-дисков, установить с имеющегося дистрибутива, с загрузочного CD (например, Knoppix), из сетевой загрузочной среды, с дискеты аварийного восстановления и т.д.

В этом руководстве описывается установка с установочных дисков Gentoo, и, в некоторых случаях, с помощью сетевой загрузки (netboot). Предполагается, что вы собираетесь устанавливать самые свежие версии пакетов. Если вам нужна установка, при которой не требуется использование сети, обратитесь к настольным книгам Gentoo 2006.1 (англ.), где даются указания по установке в бессетевой среде.

Если вы планируете использовать GRP (Gentoo Reference Platform — эталонная платформа Gentoo, набор бинарных пакетов, предназначенных для немедленного использования сразу после установки Gentoo), вам необходимо следовать инструкциям, приведенных в настольной книге Gentoo 2006.1 (англ.).

Чтобы получить сведения о других способах установки, прочитайте описание альтернативных способов установки. Также рекомендуется прочитать полезные советы по установке Gentoo. Если вы почувствуете, что приведенные указания по установке слишком подробны, обратитесь к краткому руководству по установке (см. перечень документации), если, конечно, такое существует для вашей архитектуры.

Кроме того, вы можете выбрать между компиляцией своей системы полностью «с нуля» или установкой заранее собранной среды, позволяющей запустить Gentoo практически моментально. Естественно, есть и промежуточные варианты, в которых вы не компилируете все подряд, а начинаете с полу готовой системы.

Появились затруднения?

Если при установке вы столкнулись с проблемой (или с ошибкой в документации по установке), войдите в нашу систему распределения запросов (англ.) и убедитесь, что такая ошибка еще не заявлена. В этом случае, создайте отчет об ошибке, чтобы мы о ней позаботились. Не бойтесь разработчиков, которым выпадает работа над (вашими) ошибками — людей они обычно не едят.

Обратите внимание, что, хотя руководство, которое вы сейчас читаете, посвящено определенной архитектуре, в нем упоминаются и другие архитектуры. Это связано с тем, что значительная часть текста настольной книги Gentoo является общей для всех архитектур (чтобы не дублировать работу, а также из-за острой нехватки разработчиков). Во избежание путаницы, мы стараемся сократить такие упоминания до минимума.

Если вы не уверены, пользовательская ли у вас ошибка (какую-то ошибку совершили вы, хотя внимательно прочитали документацию), или программная (какую-то ошибку совершили мы, хотя тщательно тестировали установку/документацию), то не стеснясь, заходите на канал **#gentoo** (а также **#gentoo-amd64**, **#gentoo-ru**, **#rugentoo**) сервера **irc.freenode.net**. Разумеется, мы в любом случае будем вам рады :)

Если у вас есть вопрос, касающийся Gentoo, сначала загляните в список распространенных вопросов (FAQ), входящий в состав документации Gentoo. Можно также обратиться к FAQ (англ.) в наших форумах. Если вы не найдете там ответа, задайте вопрос на **#gentoo**, нашем IRC-канале на **irc.freenode.net**. Да-да, кое-кто из нас — маньяки, висящие на IRC :-)

2. Выбор подходящего источника для установки

2.a. Аппаратные требования

Введение

Прежде чем начать, перечислим аппаратные требования, необходимые для успешной установки Gentoo на вашем компьютере.

Аппаратные требования:

- Центральный процессор i486 или новее;
- Память 64 МБ;
- Дисковое пространство 1.5 ГБ (исключая пространство подкачки);

- Пространство подкачки не менее 256 МБ.

2.b. Установочные компакт-диски Gentoo

Введение

Установочные компакт-диски Gentoo — это загрузочные диски, содержащие самодостаточную среду Gentoo. Они позволяют загружать Linux прямо с компакт-диска. При запуске определяются ваши устройства и загружаются соответствующие драйверы. Эти диски сопровождаются разработчиками Gentoo.

Все установочные компакт-диски позволяют загрузиться, настроить сеть, разметить разделы жесткого диска и начать установку Gentoo из интернета. В настоящее время мы выпускаем два установочных компакт-диска, одинаково подходящих для установки Gentoo с загрузкой последних версий существующих пакетов из интернета.

Если вы хотите установить Gentoo без работоспособного соединения с интернетом, или желаете использовать одну из существующих программ установки, пожалуйста, следуйте указаниям по установке, приведенным в настольной книге Gentoo 2006.0 (англ.).

В настоящее время мы выпускаем два установочных компакт-диска: Минимальный установочный диск Gentoo, маленький загрузочный компакт-диск без излишеств, единственное назначение которого — загрузить систему, подготовить подключение к сети и перейти к установке Gentoo. Живой установочный диск Gentoo, содержащий все необходимое для установки Gentoo. На нем есть графическая среда, графическая и консольная программы автоматической установки, и, конечно, указания по установке для вашей архитектуры.

Чтобы помочь вам решить, какой из них нужен, мы привели главные преимущества и недостатки каждого диска.

Минимальный установочный диск Gentoo

Минимальный установочный диск называется `install-x86-minimal-2006.0.iso` и занимает всего 49 МБ. Gentoo можно устанавливать с этого диска только при постоянном наличии работоспособного подключения к интернету. Минимальный установочный диск:

За и против

- + наименьший объем загрузки из сети
- не содержит ни архива `stage3`, ни снимков Portage, ни готовых двоичных пакетов, и поэтому не подходит для установки без сети

Живой (live) установочный диск Gentoo

Живой установочный диск называется `install-x86-universal-2006.0.iso` и занимает 697 МБ. С этого диска вы сможете установить Gentoo даже без исправного подключения к интернету, на случай, если в процессе установки Gentoo на свой компьютер у вас появится неодолимое желание занести его еще и на соседний :) Живой установочный диск Gentoo:

За и против

- + содержит все, что требуется; установку можно выполнять даже без сетевого подключения;
- огромный объем загрузки.

Файл Stage3

Архив третьей стадии — `stage3` — это файл, содержащий минимальную среду Gentoo, пригодную для продолжения установки Gentoo в соответствии с инструкциями, данными в этом руководстве (другими словами: образ (снимок) минимальной системы).

Когда-то в настольной книге Gentoo описывались варианты установки с использованием файла любой из трех существующих стадий создания системы. Несмотря на то, что в Gentoo до сих пор представлены файлы `stage1` и `stage2`, в официальном способе установки используется архив `stage3`. Если вас интересует установка Gentoo из файлов `stage1` или `stage2`, пожалуйста, обратитесь к сборнику распространенных вопросов по Gentoo, раздел как установить Gentoo, используя файлы Stage1 или Stage2?

Архивы **stage3** можно загрузить из `releases/x86/2007.0/stages/` или с любого из официальных зеркал Gentoo; на «живом диске» они не поставляются.

2.с. Скачивание, запись и загрузка с установочного диска Gentoo

Скачивание образа и запись на диск

Итак, вы выбрали нужный установочный диск Gentoo. Начнем с загрузки его образа из сети, и его записи на компакт-диск. Ранее обсуждались варианты доступных дисков, но где же их взять?

Любой из установочных дисков (а при желании — заодно и диск пакетов, если есть) можно загрузить с одного из наших зеркал. Установочные диски расположены в каталоге `releases/x86/2007.0/installcd`.

В этом каталоге находятся ISO-файлы. Это полные образы компакт-дисков, которые можно записать на CD-R.

Если вас волнует, не искажен ли загруженный файл, можно сверить его контрольную сумму MD5 с рассчитанной нами (например, `install-x86-minimal-2007.0.iso.DIGESTS`). Вычислить контрольную сумму MD5 можно утилитой `md5sum` в Linux/Unix, или программой `md5sum` (англ.) для Windows.

Другой способ убедиться в пригодности загруженного файла — с помощью GnuPG сверить криптографическую подпись с той, что хранится у нас (в файле с расширением .asc). Скачайте файл подписи, затем получите открытый ключ:

Листинг 1: Получение открытого ключа

```
$ gpg --keyserver subkeys.gpg.net --recv-keys 17072058
```

Теперь сверьте подпись:

Листинг 2: Сверка криптографической подписи

```
$ gpg --verify <файл подписи> <скачанный образ диска>
```

Записывать загруженный ISO-образ на компакт-диск нужно в «сыром» режиме. Как это сделать, сильно зависит от используемой программы. Здесь мы обсудим запись с помощью cdrecord и K3B. Дополнительные сведения находятся в нашем сборнике распространенных вопросов.

При использовании cdrecord, просто введите `cdrecord dev=/dev/hdc <скачанный iso-файл>` (/dev/hdc замените на путь к вашему устройству CD-RW).

При использования K3B, выберите Tools > CD > Burn Image (Инструменты > Компакт-диск > Прожечь образ). Затем в области 'Image to Burn' ('Образ для записи') укажите свой ISO-файл. В завершение нажмите Start (Запуск).

Загрузка с установочного компакт-диска

Важно: Перед тем, как приступить, полностью прочитайте этот подраздел, так как перед последующими действиями вам, скорее всего, больше не представится такая возможность.

Записав установочный компакт-диск, пора его загрузить. Уберите все компакт-диски из своих приводов CD, перезагрузите компьютер и войдите в BIOS. В зависимости от BIOS, для этого обычно нужно нажать DEL, F1 или ESC. В BIOS измените порядок загрузки так, чтобы обращение к CD-ROM выполнялось до обращения к жесткому диску. Этот параметр часто задается в разделе «CMOS Setup». Если порядок загрузки не изменить, система просто перезагрузится с жесткого диска, игнорируя CD-ROM.

Теперь поместите установочный диск в привод CD-ROM (наконец-то!) и перезагрузитесь. Должно появиться загрузочное приглашение. Здесь можно просто нажать ENTER, чтобы запустить процесс загрузки с параметрами по умолчанию, или загрузить установочный диск по-своему: указать ядро, потом загрузочные параметры, а затем нажать ENTER.

Указать ядро? Да, на нашем установочном компакт-диске есть несколько вариантов ядра. По умолчанию используется **gentoo**. Другие ядра предназначены для специфических аппаратных задач; вариант **-nofb** отключает кадровый буфер.

Ниже перечислены доступные варианты ядра:

Ядро	Описание
gentoo	ядро 2.6 с многопроцессорной поддержкой (используется по умолчанию)
gentoo-nofb memtest86	аналогично gentoo, но без поддержки кадрового буфера для выявления ошибок локальной оперативной памяти

Также можно указать параметры ядра, явно включающие или отключающие определенные режимы. Приведенный список соответствует тому, что выводится при нажатия F2 в экране загрузки.

Листинг 3: Параметры, которые можно сообщить выбранному ядру

- agpgart	загрузка agpgart (используйте при сбоях графики, зависаниях)
- acpi=on	включение поддержки ACPI
- ide=nodma	принудительное отключение DMA для сбойных IDE-устройств
- doscsi	поиск scsi-устройств (нарушает работу некоторых ethernet-плат)
- dopcmcia	запуск службы pcmcia для PCMCIA-приводов компакт-дисков
- nofirewire	отключение в initrd модулей firewire (предназначенных для приводов компакт-дисков с подключением firewire и т.п.)
- nokeymap	отключение выбора неамериканских раскладок клавиатуры
- docache	кэширование исполняемой части компакт-диска в памяти, позволяет размонтировать /mnt/cdrom для установки другого диска
- nodetect	запрет запуска hwsetup/kudzu и hotplug
- nousb	отключение загрузки модулей usb в initrd, отключает hotplug
- nodhcp	отмена автоматического запуска dhcp при обнаружении сетевой платы
- nohotplug	отключение загрузки службы hotplug
- noapic	отключение apic (попробуйте, если есть аппаратные проблемы с устройствами scsi, сетевыми платами и т.д.)
- noevms	отключение поддержки модулей EVMS2
- noilm2	отключение поддержки модулей LVM2
- hdx=stroke	позволяет использовать жесткий диск целиком, даже если BIOS не поддерживает диски большого размера
- noload=module1,[module2,...]	запрет загрузки определенных модулей ядра

Теперь загрузите систему с компакт-диска, выбрав ядро (если вас не устраивает ядро по умолчанию gentoo) и указав дополнительные параметры. В качестве примера мы покажем, как загрузить ядро gentoo с параметром dopcmcia:

Листинг 4: Загрузка установочного диска

boot: gentoo dopcmcia

Появится заставка с полосой индикатора загрузки. Если вы устанавливаете Gentoo на машину с не американской раскладкой клавиатуры, нужно немедленно нажать ALT+F1, чтобы переключиться в подробный режим, и следовать появившимся указаниям. Если ничего не выбрано в течение 10 секунд, устанавливается раскладка по умолчанию (клавиатура США), а загрузка продолжается. По окончании загрузки для вас будет выполнен автоматический вход в «живой» Gentoo Linux в качестве «root» (администратора). В текущей консоли должно появиться приглашение root («#»). Также можно переключаться в

другие консоли, нажимая ALT-F2, ALT-F3 и ALT-F4. Вернуться в первоначальную консоль можно нажатием ALT-F1.

Приступим к дополнительной настройке оборудования.

Дополнительная настройка оборудования

При загрузке с установочного компакт-диска система пытается определить все устройства и загрузить соответствующие модули для их поддержки. В подавляющем большинстве случаев она очень хорошо справляется с этим. Однако, в редких случаях некоторые из нужных модулей не загружаются автоматически. Если какие-либо устройства пропущены при автоматическом опросе шины PCI, модули ядра для их поддержки вам придется загрузить вручную.

В следующем примере мы попробуем загрузить модуль **8139too** (для поддержки некоторых типов сетевого интерфейса):

Листинг 5: Загрузка модуля ядра
modprobe 8139too

Если вам нужна поддержка PCMCIA, требуется запустить сценарий инициализации pcmcia:

Листинг 6: Запуск сценария инициализации PCMCIA
/etc/init.d/pcmcia start

Дополнительно: оптимизация скорости жесткого диска

Если вы опытный пользователь, возможно, вам захочется настроить скоростные параметры жесткого диска IDE с помощью программы **hdparm**. Указав параметр **-tT**, можно измерить скорость вашего диска (для уточнения значений запускайте программу несколько раз подряд):

Листинг 7: Тестирование производительности диска
hdparm -tT /dev/hda

Для настройки вы можете использовать любой из приведенных примеров (или поэкспериментировать самостоятельно), при необходимости указав название своего диска вместо **/dev/hda**:

Листинг 8: Оптимизация скорости жесткого диска
Включение DMA:
hdparm -d 1 /dev/hda
Установка безопасных значений:
hdparm -d 1 -A 1 -m 16 -u 1 -a 64 /dev/hda

Дополнительно: учетные записи пользователей

Если вы собираетесь дать другим людям доступ к своей системе, или намерены входить в чат irssi без прав администратора (из соображений безопасности), потребуется создать учетные записи пользователей и изменить пароль администратора.

Для изменения пароля администратора используйте **passwd**:

Листинг 9: Изменение пароля администратора

```
# passwd
New password: (введите новый пароль)
Re-enter password: (введите новый пароль еще раз)
```

Для создания учетной записи пользователя сначала задаются его реквизиты, а затем вводится пароль. Для этого служат программы **useradd** и **passwd**. В следующем примере мы создадим пользователя «john»:

Листинг 10: Создание учетной записи пользователя

```
# useradd -m -G users john
# passwd john
New password: (введите новый пароль для john)
Re-enter password: (введите новый пароль Джона еще раз)
```

Вы также можете изменить свой идентификатор пользователя с администратора (root) на вновь созданного пользователя командой **su**:

Листинг 11: Изменение идентификатора пользователя

```
# su - john
```

Дополнительно: просмотр документации во время установки

Если вам нужна возможность просмотра настольной книги Gentoo в процессе установки (не важно, из сети или с компакт-диска), убедитесь, что вы создали учетную запись обычного пользователя (см. дополнительно: учетные записи пользователей). Затем нажмите ALT+F2 для перехода в другой терминал, и войдите в систему.

Для чтения документации с компакт-диска, вы можете сразу запустить **links**:

Листинг 12: Просмотр документации с компакт-диска

```
# links /mnt/cdrom/docs/html/index.html
```

Но лучше открывать настольную книгу Gentoo из сети (там она новее, чем на компакт-диске). Для этого тоже можно пользоваться **links**, но только после настройки сети (иначе вы не сможете выйти в интернет, чтобы прочесть

документ):

Листинг 13: Просмотр документации, находящейся в сети
links <http://www.gentoo.org/doc/ru/handbook/handbook-x86.xml>

На исходный терминал можно переключаться нажатием **ALT+F1**.

Дополнительно: запуск демона SSH

Если вы хотите разрешить другим пользователям доступ к своей системе в процессе установки Gentoo (возможно затем, чтобы они помогли вам с установкой или даже провели ее за вас), для них потребуется создать учетные записи, а возможно, даже дать им пароль администратора (делайте это только в том случае, если вы им полностью доверяете).

Чтобы запустить демон SSH, выполните следующую команду:

Листинг 14: Запуск демона SSH
/etc/init.d/sshd start

Для использования sshd, сначала требуется настроить сеть. Переходите к главе настройка сети.

3. Настройка сети

3.а. Автоматическое подключение к сети

Может, она уже настроена?

Если ваша система подсоединена к сети Ethernet, в которой есть сервер DHCP, очень вероятно, что сетевое подключение на вашей машине уже автоматически настроено. Если так, вы сможете воспользоваться многими сетевыми командами, находящимися на установочном компакт-диске, например, **ssh**, **scp**, **ping**, **irssi**, **wget** и **links**.

Если сеть уже настроена, команда **/sbin/ifconfig** должна показывать сетевые интерфейсы кроме **lo**, например, **eth0**:

Листинг 1: /sbin/ifconfig для рабочей сетевой конфигурации
/sbin/ifconfig
(...)
eth0 Link encap:Ethernet HWaddr 00:50:BA:8F:61:7A
inet addr:192.168.0.2 Bcast:192.168.0.255 Mask:255.255.255.0
inet6 addr: fe80::50:ba8f:617a/10 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:1498792 errors:0 dropped:0 overruns:0 frame:0
TX packets:1284980 errors:0 dropped:0 overruns:0 carrier:0
collisions:1984 txqueuelen:100

RX bytes:485691215 (463.1 Mb) TX bytes:123951388 (118.2 Mb)
 Interrupt:11 Base address:0xe800

Дополнительно: указание прокси-серверов

Если вы подключены к интернету через прокси-сервер, при установке вам может потребоваться ввести сведения о нем. Задать прокси-сервер очень просто: нужно определить переменные, содержащие необходимые сведения.

В большинстве случаев в определении переменных достаточно указать имя прокси-сервера. Предположим, например, что прокси-сервер называется proxy.gentoo.org, а его порт — 8080:

Листинг 2: Указание прокси-сервера

```
(если прокси-сервер фильтрует трафик HTTP)
# export http_proxy="http://proxy.gentoo.org:8080"
(если прокси-сервер фильтрует трафик FTP)
# export ftp_proxy="ftp://proxy.gentoo.org:8080"
(если прокси-сервер фильтрует трафик RSYNC)
# export RSYNC_PROXY="proxy.gentoo.org:8080"
```

Если прокси-сервер запрашивает имя пользователя и пароль, для значения переменной следует использовать такой формат:

Листинг 3: Указание имени/пароля в адресе прокси-сервера
 http://имя_пользователя:пароль@proxy.gentoo.org:8080

Проверка сети

Может оказаться полезным проверить отклик DNS-сервера вашего поставщика услуг интернета (адрес сервера находится в `/etc/resolv.conf`), а также произвольного веб-сайта, чтобы убедиться, что ваши пакеты выходят в интернет, разрешение имен DNS работает и т.д.

Листинг 4: Проверка доступности сети
 # ping -c 3 google.com

Сеть доступна? Тогда пропустите остаток этого раздела, и переходите к разделу подготовка дисков. Если сеть все же недоступна, то, к сожалению, вам придется еще поработать над ее настройкой.

3.b. Автоматизированная настройка сети

Если сеть не заработает сразу, при установке с некоторых носителей можно использовать **net-setup** (для обычных или беспроводных сетей), **pppoe-setup** (для пользователей ADSL) или **pptp** (для пользователей PPTP; есть на x86, amd64,

alpha, ppc и ppc64).

Если на вашем установочном диске нет ни одного из этих средств, или сеть еще не подключена, приступайте к ручной настройке сети:

- пользователи обычной сети Ethernet — переходите к разделу по умолчанию: использование **net-setup**;
- пользователи **ADSL** — переходите к разделу альтернатива: использование **RP-PPPoE**;
- пользователи **PPTP** — переходите к разделу альтернатива: использование **PPTP**;

По умолчанию: использование net-setup

Простейший способ настройки сети, если она не произошла автоматически — запуск сценария **net-setup**:

Листинг 5: Запуск сценария net-setup
net-setup eth0

net-setup задаст вам несколько вопросов о вашей сетевой среде. В результате его работы у вас должно появиться работоспособное подключение к сети. Проверьте сетевое подключение, как это описано выше. Если проверка прошла успешно, примите наши поздравления — теперь вы готовы к установке Gentoo. Пропустите оставшуюся часть этого раздела и приступайте к подготовке дисков.

Если ваша сеть все еще не заработала, переходите к ручной настройке сети.

Альтернатива: использование RP-PPPoE

Для простоты подключения к интернету по **PPPoE**, в установочный диск (любой версии) включен **gr-pppoe**. Для настройки соединения используйте сценарий **pppoe-setup**, входящий в комплект. У вас будет запрошена информация о сетевом устройстве ethernet, подключенном к **adsl**-модему, имя пользователя, пароль, IP-адреса ваших серверов DNS. Также задается вопрос, нужно ли включать базовый межсетевой экран (**firewall**).

Листинг 6: Использование gr-pppoe
pppoe-setup
pppoe-start

Если что-то пошло не так, проверьте, правильно ли вы ввели имя пользователя и пароль, посмотрев в **/etc/ppp/pap-secrets** или **/etc/ppp/chap-secrets**, и убедитесь, что устройство ethernet указано верно. Если ваше устройство не видно в системе, потребуется загрузить соответствующие сетевые модули. Для этого нужно перейти к разделу ручная настройка сети где мы объясняем, как их загрузить.

Если же все заработало, переходите к подготовке дисков.

Альтернатива: использование PPTP

Примечание: Поддержка **PPTP** имеется только для архитектуры x86.

Если вам нужна поддержка PPTP, можно использовать **pptpclient**, который входит в состав нашего установочного диска. Но сначала нужно обеспечить правильность настройки. Отредактируйте файлы **/etc/ppp/pap-secrets** или **/etc/ppp/chap-secrets** так, чтобы в них находилось правильное сочетание имени пользователя и пароля.

Листинг 7: Редактирование **/etc/ppp/chap-secrets**
nano -w /etc/ppp/chap-secrets

Затем, если нужно, измените параметры PPTP в файле **/etc/ppp/options.pptp**:

Листинг 8: Редактирование **/etc/ppp/options.pptp**
nano -w /etc/ppp/options.pptp

Когда все будет готово, просто запустите **pptp** (с параметрами, которые вы не стали прописывать в **options.pptp**), чтобы соединиться с сервером:

Листинг 9: Подключение к серверу коммутируемого доступа
pptp <server ip>

Теперь переходите к подготовке дисков.

3.с. Ручная настройка сети

Загрузка нужных сетевых модулей

При загрузке установочный диск пытается выявить все установленные устройства и загружает подходящие модули ядра (драйверы) для поддержки вашего оборудования. В подавляющем большинстве случаев он очень хорошо справляется с этой работой. Однако, в некоторых случаях он может не справиться с автозагрузкой нужных вам модулей ядра.

Если **net-setup** или **pppoe-setup** не удалось загрузить нужный модуль, возможно, ваша сетевая плата сразу не обнаружена. При этом вам может потребоваться ручная загрузка необходимых модулей ядра.

Чтобы выяснить, какие модули ядра для поддержки сети существуют, используйте **ls**:

Листинг 10: Поиск имеющихся модулей
ls /lib/modules/`uname -r`/kernel/drivers/net

Если вы найдете драйвер для своей сетевой платы, для загрузки модуля ядра используйте **modprobe**:

Листинг 11: Использование `modprobe` для загрузки модуля ядра
(для примера загрузим модуль `pcnet32`)
modprobe pcnet32

Чтобы убедиться, что ваша сетевая плата теперь обнаружена, используйте **ifconfig**. Если сетевая плата обнаружена, результат выглядит подобным образом:

Листинг 12: Проверка доступности сетевой платы (успешная)
ifconfig eth0
eth0 Link encap:Ethernet HWaddr FE:FD:00:00:00:00
BROADCAST NOARP MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

Однако, если вы получите такое сообщение об ошибке, сетевая плата не обнаружена:

Листинг 13: Проверка доступности сетевой платы (неуспешная)
ifconfig eth0
eth0: error fetching interface information: Device not found

Если в вашей системе установлено несколько сетевых плат, они будут называться **eth0**, **eth1** и т.д. Убедитесь, что сетевая плата, которую вы собираетесь использовать, работает хорошо, и в дальнейшем не забудьте везде подставлять верное имя. Мы предполагаем, что используется сетевая плата **eth0**.

Когда ваша сетевая плата обнаружена, можно попробовать снова запустить **net-setup** или **pppoe-setup** (которые теперь должны сработать), но на случай, если вы из крутых, мы опишем, как настроить сеть вручную.

Выберите один из следующих разделов, в зависимости от необходимого вида настройки:

- [использование DHCP для автоматического присвоения IP-адреса;](#)
- [подготовка беспроводного доступа, если у вас есть беспроводная плата;](#)
- [освоение сетевой терминологии: рассказ о том, что нужно знать о подключении к сети;](#)
- [использование ifconfig и route: описание ручной настройки сети;](#)

Использование DHCP

DHCP (Dynamic Host Configuration Protocol — протокол динамической настройки хоста) дает возможность автоматически получить параметры сетевого подключения (IP-адрес, маску сети, широковещательный адрес, шлюз, сервера имен и т. д.) Все это работает, только когда в вашей сети есть сервер DHCP (или ваш поставщик предоставляет услугу DHCP). Чтобы сетевой интерфейс получал эти сведения автоматически, используйте **dhcpcd**:

Листинг 14: Использование dhcpcd**# dhcpcd eth0**

Некоторые сетевые администраторы требуют, чтобы вы использовали имя хоста и домена, назначенное сервером DHCP.

В этом случае используйте

dhcpcd -HD eth0

Если это сработало (попробуйте «попинговать» какой-нибудь сервер интернета, например, Google), то у вас все настроено, и можно двигаться дальше. Пропустите остаток этого раздела и приступайте к подготовке дисков.

Подготовка беспроводного доступа

Примечание: Поддержка команды **iwconfig** есть только на установочных дисках для платформ x86, amd64 и ppc. В других случаях вы все же сможете запустить расширения, следуя инструкциям проекта [linux-wlan-ng](#) (англ.).

Если вы используете беспроводную плату (802.11), перед дальнейшими действиями может потребоваться настройка параметров беспроводного подключения. Для просмотра текущей настройки беспроводной платы можете использовать **iwconfig**. При запуске **iwconfig** вы увидите подобные сведения:

Листинг 15: Вывод текущих параметров беспроводного подключения**# iwconfig eth0**

```
eth0 IEEE 802.11-DS ESSID:"GentooNode"
      Mode:Managed Frequency:2.442GHz Access Point: 00:09:5B:11:CC:F2
      Bit Rate:11Mb/s Tx-Power=20 dBm Sensitivity=0/65535
      Retry limit:16 RTS thr:off Fragment thr:off
      Power Management:off
      Link Quality:25/10 Signal level:-51 dBm Noise level:-102 dBm
      Rx invalid nwid:5901 Rx invalid crypt:0 Rx invalid frag:0 Tx
      excessive retries:237 Invalid misc:350282 Missed beacon:84
```

Примечание: Некоторые беспроводные платы могут называться подобно wlan0 или ra0, а не eth0. Для определения верного имени устройства запускайте **iwconfig** без каких-либо параметров командной строки.

Для большинства пользователей может потребоваться изменение только двух параметров: ESSID (названия беспроводной сети) и ключа шифрования WEP. Если ESSID и указанный адрес точки доступа уже принадлежат вашей точке доступа, и вы не используете WEP, значит, ваше беспроводное подключение работает. Если вам необходимо изменить свой ESSID или добавить ключ WEP, можно запустить следующие команды:

Листинг 16: Замена ESSID и/или добавление ключа WEP

(так название сети устанавливается в "GentooNode")

iwconfig eth0 essid GentooNode

(так устанавливается шестнадцатиричный ключ WEP)

```
# iwconfig eth0 key 1234123412341234abcd
```

(так устанавливается текстовый ключ (ASCII); вначале нужно добавлять "s:")

```
# iwconfig eth0 key s:some-password
```

Проверить сделанную настройку можно, снова запустив **iwconfig**. Как только ваша беспроводная сеть заработает, вы можете перейти к установке параметров сети, относящихся к протоколу IP, которые описываются в следующем разделе (освоение сетевой терминологии), или использовать утилиту **net-setup**, как описано ранее.

Освоение сетевой терминологии

Примечание: Если вы знаете свой IP-адрес, широковещательный адрес, маску сети и серверы имен, можете пропустить этот подраздел и перейти к разделу использование **ifconfig** и **route**.

Если все, рассказанное выше, не помогло, вам придется настроить свою сеть вручную. Это совсем нетрудно. Однако, вам понадобится освоить кое-какую сетевую терминологию, знание которой требуется для удовлетворительной настройки сети. Прочитав этот текст, вы узнаете, что такое шлюз, зачем служит маска сети, как формируется широковещательный адрес, и зачем нужны серверы имен.

В сети узлы (хосты, компьютеры) различаются по IP-адресу (адресу протокола интернета). Такой адрес — это сочетание четырех чисел от 0 до 255. Ну, по крайней мере, так мы его воспринимаем. В действительности, IP-адрес состоит из 32 бит (единиц и нулей). Давайте рассмотрим пример:

Листинг 17: Пример IP-адреса

```
IP-адрес (числа): 192.168.0.2
IP-адрес (биты): 11000000 10101000 00000000 00000010
                -----
                192   168   0   2
```

Такой IP-адрес уникален для узла в рамках всех доступных сетей (т. е. каждый узел, с которым вы можете связаться, должен иметь уникальный IP-адрес). Чтобы различать узлы, находящиеся внутри и извне сети, IP-адрес подразделяется на две части: часть сети и часть узла.

Это разделение записывается с помощью маски сети — набора единиц, за которым следует набор нулей. Часть IP-адреса, которая попадает на единицы — сетевая, оставшаяся часть — узловая. Как обычно, маска сети может записываться в виде IP-адреса.

Листинг 18: Пример разделения сети/узла

```
IP-адрес: 192 168 0 2
          11000000 10101000 00000000 00000010
Маска сети: 11111111 11111111 11111111 00000000
            255 255 255 0
            +-----+-----+
```

Сеть Узел

Другими словами, 192.168.0.14 — все еще входит в состав сети из нашего примера, а 192.168.1.2 — уже нет.

Широковещательный адрес — это IP-адрес с такой же сетевой частью, как у вашей сети, но у которого узловая часть состоит только из единиц. Каждый узел вашей сети слушает этот IP-адрес. Он действительно предназначен для широковещательной рассылки пакетов.

Листинг 19: Широковещательный адрес

```
IP-адрес:      192   168   0   2
               11000000 10101000 00000000 00000010
Широковещательный 11000000 10101000 00000000 11111111
адрес:         192   168   0   255
               +-----+-----+
               Сеть      Узел
```

Чтобы бороздить просторы интернета, вы должны знать, через какой узел происходит подключение к интернету. Этот узел называется шлюзом. Так как это обычный узел, ему присвоен обычный IP-адрес (например, 192.168.0.1).

Выше мы говорили, что каждому узлу присваивается свой собственный IP-адрес. Чтобы связываться с узлом по имени (вместо IP-адреса), нужна служба, которая переводит имя (такое, как `dev.gentoo.org`) в IP-адрес (например, 64.5.62.82). Такая служба называется службой имен. Чтобы пользоваться ей, нужно указать необходимые серверы имен в `/etc/resolv.conf`.

Иногда ваш шлюз сам является сервером имен. В остальных случаях вам необходимо указывать серверы имен, предоставляемые поставщиком интернета.

В итоге, для дальнейшего вам потребуется следующая информация:

Параметр настройки сети	Пример
Ваш IP-адрес	192.168.0.2
Маска сети	255.255.255.0
Широковещательный адрес	192.168.0.255
Шлюз	192.168.0.1
Сервер(ы) имен	195.130.130.5, 195.130.130.133

Использование `ifconfig` и `route`

Настройка вашей сети состоит из трех шагов. Сначала мы назначаем себе IP-адрес с помощью **`ifconfig`**. Затем мы настраиваем маршрутизацию к шлюзу, пользуясь **`route`**. И в завершение мы помещаем IP-адреса серверов имен в `/etc/resolv.conf`.

Для назначения IP-адреса потребуется ваш IP-адрес, широковещательный адрес

и маска сети. Узнав их, выполните следующую команду, заменив `${IP_ADDR}` на свой IP-адрес, `${BROADCAST}` на свой широковещательный адрес, а `${NETMASK}` на свою маску сети:

Листинг 20: Использование `ifconfig`

```
# ifconfig eth0 ${IP_ADDR} broadcast ${BROADCAST} netmask ${NETMASK} up
```

Теперь настройте маршрутизацию с помощью **route**. Подставьте IP-адрес своего шлюза вместо `${GATEWAY}`:

Листинг 21: Использование `route`

```
# route add default gw ${GATEWAY}
```

Затем откройте `/etc/resolv.conf` в своем любимом редакторе (в нашем примере используется `nano`):

Листинг 22: Создание `/etc/resolv.conf`

```
# nano -w /etc/resolv.conf
```

Заполните данные своих серверов имен по следующему образцу. Обязательно замените `${NAMESERVER1}` и `${NAMESERVER2}` на соответствующие адреса серверов имен:

Листинг 23: Образец `/etc/resolv.conf`

```
nameserver ${NAMESERVER1}
nameserver ${NAMESERVER2}
```

Готово. Теперь проверьте свою сеть, «попингав» какой-либо сервер интернета (например, Google). Если все заработало, примите наши поздравления! Теперь вы готовы к установке Gentoo. Приступайте к подготовке дисков.

4. Подготовка дисков

4.а. Общие сведения о блочных устройствах

Блочные устройства

Мы достаточно подробно рассмотрим аспекты работы с дисками в Gentoo Linux и Linux вообще, включая файловые системы, разделы и блочные устройства. Позже, когда вы уже освоитесь с дисками и файловыми системами, мы проведем вас через процесс настройки разделов и файловых систем для установки Gentoo Linux.

Для начала, давайте познакомимся с блочными устройствами. Вероятно, наиболее известно блочное устройство, соответствующее в системе Linux

первому IDE-диску, а именно `/dev/hda`. А если в вашей системе используются диски SCSI или SATA, то первым жестким диском будет `/dev/sda`.

Подобные блочные устройства представляют собой абстрактный интерфейс к диску. Прикладные программы могут использовать их для обращения к дискам, не беспокоясь, к какому типу те принадлежат: IDE, SCSI или какому-то еще. Программы могут просто обращаться к накопителю, как к набору смежных 512-байтных блоков с произвольным доступом.

Разделы

Хотя теоретически для размещения системы Linux диск можно использовать целиком, так почти никогда не делают на практике. Вместо этого большое физическое блочное устройство разбивают на меньшие блочные устройства, более удобные для обращения. В архитектуре x86 они называются разделами.

Существуют разделы трех типов: первичные (**primary**), расширенные (**extended**) и логические (**logical**).

Первичный раздел — это раздел, информация о котором хранится в **MBR** (Master Boot Record — главной загрузочной записи). Так как MBR очень мала (512 байт), можно определять всего четыре первичных раздела (например, от `/dev/hda1` до `/dev/hda4`).

Расширенный раздел — это особый первичный раздел (имеется в виду, что расширенный раздел должен быть одним из четырех возможных первичных разделов), в котором содержатся другие разделы. Таких разделов изначально не существовало, но их введение помогло расширить существующую схему разметки без потери совместимости, как только четырех разделов перестало хватать.

Логический раздел — это раздел, входящий в расширенный раздел. Его определение находится не в **MBR**, а внутри расширенного раздела.

Дополнительные возможности

На установочном компакт-диске для x86 предусмотрена поддержка EVMS и LVM2. Использование EVMS и LVM2 повышает гибкость разбиения диска. В ходе установки мы останавливаемся на «обычных» разделах, но стоит запомнить, что EVMS и LVM2 тоже поддерживаются.

4.b. Разработка схемы разбиения диска

Схема разбиения по умолчанию

Если вам не интересно заниматься разработкой схемы для своей системы, можно воспользоваться схемой, используемой в этой книге:

Раздел	Файловая система	Размер	Описание
/dev/hda1	ext2	32МБ	загрузочный раздел
/dev/hda2	(swap)	512МБ	раздел подкачки
/dev/hda3	ext3	оставшаяся часть диска корневой раздел	

Если вам интересно узнать, какого размера должны быть разделы, и сколько их вам вообще может потребоваться, читайте дальше. В противном случае приступайте к созданию разделов, описанному в главе использование **fdisk** для создания разделов.

Сколько и каких именно?

Количество разделов очень сильно зависит от назначения системы. Например, если у вас много пользователей, вам, скорее всего, захочется отделить /home для повышения безопасности и упрощения резервного копирования. Если вы устанавливаете Gentoo в роли почтового сервера, следует отделить /var, т.к. вся почта хранится там. Затем, правильно выбрав файловую систему, вы добьетесь максимальной производительности. Для игровых серверов потребуется отдельный раздел /opt, так как большинство программ для их работы устанавливается туда. Причины выделения те же, что и для /home: безопасность и резервное копирование. Определенно не помешает побольше места для /usr: не только потому, что там хранится большинство приложений, а еще из-за того, что лишь дерево Portage, не считая размещенных в нем архивов с исходными кодами, занимает около 500 МБ.

Как видите, все зависит от ваших целей. Наличие отдельных разделов или томов имеет следующие плюсы:

- для каждого раздела или тома можно выбрать наиболее подходящую файловую систему;
- свободное место во всей системе не кончится «вдруг» из-за того, что одна-единственная сбойная программа постоянно записывает файлы в раздел или том
- необходимая проверка файловых систем будет занимать меньше времени, т.к. проверка разных разделов может выполняться параллельно (еще больший выигрыш времени дает использование нескольких физических дисков) ;
- можно повысить безопасность системы, монтируя часть разделов в режиме read-only (только для чтения), **nosuid** (игнорируется бит setuid), noexec (игнорируется бит исполнения) и т.д..

Однако, у создания множества разделов есть один большой минус: при неправильной настройке можно получить систему, в которой много свободного места на одном разделе, и совершенно нет на другом. Кроме того, на дисках SCSI и SATA возможно создание не более 15 разделов.

Для примера мы покажем разбиение диска объемом 20 ГБ, используемого в демонстрационном ноутбуке (с веб-сервером, почтовым сервером, средой Gnome и т.д.):

Листинг 1: Пример файловой системы

```
$ df -h
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/hda5       ext3   509M  132M  351M  28%  /
/dev/hda2       ext3   5.0G   3.0G   1.8G  63%  /home
/dev/hda7       ext3   7.9G   6.2G   1.3G  83%  /usr
/dev/hda8       ext3  1011M  483M  477M  51%  /opt
/dev/hda9       ext3   2.0G   607M   1.3G  32%  /var
/dev/hda1       ext2    51M    17M    31M  36%  /boot
/dev/hda6       swap   516M   12M   504M   2%  <not mounted>
(свободное место для будущего использования: 2 ГБ)
```

Раздел `/usr`, как видим, почти заполнен (использовано 83%), но когда все программы установлены, `/usr` растет не слишком быстро. Хотя отведение нескольких гигабайт дискового пространства для `/var` может показаться расточительством, помните, что этот раздел по умолчанию используется Portage для компиляции пакетов. Если вы захотите удержать `/var` в рамках более разумного размера, например, 1ГБ, вам потребуется изменить переменную **PORTAGE_TMPDIR** в `/etc/make.conf`, чтобы она указывала на раздел, где достаточно свободного места для компиляции чрезвычайно больших пакетов, таких как OpenOffice.

4.с. Использование fdisk для создания разделов

В следующих разделах описывается, как разбить диск в соответствии с примерной схемой, описанной ранее, а именно:

Раздел	Описание
<code>/dev/hda1</code>	загрузочный раздел
<code>/dev/hda2</code>	раздел подкачки
<code>/dev/hda3</code>	корневой раздел

Эту схему вы можете изменять по своему усмотрению.

Просмотр текущей схемы разбиения диска

fdisk — это популярная и очень мощная утилита для создания разделов на ваших дисках. Запустите `fdisk`, указав свой диск в качестве параметра (в примере мы используем `/dev/hda`):

Листинг 2: Запуск `fdisk`

```
# fdisk /dev/hda
```

После запуска **fdisk** выдаст такое приглашение:

Листинг 3: Приглашение `fdisk`
Command (m for help):

Нажмите **p**, чтобы вывести текущую схему разбиения диска:

Листинг 4: Пример схемы разделов диска

Command (m for help): p

Disk /dev/hda: 240 heads, 63 sectors, 2184 cylinders
Units = cylinders of 15120 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	14	105808+	83	Linux
/dev/hda2		15	49	264600	82	Linux swap
/dev/hda3		50	70	158760	83	Linux
/dev/hda4		71	2184	15981840	5	Extended
/dev/hda5		71	209	1050808+	83	Linux
/dev/hda6		210	348	1050808+	83	Linux
/dev/hda7		349	626	2101648+	83	Linux
/dev/hda8		627	904	2101648+	83	Linux
/dev/hda9		905	2184	9676768+	83	Linux

Command (m for help):

В данном случае на диске есть семь разделов Linux (для которых в графе «System» указано «Linux») и один раздел подкачки (в списке показан как «Linux swap»).

Удаление всех разделов

Сначала удалим с диска все существующие разделы. Для удаления раздела вводите **d**. Например для удаления существующего /dev/hda1:

Листинг 5: Удаление раздела

Command (m for help): d

Partition number (1-4): 1

Удаление раздела будет запланировано. Он больше не будет отображаться при вводе **p**, но фактически не будет удалаться до тех пор, пока вы не сохраните внесенные изменения. Если вы ошиблись и хотите прервать разбиение без сохранения изменений, немедленно введите **q** и нажмите ENTER; тогда ваш раздел сохранится в неизменном виде.

Теперь, если вы действительно собираетесь удалить в своей системе все разделы, повторяйте ввод **p** для показа оставшихся разделов, затем **d** и номера удаляемого раздела до тех пор, пока разделы не кончатся. В итоге вы получите пустую таблицу разделов:

Листинг 6: Пустая таблица разделов

Disk /dev/hda: 30.0 GB, 30005821440 bytes
240 heads, 63 sectors/track, 3876 cylinders
Units = cylinders of 15120 * 512 = 7741440 bytes

Device	Boot	Start	End	Blocks	Id	System
Command (m for help):						

Теперь, когда мы очистили таблицу разделов, хранящуюся в оперативной памяти, настало время создавать разделы. Мы воспользуемся примерной схемой разбиения, описанной выше. Естественно, не следуйте этим инструкциям буквально, если только вам не нужна таблица разделов, идентичная нашей!

Создание загрузочного раздела

Сначала создадим маленький загрузочный раздел. Введите **n** для создания нового раздела, затем **p**, чтобы выбрать первичный раздел, и **1** для указания первого первичного раздела. На вопрос о первом цилиндре нажмите ввод. На вопрос о последнем цилиндре введите **+32M**, чтобы создать раздел размером **32МБ**:

Листинг 7: Создание загрузочного раздела

Command (m for help): n

Command action

e extended

p primary partition (1-4)

p

Partition number (1-4): 1

First cylinder (1-3876, default 1): (нажмите ввод)

Using default value 1

Last cylinder or +size or +sizeM or +sizeK (1-3876, default 3876): +32M

Теперь, введя **p**, вы должны увидеть следующий текст:

Листинг 8: Созданный загрузочный раздел

Command (m for help): p

Disk /dev/hda: 30.0 GB, 30005821440 bytes

240 heads, 63 sectors/track, 3876 cylinders

Units = cylinders of 15120 * 512 = 7741440 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	14	105808+	83	Linux

Нам необходимо сделать этот раздел загружаемым. Введите **a** для включения признака загрузки с раздела, затем нажмите **1**. Если снова ввести **p**, можно увидеть, что в столбце «boot» появился знак *.

Создание раздела подкачки

Теперь создадим раздел подкачки. Для этого введите **n** (создание нового раздела), затем **p** для выбора первичного раздела. Потом нажмите **2**, чтобы создать второй первичный раздел, в нашем случае **/dev/hda2**. На вопрос о первом цилиндре просто нажмите ввод. На вопрос о последнем — ответьте **+512M**, чтобы

создать раздел размером 512МБ. Сделав это, введите **t** для указания типа раздела, **2**, для выбора только что созданного раздела, и **82**, чтобы установить тип раздела «Linux Swap». При нажатии **p** по завершении этих действий, таблица разделов должна выглядеть примерно так:

Листинг 9: Список разделов после создания раздела подкачки

Command (m for help): p

```
Disk /dev/hda: 30.0 GB, 30005821440 bytes
240 heads, 63 sectors/track, 3876 cylinders
Units = cylinders of 15120 * 512 = 7741440 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	14	105808+	83	Linux
/dev/hda2		15	81	506520	82	Linux swap

Создание корневого раздела

Теперь создадим корневой раздел. Для этого введите **n** (создание нового раздела), затем **p** (первичный раздел). После этого нажмите **3** для создания третьего первичного раздела, в нашем случае /dev/hda3. На вопрос о первом цилиндре нажмите ввод. На вопрос о последнем — также нажмите ввод, чтобы раздел занял все оставшееся свободное место на диске. По завершении этих шагов, при вводе **p** должна выводиться подобная таблица разделов:

Листинг 10: Список разделов после создания корневого раздела

Command (m for help): p

```
Disk /dev/hda: 30.0 GB, 30005821440 bytes
240 heads, 63 sectors/track, 3876 cylinders
Units = cylinders of 15120 * 512 = 7741440 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1	*	1	14	105808+	83	Linux
/dev/hda2		15	81	506520	82	Linux swap
/dev/hda3		82	3876	28690200	83	Linux

Сохранение схемы разбиения

Для сохранения схемы разбиения и выхода из **fdisk**, введите **w**.

Листинг 11: Сохранение и выход из fdisk

Command (m for help): w

Теперь, создав все разделы, перейдем к созданию файловых систем.

4.d. Создание файловых систем

Введение

Разделы созданы, настало время разместить на них файловые системы.

Если вам безразлично, какую файловую систему использовать, и вы вполне довольны той, что мы предлагаем в книге по умолчанию, переходите к размещению файловой системы в разделе. Если нет, читайте дальше, чтобы разузнать о существующих файловых системах...

Файловые системы (сравнение !)

В ядре Linux поддерживаются различные файловые системы. Мы расскажем о самых распространенных из них: **ext2**, **ext3**, **ReiserFS**, **XFS** и **JFS**.

- **ext2** — испытанная файловая система Linux, в которой отсутствуют средства журналирования метаданных. Поэтому ее рядовая проверка при запуске может быть очень долгой. Сегодня существует довольно много журналируемых файловых систем нового поколения, целостность которых проверяется очень быстро, и поэтому обычно есть смысл использовать их. Журналируемые файловые системы позволяют избежать долгих задержек при запуске, когда состояние файловой системы неустойчиво.
- **ext3** — это журналируемая версия ext2, где для ускорения восстановления появилось журналирование метаданных, а также другие режимы, наподобие полного или упорядоченного журналирования данных. ext3 — очень хорошая и надежная файловая система. В ней есть дополнительная возможность индексации через хэшированные сбалансированные деревья (b-tree), что в большинстве случаев обеспечивает высокую скорость. Такую индексацию можно включить, добавив к команде `mke2fs` параметр `-O dir_index`. Короче говоря, ext3 — превосходная файловая система.
- **ReiserFS** — файловая система, построенная на B*-деревьях. У нее очень хорошая скорость, и она намного (часто в 10-15 раз) быстрее ext2 и ext3 при работе с маленькими файлами (менее 4КБ). ReiserFS также великолепно масштабируется; в ней есть средства журналирования. На дату выхода ядра 2.4.18, ReiserFS признана стабильной и пригодной как для систем общего назначения, так и для крайних случаев типа создания больших томов, использования множества маленьких файлов, гигантских файлов или же каталогов с десятками тысяч файлов.
- **XFS** — это файловая система с журналированием метаданных и дополнительными средствами, оптимизированными для работы с максимальным объемом данных. Ее применение рекомендуется только в Linux-системах с высококлассными дисками SCSI или дисками с оптическим подключением (fibre-channel), при наличии источников бесперебойного питания. Из-за того, что XFS выполняет очень агрессивное промежуточное кэширование в оперативной памяти, при внезапном отключении питания очень вероятна потеря небрежно спроектированными программами (не принимающими адекватные меры предосторожности при записи файлов на диск, и таких довольно много) изрядного количества данных.
- **JFS** — высокопроизводительная журналируемая файловая система

производства IBM. Она лишь недавно стала пригодной для широкого применения, и пока слишком мало данных, чтобы сказать что-то хорошее или плохое о ее общей стабильности.

Размещение файловой системы в разделе

Для создания в разделе или томе файловой системы каждого типа существуют специальные средства:

Файловая система	Команда создания
ext2	mke2fs
ext3	mke2fs -j
reiserfs	mkreiserfs
xfs	mkfs.xfs
jfs	mkfs.jfs

Например, чтобы у загрузочного раздела (/dev/hda1 в наших примерах) была файловая система ext2, а у корневого раздела (/dev/hda3 в наших примерах) — ext3, требуется выполнить:

Листинг 12: Создание файловых систем разделов

```
# mke2fs /dev/hda1
# mke2fs -j /dev/hda3
```

Теперь самостоятельно создайте файловые системы на своих только что созданных разделах (логических томах).

Подключение раздела подкачки

mkswap — команда, используемая для инициализации разделов подкачки:

Листинг 13: Создание идентификатора раздела подкачки

```
# mkswap /dev/hda2
```

Для подключения раздела подкачки воспользуйтесь **swapon**:

Листинг 14: Подключение раздела подкачки

```
# swapon /dev/hda2
```

Теперь командами, приведенными выше, создайте и подключите раздел подкачки в своей системе.

4.e. Монтирование

Теперь, когда разделы созданы, а файловые системы размещены, настало время смонтировать (подключить к системе) эти разделы. Используйте команду **mount**. Не забудьте создать для каждого раздела соответствующие каталоги монтирования. Например, смонтируем корневой и загрузочный разделы:

Листинг 15: Монтирование разделов

```
# mount /dev/hda3 /mnt/gentoo
# mkdir /mnt/gentoo/boot
# mount /dev/hda1 /mnt/gentoo/boot
```

Примечание: Если вы хотите разместить каталог `/tmp` в отдельном разделе, не забудьте изменить права доступа к этому каталогу после монтирования: **`chmod 1777 /mnt/gentoo/tmp`**. Это также относится к `/var/tmp`.

Еще нам потребуется смонтировать файловую систему **proc** (виртуальный интерфейс ядра) в каталог **`/proc`**. Но сначала надо поместить в разделы нужные файлы.

Переходите к установке установочных файлов Gentoo.

5. Установка установочных файлов Gentoo

5.a. Установка архива стадии

Установка текущей даты и времени

Перед тем, как продолжать, убедитесь в правильности системной даты и времени, и при необходимости обновите их. Если часы установлены неверно, это в будущем может привести к странным результатам!

Для проверки времени, введите команду **`date`**:

Листинг 1: Проверка даты и времени

```
# date
Fri Mar 29 16:21:18 UTC 2007
```

Если часы установлены неправильно, измените время командой **`date ММДДчммГГГГГ`** (формат: Месяц, День, часы, минуты и Год). На этом этапе следует указывать время UTC. Позднее вы сможете установить свой часовой пояс. Например, для установки часов на 29 марта, 16:21, 2005 год, выполните:

Листинг 2: Установка даты и времени

```
# date 032916212007
```

Выбор источника

Следующий необходимый шаг — установка архива **stage3** на вашу систему. Нужный архив можно скачать из интернета, или, если вы загрузились с универсального установочного диска Gentoo, скопировать с самого диска. Если у

у вас есть универсальный диск, а на нем — нужная стадия, то загрузка из интернета — лишняя трата трафика и времени, т.к. файлы стадии окажутся совершенно одинаковы. В большинстве случаев, определиться с выбором архива стадии вам поможет команда **uname -m**.

по умолчанию: использование файла стадии из интернета

альтернатива: использование файла стадии с установочного диска

5.b. По умолчанию: использование файла стадии из интернета

Загрузка архива стадии

Перейдите в точку подключения файловой системы Gentoo (обычно — /mnt/gentoo):

Листинг 3: Переход к точке подключения Gentoo

```
# cd /mnt/gentoo
```

Для загрузки файла стадии у вас уже есть различные инструменты, зависящие от типа диска, с которого производится установка. Если имеется **links**, можете сразу открыть список зеркал Gentoo и выбрать ближайшее.

Когда **links** нет, в вашем распоряжении должен оказаться **lynx**. Если требуется использовать прокси-сервер, экспортируйте переменные **http_proxy** и **ftp_proxy**:

Листинг 4: Настройка информации о прокси-сервере для **lynx**

```
# export http_proxy="http://proxy.server.com:port"
# export ftp_proxy="http://proxy.server.com:port"
```

Предположим, у вас под рукой есть *links*.

Перейдите в каталог **releases/**, в нем откройте каталог, соответствующий вашей архитектуре (например, **x86/**), затем каталог с версией Gentoo (**2006.1/**), и, наконец, каталог **stages/**. Здесь находятся все доступные архивы стадий для вашей архитектуры (они могут располагаться в подкаталогах с названиями разновидностей архитектуры). Выберите один из них и нажмите **D** для загрузки. По окончании — нажмите **Q**, чтобы выйти из браузера.

Листинг 5: Открытие списка зеркал в **links**

```
# links http://www.gentoo.org/main/ru/mirrors.xml
```

(если в **links** нужна поддержка прокси-сервера:)

```
# links -http-proxy proxy.server.com:8080 http://www.gentoo.org/main/ru/mirrors.xml
```

Удостоверьтесь, что вы скачали **stage3**-архив — установка с использованием файлов **stage1** и **stage2** более не поддерживается.

Если вы хотите проверить целостность загруженного архива стадии, с помощью **md5sum** сравните результат с контрольной суммой MD5, взятой с зеркала. Например, для проверки целостности архива стадии для x86:

Листинг 6: Пример проверки целостности архива стадии

```
# md5sum -c stage3-x86-2007.0.tar.bz2.DIGESTS
stage3-x86-2007.0.tar.bz2: OK
```

Распаковка архива стадии

Распакуйте загруженный архив стадии в своей системе. Мы используем **tar**, т.к. это простейший способ:

Листинг 7: Распаковка архива стадии

```
# tar xvjpf stage3-*.tar.bz2
```

Убедитесь, что вы используете точно такие же параметры командной строки (**xvjpf**). Значения параметров:

- **x** — *извлечение*;
- **v** — *подробные сообщения, чтобы видеть, что происходит во время распаковки (необязательный параметр)*;
- **j** — *декомпрессия bzip2*;
- **p** — *сохранение прав доступа*;
- **f** — *указывает на то, что мы распаковываем файл, а не то, что подается на стандартный ввод*.

Примечание: На установочных компакт-дисках и загрузочных образах для некоторых архитектур (например, MIPS) **tar** встроен в оболочку BusyBox, которая на данный момент не поддерживает параметр **v**. Тогда вместо приведенных указывайте параметры **xjpf**.

После развертывания установки стадии переходите к установке дерева портежей.

5.с. Альтернатива: использование файла стадии с установочного диска

Извлечение архива стадии

Важно: Если вы — в x86 и используете «живой диск» с установщиком, на нем нет стадий. Вам понадобится последовать указаниям по использованию файла стадии из интернета.

Архивы стадий находятся на компакт-диске в каталоге `/mnt/cdrom/stages`. Для получения списка доступных стадий, воспользуйтесь **ls**:

Листинг 8: Список доступных архивов стадий

```
# ls /mnt/cdrom/stages
```

Если система сообщает об ошибке, возможно, сначала нужно смонтировать CD-ROM:

Листинг 9: Монтирование CD-ROM

```
# ls /mnt/cdrom/stages
ls: /mnt/cdrom/stages: No such file or directory
# mount /dev/cdroms/cdrom0 /mnt/cdrom
# ls /mnt/cdrom/stages
```

Перейдите в точку подключения Gentoo (обычно — /mnt/gentoo):

Листинг 10: Смена каталога на /mnt/gentoo

```
# cd /mnt/gentoo
```

Теперь утилитой tar распакуем выбранный архив стадии. Убедитесь, что вы используете точно такие же параметры (xvjpf)! Параметр v необязателен и может не работать в некоторых версиях tar. В следующем примере мы распакуем архив стадии stage3-<архитектура>-2006.1.tar.bz2. В качестве имени архива обязательно указывайте название файла нужной вам стадии.

Листинг 11: Распаковка архива стадии

```
# tar xvjpf /mnt/cdrom/stages/stage3-<архитектура>-2007.0.tar.bz2
```

После установки стадии переходите к установке дерева портежей.

5.d. Установка дерева портежей

Распаковка снимка портежей

Теперь вам нужно установить снимок дерева портежей — набор файлов, содержащих сведения для Portage о программном обеспечении, доступном для установки, имеющихся профилях и т.д.

Загрузка и установка снимка дерева портежей

Перейдите к точке подключения вашей системы (обычно — /mnt/gentoo):

Листинг 12: Переход к точке подключения

```
# cd /mnt/gentoo
```

Запустите **links** (или **lynx**) и откройте список зеркал Gentoo. Выберите ближайшее зеркало и перейдите в каталог snapshots/. Оттуда загрузите новейший снимок дерева портежей (portage-latest.tar.bz2), выбрав его из списка и нажав **D**.

Листинг 13: Просмотр списка зеркал Gentoo

```
# links http://www.gentoo.org/main/ru/mirrors.xml
```

Закройте браузер нажатием **Q**. Теперь у вас в `/mnt/gentoo` появился снимок дерева портежей.

Если нужно убедиться в целостности загруженного снимка, воспользуйтесь `md5sum`, и сравните результат с контрольной суммой MD5, имеющейся на зеркале.

Листинг 14: Checking integrity of a Portage snapshot

```
# md5sum -c portage-latest.tar.bz2.md5sum
portage-latest.tar.bz2: OK
```

Теперь нужно распаковать снимок. Убедитесь в том, что вы используете в точности такие же параметры; последний параметр — заглавная **C**, а не строчная **c**.

Листинг 15: Извлечение снимка дерева портежей

```
# tar xvjf /mnt/gentoo/portage-latest.tar.bz2 -C /mnt/gentoo/usr
```

5.e. Настройка параметров компиляции

Введение

Для оптимизации Gentoo вы можете создать ряд переменных, которые повлияют на поведение Portage. Все их можно создавать как переменные среды (с помощью `export`), но тогда они будут лишь временными. Для хранения ваших настроек предназначен конфигурационный файл Portage, `/etc/make.conf`. Именно его мы сейчас будем редактировать.

Примечание: Список и описание всех допустимых переменных находятся в файле `/mnt/gentoo/etc/make.conf.example`. Для успешной установки Gentoo, достаточно установить значения только тех переменных, которые описаны ниже.

Запустите ваш любимый редактор (в этом руководстве используется **nano**) для изменения параметров оптимизации, которые обсуждаются далее.

Листинг 16: Открытие `/etc/make.conf`

```
# nano -w /mnt/gentoo/etc/make.conf
```

Как вы могли заметить, у файла **make.conf.example** обычная структура: строки комментария начинаются со знака «**#**», а в остальных строках выражениями вида **ПЕРЕМЕННАЯ**="значение" определяются переменные. У файла `make.conf` такой же формат. Некоторые из переменных мы обсудим ниже.

CHOST

В переменной CHOST определяется, для какой архитектуры скомпилирована система. В ней уже должно быть установлено правильное значение. Не изменяйте эту переменную, т.к. это может повредить вашу систему. Если переменная CHOST, по-вашему, выглядит неправильно, возможно, вы используете не тот архив stage3.

CFLAGS и CXXFLAGS

Переменные CFLAGS и CXXFLAGS определяют параметры оптимизации компилятора gcc для языков C и C++, соответственно. При том, что общие значения следует устанавливать здесь, максимальная производительность достигается в том случае, когда для каждой компилируемой программы устанавливаются свои собственные оптимальные параметры. Причина в том, что все программы различны.

В файле **make.conf** следует указывать параметры оптимизации, которые, на ваш взгляд, повысят скорость системы в целом. Не помещайте сюда экспериментальные значения; излишняя оптимизация может привести к плохому поведению программ (аварийным завершениям, или, что хуже, неправильной работе).

Мы не будем описывать все возможные параметры оптимизации. Если вы хотите узнать о них всё, почитайте онлайн-руководства GNU (англ.) или страницу описания **gcc** (info gcc — доступна только на работающей системе Linux). Не забудьте прочитать сам файл **make.conf.example**: в нем также есть немало примеров и полезных сведений.

Первый параметр — флаг **-march=**, где указывается название целевой архитектуры. Возможные варианты описаны в комментариях в файле **make.conf.example**. Например, для архитектуры x86 Athlon XP:

Листинг 17: Значение **-march** для GCC

```
# пользователям AMD64, желающим получить действительно 64-битную
# систему, следует использовать -march=k8
# пользователи EM64T должны указать -march=nocona
-march=athlon-xp
```

Примечание(JohnBat26) (обновлено в 1.5): Если Вы используете компилятор версии 4.2.0 и выше, то вместо указания специфичного типа процессора, можно указывать одно из двух (в параметрах: **-march** и **-mtune**):

1. **generic**: если Вы хотите, чтобы Ваш скомпилированный код запускался на всех процессорах, архитектуры x86;
2. **native**: если Вы хотите оптимизировать код только для Вашего процессора. В этом случае компилятор будет брать сведения о процессоре путем вызова **cruid** !

Следующий — флаг **-O** (заглавная латинская «O», а не ноль), определяющий

класс оптимизации в gcc. Допустимые значения:

- **s** — оптимизация по размеру;
- **0** — ноль, без оптимизации;
- **1, 2** или **3** — все большая оптимизация по скорости (в каждый класс входят все флаги предыдущего, и некоторые дополнительные).

Например, для оптимизации класса 2:

Листинг 18: Значение -O для GCC
-O2

Другой популярный флаг оптимизации — **-pipe** (использование для связи между различными проходами компилятора каналов вместо временных файлов).

Заметьте, что использование **-fomit-frame-pointer** (не хранить в регистре указатель стекового кадра для функций, которым он не нужен) может всерьез помешать отладке программ!

При определении переменных **CFLAGS** и **CXXFLAGS**, нужно объединять несколько флагов оптимизации, как в следующем примере:

Листинг 19: Определение переменных CFLAGS и CXXFLAGS (gcc < 4.2.0)
 CFLAGS="-march=athlon-xp -pipe -O2" # для пользователей AMD64: -march=k8
 # для пользователей EM64T: -march=nocona
 CXXFLAGS="\${CFLAGS}" # указывайте одинаковые значения обеих переменных

MAKEOPTS

С помощью **MAKEOPTS** определяется, сколько параллельных процессов компиляции можно запускать при установке пакета. Хороший (но не всегда идеальный) вариант — значение, равное количеству процессоров в системе плюс один.

Листинг 20: MAKEOPTS для обычной однопроцессорной системы
MAKEOPTS="-j2"

На старт, внимание, марш!

Измените **/mnt/gentoo/etc/make.conf** в соответствии со своими пожеланиями, и сохраните изменения (пользователям nano нужно нажать CTRL+X). Теперь вы готовы к установке базовой системы Gentoo.

6. Установка базовой системы Gentoo

6.a. Изменение корневого каталога

Дополнительно: выбор зеркала

Для ускорения загрузки исходного кода рекомендуется выбрать быстрое зеркало. Portage ищет переменную **GENTOO_MIRRORS** в файле **make.conf** и использует зеркала, перечисленные в ней. Конечно, можно просмотреть наш список зеркал и выбрать одно или несколько, географически ближайших к вам (чаще всего они и будут самыми быстрыми), но мы предлагаем вам удобную утилиту **mirrorselect**, которая позволяет выбрать желаемые зеркала более удобным способом.

Листинг 1: Запуск **mirrorselect** для установки переменной **GENTOO_MIRRORS**
mirrorselect -i -o >>/mnt/gentoo/etc/make.conf

Предупреждение: Не выбирайте зеркала IPv6. Файлы стадий пока не поддерживают протокол IPv6.

Вторая важная настройка — установка значения переменной **SYNC** в файле **make.conf**. Эта переменная указывает на сервер **rsync** (сервер удаленной синхронизации), выбранный вами для обновления дерева Portage (коллекции файлов ebuild — сборочных сценариев, содержащих все данные, нужные Portage для скачивания и установки программ). Хотя вписать адрес сервера в **SYNC** можно и вручную, **mirrorselect** позволяет упростить это действие:

Листинг 2: Выбор зеркала **rsync** с помощью **mirrorselect**
mirrorselect -i -r -o >> /mnt/gentoo/etc/make.conf

После выполнения **mirrorselect**, мы рекомендуем проверить все значения в файле **/mnt/gentoo/etc/make.conf**!

Копирование сведений о DNS

Перед тем, как перейти в новую среду, осталось сделать одно дело: скопировать информацию о DNS (системе доменных имен) в файл **/etc/resolv.conf**. Это нужно, чтобы при переходе сохранить работоспособность сети. В файле **/etc/resolv.conf** содержатся адреса серверов имен, используемых в вашей сети.

Листинг 3: Копирование информации о DNS
(параметр «-L» нужен, чтобы случайно не скопировать
символьную ссылку)
cp -L /etc/resolv.conf /mnt/gentoo/etc/resolv.conf

Монтирование файловых систем **/proc** и **/dev**

Смонтируйте файловую систему **/proc** в **/mnt/gentoo/proc**, чтобы после изменения корневого каталога устанавливаемая система смогла обращаться к информации, предоставляемой ядром, а затем создайте привязку монтирования файловой системы **/dev**.

Листинг 4: Монтирование **/proc** и **/dev**
mount -t proc none /mnt/gentoo/proc

```
# mount -o bind /dev /mnt/gentoo/dev
```

Переход в новую среду

Итак, все разделы подготовлены, а базовая операционная среда установлена. Теперь пора войти в нее, изменив корневой каталог. Таким образом, мы перейдем из текущей установочной среды (с компакт-диска или другого установочного носителя) в свою устанавливаемую систему (находящуюся в недавно размеченных разделах).

Изменение выполняется в три этапа:

1. Сначала мы с помощью **chroot** изменим корневой каталог с **/** (находящийся на установочном носителе) на **/mnt/gentoo** (находящийся на ваших дисковых разделах).
2. Затем мы создадим новую среду, пользуясь утилитой **env-update**, которая, собственно, создает переменные среды.
3. Наконец, мы загрузим эти переменные в память при помощи **source**.

Листинг 5: Изменение корневого каталога для перехода в новую среду

```
# chroot /mnt/gentoo /bin/bash
# env-update
>> Regenerating /etc/ld.so.cache...
# source /etc/profile
# export PS1="(chroot) $PS1"
```

Поздравляем! Теперь вы в своей собственной среде Gentoo Linux. Конечно, на этом она далеко еще не готова, поэтому в руководстве еще осталось несколько разделов :-)

6.b. Настройка Portage

Обновление дерева Portage

Теперь надо обновить дерево Portage до самой последней версии с помощью команды **emerge --sync**:

Листинг 6: Обновление дерева портежей

```
# emerge --sync
(если вы пользуетесь медленным терминалом, например, последовательным
терминалом или кадровым буфером, для ускорения процесса можно добавить
параметр --quiet :)
# emerge --sync --quiet
```

Если компьютер подключен к интернету через межсетевой экран, блокирующий прохождение rsync-пакетов, вы можете воспользоваться командой **emerge-webrsync**, которая скачивает и устанавливает снимок дерева.

Если выдано предупреждение, что имеется новая версия Portage и ее нужно

обновить, выполните обновление командой **emerge portage**.

Выбор нужного профиля

Сначала дадим небольшое определение.

Профиль — конструктивный элемент любой системы Gentoo. В нем указываются не только значения по умолчанию для **CHOST**, **CFLAGS** и других важных переменных, он также привязывает систему к определенному диапазону допустимых версий пакетов. Диапазоны поддерживаются разработчиками Gentoo.

Раньше пользователь редко касался профиля. Однако, пользователи **x86**, **hppa** и **alpha** могут выбирать из двух вариантов профиля: одного для ядра 2.4, другого для ядра 2.6. Это вызвано необходимостью улучшения интеграции ядер 2.6. Для архитектур ppc и ppc64 также существует несколько профилей. Мы поговорим о них позже.

Узнать, какой профиль используется в системе в данный момент, вы можете командой:

Листинг 7: Выяснение используемого профиля

```
# ls -FGg /etc/make.profile
```

```
lrwxrwxrwx 1 48 Apr 8 18:51 /etc/make.profile -> ../usr/portage/profiles/default-linux/x86/2007.0/
```

Если вы используете одну из трех упомянутых архитектур, профиль по умолчанию даст вам систему с ядром 2.6. Это рекомендуется по умолчанию, но за вами сохраняется и право выбора другого профиля.

Для некоторых архитектур также предусмотрены подпрофили **desktop** и **server**. Загляните в профиль 2007.0/, чтобы узнать, есть ли подпрофили для вашей архитектуры. Возможно, вы захотите заглянуть в **make.defaults** профиля **desktop**, чтобы определить, подходит ли он вам.

Некоторым пользователям, возможно, захочется установить систему, основанную на старом профиле, с ядром 2.4. Если для этого есть веские основания, сначала надо проверить, существует ли такой дополнительный профиль. На системах x86 это можно сделать следующей командой:

Листинг 8: Выяснение наличия дополнительного профиля

```
# ls -d /usr/portage/profiles/default-linux/x86/no-nptl/2.4
```

```
/usr/portage/profiles/default-linux/x86/no-nptl/2.4
```

В приведенном примере дополнительный профиль 2.4 существует (т.е. нет сообщений об отсутствующем файле или каталоге). Рекомендуется использовать профиль по умолчанию, но если вы все же хотите сменить его, это можно сделать так:

Листинг 9: Переключение профиля на 2.4

(убедитесь, что используете нужную архитектуру; этот пример - для x86)

```
# ln -snf /usr/portage/profiles/default-linux/x86/no-nptl/2.4 /etc/make.profile
```


(вывод списка файлов в профиле 2.4)

```
# ls -FGg /etc/make.profile/
total 12
-rw-r--r-- 1 939 Dec 10 14:06 packages
-rw-r--r-- 1 347 Dec 3 2004 parent
-rw-r--r-- 1 573 Dec 3 2004 virtuals
```

Для архитектуры **ppc** в выпуске 2006.1 появилось несколько новых профилей:

Листинг 10: Профили для PPC

(базовый профиль PPC, подходит для всех PPC-машин, минимальный)

```
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc32/2006.1 /etc/make.profile
(профиль для G3)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc32/2006.1/G3 /etc/make.profile
(профиль для G3 Pegasos)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc32/2006.1/G3/Pegasos/ /etc/make.profile
(профиль для G4 (Altivec))
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc32/2006.1/G4 /etc/make.profile
(профиль для G4 (Altivec) Pegasos)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc32/2006.1/G4/Pegasos/ /etc/make.profile
```

Для архитектуры **ppc64** в выпуске 2006.1 также появилось несколько новых профилей:

Листинг 11: Профили для PPC64

(базовый профиль PPC64 с 64-битным режимом пользователя, для всех PPC64-машин)

```
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/64bit-userland /etc/make.profile
(базовый профиль PPC64 с 32-битным режимом пользователя, для всех PPC64-машин)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/32bit-userland /etc/make.profile
(для каждого режима пользователя существуют подпрофили, как показано ниже.)
(«userland» необходимо заменять на нужный вариант режима из примеров выше)
(профиль 970 для JS20)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/(userland)/970 /etc/make.profile
(профиль для G5)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/(userland)/970/pmac /etc/make.profile
(профиль для POWER3)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/(userland)/power3 /etc/make.profile
(профиль для POWER4)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/(userland)/power4 /etc/make.profile
(профиль для POWER5)
# ln -snf /usr/portage/profiles/default-linux/ppc/ppc64/2006.1/(userland)/power5 /etc/make.profile
(многорежимные профили на дату выхода этого выпуска нестабильны)
```

Настройка переменной USE

USE («использовать») — одна из самых мощных переменных, имеющих в распоряжении пользователей Gentoo. Она позволяет при компиляции программ включать или отключать поддержку определенных необязательных функций. Например, некоторые программы можно компилировать с поддержкой **gtk** или **qt** на выбор. Другие можно собирать, включая или отключая поддержку **SSL**. Некоторые программы можно даже компилировать с поддержкой кадрового

буфера (**svgalib**) вместо поддержки X11 (X-сервера).

В большинстве дистрибутивов пакеты собраны с поддержкой практически всех мыслимых вариантов. Это увеличивает размер программ и время запуска, не говоря уже о чрезмерных зависимостях. В Gentoo вы сами можете определять, с какими возможностями следует компилировать пакет. Здесь играет роль переменная **USE**.

В переменной **USE** указываются ключевые слова, которые отражаются на параметрах компиляции. Например, параметр `ssl` включает компиляцию с поддержкой `ssl` всех программ, которые способны его поддерживать. `-X` отключает поддержку X-сервера (обратите внимание на предшествующий знак «минус»). Параметры `gnome` `gtk` `-kde` `-qt` обеспечивают компиляцию ваших программ с поддержкой Gnome и `gtk`, но без поддержки KDE и `qt`, делая систему оптимальной для GNOME.

Настройки **USE** по умолчанию хранятся в файлах **make.defaults** вашего профиля. Файлы `make.defaults` находятся в каталоге, на который указывает `/etc/make.profile`, а также во всех родительских каталогах. Значение **USE** по умолчанию — это сумма всех значений **USE** во всех файлах `make.defaults`. Все, что вы вносите в `/etc/make.conf`, рассчитывается относительно этих значений. Когда вы добавляете что-либо к значению **USE**, оно добавляется в список по умолчанию. Когда удаляете что-либо (указывая ключевое слово с предшествующим знаком минус), оно удаляется из списка по умолчанию (если оно там вообще было). Никогда ничего не меняйте в `/etc/make.profile`. Все, что там находится, перезаписывается при обновлении Portage!

Полное описание переменной **USE** находится во второй части настольной книги Gentoo в главе USE-флаги. Полное описание возможных значений признаков использования находится в вашей системе в файле `/usr/portage/profiles/use.desc`.

Листинг 12: Просмотр допустимых USE-флагов

```
# less /usr/portage/profiles/use.desc
```

(для прокрутки пользуйтесь клавишами стрелок, для выхода нажимайте "q")

Например, приведем значение **USE** для системы, базирующейся на KDE, с включением поддержки DVD, ALSA и записи CD:

Листинг 13: Открытие файла `/etc/make.conf`

```
# nano -w /etc/make.conf
```

Листинг 14: Значение **USE**

```
USE="-gtk -gnome qt kde dvd alsa cdr"
```

Дополнительно: региональные параметры GLIBC

Скорее всего, вы будете использовать в системе один-два набора региональных параметров. Нужные регионы можно указать в `/etc/locale.gen`.

Листинг 15: Открытие `/etc/locale.gen`
nano -w /etc/locale.gen

Вот пример одновременного подключения как английского (США), так и немецкого (Германия) с соответствующими кодировками (например, UTF-8).

Листинг 16: Укажите свои региональные настройки
 en_US/ISO-8859-1
 en_US.UTF-8/UTF-8
 de_DE/ISO-8859-1
 de_DE@euro/ISO-8859-15

Следующий шаг — запустить **locale-gen**. Это сгенерирует настройки для всех регионов, указанных вами в файле `/etc/locale.gen`.

Примечание: `locale-gen` имеется в `glibc-2.3.6-r4` и более новой. Если у вас старшая версия `glibc`, сейчас ее следует обновить.

Примечание: Настройки, подобной приведенной, для поддержки русского языка недостаточно. За дополнительными сведениями обращайтесь к отдельным руководствам по русификации Gentoo. — прим. пер.

Теперь приступим к настройке ядра.

7. Настройка ядра

7.а. Часовой пояс

Сначала необходимо выбрать часовой пояс (time zone), чтобы система знала, где вы находитесь. Найдите подходящий пояс в `/usr/share/zoneinfo`, затем скопируйте его в `/etc/localtime`. Постарайтесь, пожалуйста, избегать использования часовых поясов `/usr/share/zoneinfo/Etc/GMT*`, т. к. их названия не соответствуют ожидаемым поясам. Например, GMT-8 фактически является поясом GMT+8.

Листинг 1: Установка часового пояса
ls /usr/share/zoneinfo
 (допустим, вы хотите использовать GMT)
cp /usr/share/zoneinfo/GMT /etc/localtime

Примечание (JohnBat26): Если возникают проблемы со временем (особенно с часовыми поясами), то выполните следующее (получено с форума):

1. Выставьте правильное время.
2. Удалите файл `/etc/adjtime`. Все, часы идут при перезагрузке нормально.
3. `nano -w /etc/conf.d/clock`. Ставьте там вместо UTC Local, и все!!!

7.b. Установка исходных кодов ядра

Выбор ядра

Ядро Linux — это то, вокруг чего формируются все дистрибутивы. Это слой между пользовательскими программами и вашим оборудованием. Пользователи Gentoo могут выбирать из нескольких типов ядра. Их полный список и описание находится в руководстве по выбору ядра Gentoo (англ.).

Для x86 систем, помимо прочего, есть такие ядра, как

- **vanilla-sources** (ядро по умолчанию от разработчиков linux ядра);
- **gentoo-sources** (ядро с заплатками, улучшающими производительность).

С помощью команды **emerge** выберите и установите ядро. USE="-doc" требуется, чтобы на этом этапе избежать установки хорг-х11 или других связанных пакетов. USE="symlink" при новой установке не требуется, но обеспечивает правильное создание символической ссылки /usr/src/linux:

Листинг 2: Установка исходных кодов ядра
USE="-doc symlink" emerge gentoo-sources

Просмотрев содержимое каталога /usr/src, вы увидите символическую ссылку linux, которая указывает на каталог с исходными кодами ядра. В данном случае, установленный исходный код ядра указывает на gentoo-sources-2.6.12-r10. Имейте в виду, что ваша версия может отличаться.

Листинг 3: Просмотр папки с исходными кодами ядра
ls -l /usr/src/linux
lrwxrwxrwx 1 root root 12 Oct 13 11:04 /usr/src/linux -> linux-2.6.12-gentoo-r10

Теперь настало время настройки и компиляции ядра. Можно использовать сценарий **genkernel**, который сформирует стандартное ядро, аналогичное используемому на установочном компакт-диске. Однако сначала мы расскажем о «ручной» настройке ядра, так как это лучший из способов оптимизации системы.

Если вы собираетесь настроить ядро вручную, то переходите к разделу по умолчанию: ручная настройка. Если вы хотите использовать genkernel, то читайте альтернатива: использование genkernel.

7.c. По умолчанию: ручная настройка

Введение

Ручная настройка ядра считается одним из самых сложных вопросов для пользователей Linux. Зато после настройки нескольких вариантов ядра вы и не вспомните, что это когда-то было трудно ;)

Однако есть одна непреложная истина: запуская ручную настройку ядра, следует знать состав аппаратуры своего компьютера. Большую часть информации можно собрать, установив пакет **pciutils** (**emerge pciutils**), в который входит **lspci**. **lspci** можно использовать, находясь в среде с временным корневым каталогом. Можно не обращать внимания на предупреждения, касающиеся **pcilib** (например: **pcilib: cannot open /sys/bus/pci/devices [pcilib: не могу открыть /sys/bus/pci/devices]**), которые выводит на экран **lspci**. Можно запустить **lspci** и вне временной среды установки. Результат тот же. Вы также можете запустить **lsmod** для просмотра модулей ядра, используемых установочным компакт-диском (это даст хорошую подсказку, какие параметры включать).

Теперь войдите в каталог с исходными кодами ядра и введите команду **make menuconfig**. Этой командой вы вызовете меню настройки, использующее **ncurses**.

Листинг 4: Вызов меню настройки

```
# cd /usr/src/linux
# make menuconfig
```

Перед вами появятся несколько разделов настройки. Сначала перечислим пункты, включение которых обязательно (иначе Gentoo не заработает, или для его работы потребуются дополнительные ухищрения).

Включение нужных пунктов

Прежде всего, разрешите использование разрабатываемого и экспериментального кода и драйверов. Без этого не появятся некоторые важные пункты:

Листинг 5: Выбор экспериментальных кодов/драйверов, основные параметры

```
Code maturity level options --->
[*] Prompt for development and/or incomplete code/drivers"
General setup --->
[*] Support for hot-pluggable devices
```

Удостоверьтесь, что каждый драйвер, жизненно необходимый для загрузки системы (например, SCSI-контроллера), собран внутри ядра, а не как модуль, иначе система совсем не сможет загрузиться.

Далее выберите нужное семейство процессоров:

Листинг 6: Выбор подходящего семейства процессоров

```
Processor type and features --->
(измените в соответствии со своей системой)
(Athlon/Duron/K7) Processor family
```

Теперь войдите в **File Systems** и выберите поддержку всех нужных файловых

систем. Не компилируйте их как модули, иначе система Gentoo не сможет смонтировать дисковые разделы. Также выберите Virtual memory и /proc file system. При использовании ядра 2.4, вам потребуется включить /dev file system, т.к. это ядро не поддерживает udev.

Листинг 7: Выбор необходимых файловых систем

(при использовании ядра 2.4.x)

File systems --->

- ☒ Virtual memory file system support (former shm fs)
- ☒ /proc file system support
- ☒ /dev file system support (EXPERIMENTAL)
- ☒ automatically mount /dev at boot
- ☐ /dev/pts file system for Unix98 PTYs

(при использовании ядра 2.6.x)

File systems --->

Pseudo Filesystems --->

- ☒ /proc file system support
- ☒ Virtual memory file system support (former shm fs)

(включите поддержку всех нужных вам файловых систем)

- <*> Reiserfs support
- <*> Ext3 journalling file system support
- <*> JFS filesystem support
- <*> Second extended fs support
- <*> XFS filesystem support

Если ваш **BIOS** не поддерживает жесткие диски большого размера, и вы ограничили переключками сообщаемый диском объем, то для получения доступа ко всему диску нужно включить следующие параметры ядра:

Листинг 8: Включение поддержки автокоррекции геометрии

(только в ядре 2.4.x)

ATA/IDE/MFM/RLL support --->

IDE, ATA and ATAPI Block devices --->

- <*> Include IDE/ATA-2 DISK support
- ☐ Use multi-mode by default
- ☒ Auto-Geometry Resizing support

Не забудьте включить поддержку режима **DMA** для ваших дисков:

Листинг 9: Включение DMA

Device Drivers --->

ATA/ATAPI/MFM/RLL support --->

- ☒ Generic PCI bus-master DMA support
- ☒ Use PCI DMA by default when available

Если вы используете **PPPoE** или коммутируемого соединения для соединения с интернетом, потребуется включить в ядре следующие функции:

Листинг 10: Выбор нужных драйверов PPPoE

(при использовании ядра 2.4.x)

Network device support --->

```
<*> PPP (point-to-point protocol) support
<*>  PPP support for async serial ports
<*>  PPP support for sync tty ports
```

(при использовании ядра 2.6.x)

Device Drivers --->

Networking support --->

```
<*> PPP (point-to-point protocol) support
<*>  PPP support for async serial ports
<*>  PPP support for sync tty ports
```

Два параметра сжатия данных не являются обязательными, но и не повредят. То же относится и к параметру **PPP over Ethernet** который нужен только для гр-рррое, для поддержки **PPPoE** в режиме ядра.

При необходимости, не забудьте включить в ядре поддержку своей сетевой платы.

Если у вас процессор Intel с поддержкой HyperTreading или многопроцессорная система, следует включить «симметричную многопроцессорную обработку»:

Листинг 11: Включение поддержки SMP

Processor type and features --->

```
[*] Symmetric multi-processing support
```

Если вы пользуетесь устройствами ввода на шине USB, (например, мышью или клавиатурой), не забудьте включить их поддержку:

Листинг 12: Включение поддержки USB для устройств ввода

USB Support --->

```
<*>  USB Human Interface Device (full HID) support
```

Владельцам ноутбуков, которым нужна поддержка **PCMCIA**, не нужно включать драйверы **PCMCIA** в ядре, выбрав ядро версии 2.4. В пакете pcmcia-cs, который будет устанавливаться позже, есть более свежие драйверы. Пользователям ядра 2.6 следует использовать драйверы **PCMCIA**, включенные в ядро.

Наряду с включением поддержки **PCMCIA** в ядре 2.6, не забудьте включить поддержку моста PCMCIA своей системы:

Листинг 13: Включение поддержки PCMCIA для ядра 2.6

Bus options (PCI, PCMCIA, EISA, MCA, ISA) --->

PCCARD (PCMCIA/CardBus) support --->

```
<*> PCCard (PCMCIA/CardBus) support
```

(если нужно (обычно нужно), включите поддержку старых 16-битных карт PCMCIA)

```
<*> 16-bit PCMCIA support
```

```
[*] 32-bit CardBus support
```

(выберите наиболее подходящий мост)

```
--- PC-card bridges
```

```
<*> CardBus yenta-compatible bridge support (NEW)
```

```
<*> Cirrus PD6729 compatible bridge support (NEW)
<*> i82092 compatible bridge support (NEW)
<*> i82365 compatible bridge support (NEW)
<*> Databook TCIC host bridge support (NEW)
```

Закончив настройку ядра, приступайте к компиляции и установке.

Компиляция и установка

Теперь, когда ядро настроено, настало время его скомпилировать и установить. Выйдите из настройки и запустите процесс компиляции:

Листинг 14: Компиляция ядра
(для ядра 2.4)
`# make dep && make bzImage modules modules_install`

(для ядра 2.6)
`# make && make modules_install`

По завершении компиляции, скопируйте образ ядра в каталог /boot. Используйте любое имя, которое вам покажется подходящим для избранного ядра, и запомните его, так как позже при настройке начального загрузчика оно потребуется. Не забудьте заменить <версия-ядра> на имя и версию своего ядра.

Листинг 15: Установка ядра
`# cp arch/i386/boot/bzImage /boot/<версия-ядра>`

Теперь переходите к модулям ядра.

7.d. Альтернатива: использование genkernel

Если вы читаете эти строки, значит, вы собираетесь использовать наш сценарий **genkernel** для настройки ядра за вас.

Теперь когда исходные коды ядра установлены, настало время скомпилировать ядро, собрав его автоматически при помощи сценария genkernel. genkernel выполняет настройку ядра практически так же, как настраивается ядро на установочном компакт-диске. То есть система, ядро которой собрано genkernel, как правило, будет выполнять определение всех устройств при загрузке. Поскольку genkernel не требует никакой ручной настройки ядра, это идеальное решение для тех, кому неуютно от одной мысли, что придется компилировать ядро.

Рассмотрим, как использовать **genkernel**. Сначала установите его:

Листинг 16: Установка genkernel
emerge genkernel

Теперь, если вы собираетесь использовать ядро ветви 2.6, скопируйте настройку ядра, используемую установочным диском, в каталог, где genkernel ищет настройку по умолчанию:

Листинг 17: Копирование файла настройки ядра с установочного диска
 (только если вы собираетесь настраивать ядро 2.6.x)
zcat /proc/config.gz > /usr/share/genkernel/x86/kernel-config-2.6

Теперь скомпилируйте исходный код ядра, запустив genkernel all. Имейте в виду, что компиляция займет приличное время, поскольку genkernel собирает ядро, поддерживающее практически любое оборудование.

Если в загрузочном разделе не используется файловая система ext2 или ext3, то вам придется вручную настроить ядро, запустив genkernel --menuconfig all, добавив поддержку используемой файловой системы в ядро (т.е. не модулем). Пользователям EVMS2 или LVM2, вероятно, захочется также добавить параметр командной строки --evms2 или --lvm2.

Листинг 18: Запуск genkernel
genkernel all

В итоге работы **genkernel** будет создано ядро, полный набор модулей и начальный корневой диск (**initrd**). Ядро и **initrd** будут использоваться в процессе настройки загрузчика системы, которая описана далее в руководстве. Запишите имена ядра и начального корневого диска, так как они понадобятся при составлении конфигурационного файла загрузчика. **initrd** запускается сразу при начальной загрузке, выполняя автоматическое определение устройств (точно так же, как при запуске установочного диска), до запуска «настоящей» системы.

Листинг 19: Уточнение названия созданного образа ядра и initrd
ls /boot/kernel* /boot/initramfs*

Теперь давайте еще на шаг приблизим нашу систему к установочному диску: соберем **coldplug**. Если **initrd** определяет оборудование, необходимое для загрузки системы, **coldplug** выполняет автоопределение всех остальных устройств. Для установки и подключения coldplug введите команду:

Листинг 20: Установка и подключение coldplug
emerge coldplug
rc-update add coldplug boot

7.e. Модули ядра

Настройка модулей

Модули, которые требуется загружать автоматически, нужно указать в `/etc/modules.autoload.d/kernel-2.4` (или `kernel-2.6`). Также, при желании, модулям можно сообщить дополнительные параметры.

Для просмотра всех доступных модулей запустите команду `find`, заменив «<версия ядра>» на версию только что собранного ядра:

Листинг 21: Просмотр перечня доступных модулей
`# find /lib/modules/<kernel version>/ -type f -iname '*.o' -or -iname '*.ko'`

Например, для автоматической загрузки модуля `3c59x.o` измените файл `kernel-2.4` или `kernel-2.6`, указав в нем имя модуля.

Листинг 22: Изменение `/etc/modules.autoload.d/kernel-2.4`
(показан пример для ядра 2.4.x)
`# nano -w /etc/modules.autoload.d/kernel-2.4`

Листинг 23: `/etc/modules.autoload.d/kernel-2.4` или `kernel-2.6`
`3c59x`

Теперь переходите к настройке параметров системы.

8. Настройка параметров системы

8.a. Параметры файловых систем

Что такое `fstab`?

В Linux все разделы, используемые системой, должны быть перечислены в `/etc/fstab`. В этом файле указываются точки подключения разделов (mountpoints, местоположение разделов в файловой системе), порядок подключения, а также дополнительные параметры (автоматический или ручной режим подключения, достаточность прав пользователя для подключения и т.п.)

Создание `/etc/fstab`

В `/etc/fstab` используется специальный формат. Каждая строка состоит из шести полей, разделяемых пробелами, знаками табуляции или их сочетанием. Каждое поле имеет свое назначение:

1. Первое поле обозначает раздел (partition) (путь к файлу устройства).
2. Второе поле указывает точку подключения (mountpoint), в которую монтируется раздел.
3. Третье поле задает тип файловой системы (filesystem), используемой в разделе.
4. В четвертом поле указываются параметры подключения (mountoptions), используемые mount при подключении раздела. Поскольку для каждой файловой системы существуют свои параметры, рекомендуется прочитать страницу справки по mount (man mount), где приведен их полный перечень. При указании нескольких параметров подключения их следует разделять запятыми.
5. Пятое поле используется dump для определения, требуется ли резервное копирование раздела средствами dump. Обычно это поле можно просто установить в 0 (ноль).
6. Шестое поле используется **fsck** для определения порядка проверки (check) файловых систем после некорректного завершения работы системы. Для корневой файловой системы рекомендуется значение 1, а для остальных — 2 (или 0, когда проверка файловой системы не требуется).

Важно: Вариант файла **/etc/fstab** по умолчанию, входящий в Gentoo, не является работоспособным. Вам потребуется создать свой собственный **/etc/fstab**.

Листинг 1: Открытие /etc/fstab
nano -w /etc/fstab

Укажите правила, соответствующие вашей схеме разбивки, и добавьте правила для **/proc**, для **tmpfs**, для своих дисководов CD-ROM (если есть другие разделы или устройства, их тоже можно указать).

Теперь на основе приведенного примера создайте собственный файл **/etc/fstab**:

Параметр **auto** позволяет mount определять тип файловой системы автоматически (рекомендуется для съемных носителей, которые могут оказаться размечены в одной из множества существующих файловых систем), а **user** позволяет монтировать компакт-диски обычным пользователям.

Чтобы повысить быстродействие, большинству пользователей стоит добавить параметр **noatime** в параметры подключения, что приведет к ускорению за счет отключения регистрации отметки времени доступа к файлам (обычно в ней все равно нет необходимости):

Перепроверьте свой файл **/etc/fstab**, сохраните его, и выйдите из редактора, чтобы продолжить настройку.

Примечание (JohnBat26): пример моего /etc/fstab:**Листинг 2: Пример /etc/fstab**

```

/dev/sda2          /boot            ext2    defaults,noatime          1 2
/dev/sda5          /                xfs     defaults,noatime          0 1
/dev/sda3          none            swap    sw                        0 0
/dev/sda7          /home           xfs     defaults,noatime          0 1
/dev/sr0           /mnt/cdrom      auto    noauto,ro,user,icharset=utf8 0 0
# раздел с FAT32
/dev/sda1          /windows/c      vfat    auto,icharset=utf8,quiet,user,umask=000,noatime 0 0
# раздел NTFS только для чтения
/dev/sda6          /windows/d      ntfs     auto,nls=utf8,umask=0,user,noatime          0 0
/dev/sda8          /windows/e      vfat    auto,icharset=utf8,user,quiet,umask=000,noatime 0 0
/dev/sda9          /home/media     xfs     defaults,noatime          0 1
# раздел NTFS для чтения и записи при помощи ntfs-3g
/dev/sda10         /mnt/windows/f  ntfs-3g  -o silent,umask=0,locale=ru_RU.utf8
# NOTE: The next line is critical for boot!
proc              /proc           proc     defaults                  0 0
# flash drive
/dev/sdb           /mnt/flash      auto    user,noauto,rw,async,icharset=utf8          0 0
shm               /dev/shm        tmpfs    nodev,nosuid,noexec       0 0

```

8.b. Параметры сети**Hostname, Domainname и т. д.**

Еще один вопрос, который нужно решить пользователю — как назвать свой компьютер. Он кажется довольно простым, но многие затрудняются дать подходящее имя для своей Linux-системы. Чтобы вам стало легче, запомните, что какое бы имя вы не выбрали, потом его всегда можно изменить. Например, вы могли бы просто назвать свою систему tux, а домен — homenetwork.

Листинг 2: Установка имени узла
nano -w /etc/conf.d/hostname

(присвойте переменной HOSTNAME имя своего узла)
HOSTNAME="tux"

Во-вторых, установим имя домена (domainname) в /etc/conf.d/net:

Листинг 3: Установка имени домена
nano -w /etc/conf.d/net
 (присвойте переменной DNSDOMAIN имя своего домена)
dns_domain_lo="homenetwork"

Если у вас есть домен NIS (а если вы не знаете, что это такое, то у вас его точно нет), его также необходимо указать:

Листинг 4: Установка имени NIS-домена
nano -w /etc/conf.d/net

(укажите название своего домена NIS переменной `nis_domain`)
`nis_domain_lo="my-nisdomain"`

Настройка сети

Прежде, чем возмутиться: «Эй, мы же все это уже делали!» — вспомните, что подключение к сети, настроенное вначале, было предназначено лишь для установки Gentoo. Теперь же вы настраиваете сеть для постоянного использования.

Примечание: Более подробные сведения о сетях, включая дополнительные темы, такие как объединение, образование мостов, настройка виртуальных сетей (VLAN) 802.1Q или беспроводных сетей, представлены в разделе настройка сети в Gentoo.

Все настройки сети собраны в файле `/etc/conf.d/net`. В нем используется простой формат, хотя, если вы не знакомы с ручной настройкой сети, он не слишком очевиден. Но не бойтесь, мы все объясним. В файле `/etc/conf.d/net.example` приведен подробно прокомментированный пример, охватывающий много различных конфигураций.

По умолчанию используется **DHCP**. Чтобы **DHCP** заработал, требуется установить **DHCP**-клиент, как описано далее в разделе Установка нужных системных средств. Не забудьте установить DHCP-клиент.

Если настройка сетевого подключения нужна вам для указания специфических параметров DHCP, или из-за того, что вы вообще не используете DHCP, откройте `/etc/conf.d/net` в своем любимом редакторе (в этом примере использован `nano`):

Листинг 5: Открытие `/etc/conf.d/net` для изменения
`# nano -w /etc/conf.d/net`

Вы увидите следующее:

Листинг 6: `/etc/conf.d/net` по умолчанию
`# This blank configuration will automatically use DHCP for any net.*`
`# scripts in /etc/init.d. To create a more complete configuration,`
`# please review /etc/conf.d/net.example and save your configuration`
`# in /etc/conf.d/net (this file :!)).`

(# Этот пустой файл настройки приводит к автоматическому использованию
 # DHCP всеми сценариями `net.*` из `/etc/init.d`. Для создания более полной
 # настройки, пожалуйста, просмотрите `/etc/conf.d/net.example`, а свою
 # настройку сохраните в `/etc/conf.d/net` (в этом файле :!)).)

Чтобы указать свой собственный адрес IP, маску сети и шлюз, потребуется настроить как `config_eth0`, так и `routes_eth0`:

Листинг 7: Ручная настройка параметров IP для eth0

```
config_eth0=( "192.168.0.2 netmask 255.255.255.0 brd 192.168.0.255" )  
routes_eth0=( "default gw 192.168.0.1" )
```

Чтобы при использовании DHCP указать специфические параметры, определите **config_eth0** и **dhcp_eth0**:

Листинг 8: Автоматическое получение адреса IP для eth0

```
config_eth0=( "dhcp" )  
dhcp_eth0="nodns nontp nonis"
```

Список допустимых параметров дан в файле `/etc/conf.d/net.example`.

Если у вас несколько сетевых интерфейсов, повторите эти шаги для **config_eth1**, **config_eth2** и т.д.

Теперь сохраните параметры и выйдите из редактора, чтобы продолжить настройку.

Автоматический запуск сетевого подключения при загрузке

Для запуска сетевых интерфейсов при загрузке необходимо добавить их в уровень запуска по умолчанию. Если у вас интерфейсы типа PCMCIA, пропустите этот шаг, поскольку интерфейсы PCMCIA запускаются сценарием инициализации PCMCIA.

Листинг 9: Добавление net.eth0 в уровень запуска default

```
# rc-update add net.eth0 default
```

Если у вас несколько сетевых интерфейсов, потребуется создать для них соответствующие сценарии инициализации net.eth1, net.eth2 и т.д. Для этого можно использовать ln:

Листинг 10: Создание дополнительных сценариев инициализации

```
# cd /etc/init.d  
# ln -s net.lo net.eth1  
# rc-update add net.eth1 default
```

Указание сетевых узлов

Теперь расскажем системе Linux о вашей сети. Эти сведения указываются в `/etc/hosts`, и помогают разрешению имен в IP-адреса для узлов, не обрабатываемых сервером имен. Требуется определить вашу систему. Также можно определить другие системы в сети, если вы не собираетесь устанавливать собственную систему DNS.

Листинг 11: Открытие `/etc/hosts`

```
# nano -w /etc/hosts
```

Листинг 12: Указание сведений об узлах сети
(определение текущей системы)

```
127.0.0.1    tux.homenetwork tux localhost
```

(определите другие машины в своей сети,
для этого у них должен быть статический IP-адрес.)

```
192.168.0.5  jenny.homenetwork jenny
```

```
192.168.0.6  benny.homenetwork benny
```

Чтобы продолжить настройку, сохраните файл и выйдите из редактора.

8.с. Параметры системы

Пароль root

Прежде всего, нужно установить пароль root (администратора), набрав:

Листинг 13: Установка пароля root
passwd

Если вы хотите, чтобы root мог входить в систему через последовательный терминал, добавьте `tts/0` в `/etc/securetty`:

Листинг 14: Добавление `tts/0` to `/etc/securetty`
echo "tts/0" >> /etc/securetty

Параметры системы

Для общей настройки системы в Gentoo используется `/etc/rc.conf`. Откройте `/etc/rc.conf` и с удовольствием прочитайте все комментарии, находящиеся в этом файле :)

Листинг 15: Открытие `/etc/rc.conf`
nano -w /etc/rc.conf

Завершив изменение `/etc/rc.conf` сохраните файл и выйдите из редактора.

Как видите, этот файл подробно прокомментирован, что поможет вам в настройке необходимых конфигурационных переменных. Можно настроить систему на использование **unicode**, а также указать редактор по умолчанию и диспетчер отображения (например, **gdm** или **kdm**).

Для управления раскладками клавиатуры в Gentoo используется `/etc/conf.d/keymaps`. Для настройки своей клавиатуры измените его.

Листинг 16: Открытие `/etc/conf.d/keymaps`
nano -w /etc/conf.d/keymaps

Будьте особенно тщательны при установке переменной раскладки клавиатуры (KEYMAP): выбрав неверную раскладку, вы можете получить непредсказуемый результат при попытке ввода с клавиатуры.

По завершении изменения `/etc/conf.d/keymaps` сохраните файл и выйдите из редактора.

Для настройки параметров часов в Gentoo используется `/etc/conf.d/clock`. Изменяйте его в соответствии со своими потребностями.

Если аппаратные часы вашей системы настроены не на часовой пояс UTC (Гринвич), в файл необходимо добавить строку **CLOCK="local"**. В противном случае вы заметите сдвиг часового пояса.

После завершения настройки `/etc/conf.d/clock` сохраните файл и выйдите из редактора.

Переходите к установке нужных системных средств.

9. Установка нужных системных средств

9.a. Диспетчер устройств

Если вы используете ядро 2.4, и при этом устанавливаете Gentoo из файла третьей стадии (stage3), вам необходимо еще кое-что сделать. Так как теперь в Gentoo по умолчанию используется udev, а ядром 2.4 udev не поддерживается, вам потребуется установить devfsd, и убрать udev.

Листинг 1: Установка devfsd
(для тех, кто использует ядро 2.4.x при установке с третьей стадии)
emerge --unmerge udev
emerge devfsd

9.b. Системные службы журналирования

Некоторые средства не включены в архив stage3, поскольку одинаковые возможности можно обеспечить различными пакетами. Теперь вы сами выберете, какие именно установить.

Первый инструмент, который вам необходимо выбрать, должен дать системе возможность журналирования. У Unix и Linux превосходная история развития

журналирования — при желании в файлах журналов можно регистрировать любой чих, происходящий в вашей системе. Это обеспечивается системной службой журналирования.

В Gentoo предлагается несколько служб журналирования на выбор. Это **sysklogd**, традиционный набор журналирующих демонов, **syslog-ng**, расширенная служба журналирования, и **metalog** — служба журналирования с очень гибкими возможностями настройки. Возможно, в Portage имеются и другие службы журналирования: количество доступных пакетов растет день ото дня.

Если вы планируете использовать **sysklogd** или **syslog-ng**, позднее может потребоваться установка **logrotate**, поскольку в этих службах журналирования не предусмотрен никакой механизм ротации системных журналов.

Чтобы установить выбранную службу журналирования, воспользуйтесь **emerge**, а затем добавьте ее в уровень запуска по умолчанию с помощью **rc-update**. В следующем примере показана установка **syslog-ng**. Вместо нее вы можете подставить другую службу журналирования:

Листинг 2: Установка системной службы журналирования

```
# emerge syslog-ng
# rc-update add syslog-ng default
```

9.с. Дополнительно: демон cron

Следующий демон — **cron**. Хотя он является дополнительным, и не обязателен для работы вашей системы, будет благоразумным установить его. Но что же такое демон **cron**? Демон **cron** выполняет команды по расписанию. Это очень удобно, когда нужно выполнять какие-либо команды регулярно (например, ежедневно, еженедельно или ежемесячно).

В Gentoo предлагаются три варианта демона **cron** на выбор: **dcron**, **fcron** и **vixie-cron**. Установка любого из них подобна установке системной службы журналирования. Однако, при установке **dcron** или **fcron** для настройки требуется выполнение дополнительной команды, а именно **crontab /etc/crontab**. Если вы не знаете, что выбрать, используйте **vixie-cron**.

При бессетевой установке доступен только **vixie-cron**. Если вам нужен другой демон **cron**, его можно установить позже.

Листинг 3: Установка демона **cron**

```
# emerge vixie-cron
# rc-update add vixie-cron default
(только если выбран dcron или fcron) # crontab /etc/crontab
```

9.d. Дополнительно: индексация файлов

Если вы хотите индексировать файлы в своей системе, чтобы быстро находить их с помощью locate, нужно установить sys-apps/slocate:

Листинг 4: Установка slocate
emerge slocate

9.e. Утилиты для обслуживания файловых систем

Для проверки целостности файловых систем, создания дополнительных файловых систем, и т.п., вам потребуются определенные утилиты, состав которых зависит от используемых файловых систем.

В следующей таблице перечислены утилиты, которые необходимо устанавливать для обслуживания файловых систем различных типов:

Файловая система	Утилиты	Команда установки
XFS	xfspggs	emerge xfspggs
ReiserFS	reiserfspggs	emerge reiserfspggs
JFS	jfspggs	emerge jfspggs

Если вы используете EVMS, также необходимо установить evms:

Листинг 5: Установка утилит EVMS
USE="-gtk" emerge evms

USE="-gtk" предотвратит установку пакетов, от которых зависит данный. При желании включить графические средства evms, потом можно перекомпилировать evms.

Если вам не нужны никакие дополнительные сетевые средства (типа **ppp** или клиента dhcp), переходите к настройке начального загрузчика.

9.f. Сетевые средства

Дополнительно: установка клиента DHCP

Если требуется, чтобы Gentoo автоматически получала IP-адрес для ваших сетевых интерфейсов, необходимо установить dhcpd (или любой другой клиент DHCP, список имеющихся клиентов DHCP см. в главе Модульное построение сети). Если не сделать этого сейчас, то после завершения установки вы не сможете подключиться к интернету!

Листинг 6: Установка dhcpd
emerge dhcpd

Дополнительно: установка клиента PPPoE

Если для выхода в сеть требуется gr-pppoe, его нужно установить.

Листинг 7: Установка gr-pppoe
USE="-X" emerge gr-pppoe

USE="-X" запрещает установку xorg-x11 в порядке зависимости (в gr-pppoe есть графические средства; если их нужно подключить, можно перекомпилировать gr-pppoe позже, или же установить xorg-x11 сейчас: при его установке потребуются много времени на компиляцию).

Дополнительно: утилиты RAID для оборудования IBM

Если в составе системы на базе POWER5 вы используете RAID-массивы SCSI, стоит задуматься об установке iprutils, которые, среди прочего, позволяют работать с дисковыми массивами, выяснять состояние дисков в составе массива и обновлять микрокод.

Листинг 8: Установка iprutils
emerge iprutils

Теперь переходите к настройке начального загрузчика.

10. Настройка начального загрузчика

10.а. Выбор загрузчика

Введение

Теперь, когда ядро настроено и собрано, а нужные конфигурационные файлы заполнены как надо, пришло время установить программу, которая будет запускать ваше ядро при старте системы. Такого рода программа называется загрузчиком. Для архитектуры x86 в Gentoo Linux есть загрузчики GRUB и LILO. Но прежде, чем приступить к установке одного из двух загрузчиков, мы расскажем, как настроить кадровый буфер (естественно, если вы захотите). С помощью кадрового буфера можно работать в командной строке Linux на фоне графических элементов (например, симпатичного изображения из загрузочной заставки Gentoo).

Дополнительно: кадровый буфер

Если вы настроили в ядре поддержку кадрового буфера (или оставили настройки по умолчанию при использовании **genkernel**), вы можете включить буфер,

добавив параметр `vga` и/или `video` в файл конфигурации своего загрузчика.

Для начала, вам надо узнать тип используемого кадрового буфера. При использовании исходных кодов ядра, доработанных для Gentoo (таких как `gentoo-sources`), у вас есть возможность выбрать `vesafb-tng` в качестве типа драйвера VESA (в этих исходных кодах ядра он используется по умолчанию). При использовании `vesafb-tng` параметр `vga` указывать не нужно. В других случаях используется драйвер `vesafb`, а параметр `vga` необходимо указывать.

Параметр `vga` устанавливает разрешение и глубину цвета, используемые кадровым буфером **vesafb**. Как отмечено в `/usr/src/linux/Documentation/fb/vesafb.txt` (который устанавливается в составе пакета с деревом исходных кодов ядра), кадровому буферу необходимо передавать код режима VESA, соответствующий нужному разрешению и цветности.

В следующей таблице приведены коды режимов для указания в параметре `vga`, а также соответствующие им значения разрешения и цветности.

	640x480	800x600	1024x768	1280x1024
256	0x301	0x303	0x305	0x307
32 тыс	0x310	0x313	0x316	0x319
64 тыс	0x311	0x314	0x317	0x31A
16 млн	0x312	0x315	0x318	0x31B

Параметр **video** отвечает за порядок отображения кадрового буфера. В нем указывается название драйвера кадрового буфера (`vesafb` для ядра 2.6 или `vesa` для ядра 2.4), а затем режимы, которые нужно включить. Все режимы приведены в `/usr/src/linux/Documentation/fb/vesafb.txt`, а здесь мы расскажем о трех самых используемых:

1. **ywrap** считать, что видеоплата может закольцовывать свою память (продолжать выборку с начального адреса, когда достигнут конечный);
2. **mtrr:n** установка регистров MTRR; допустимые значения `n`: 0 - отключено 1 - без кэширования 2 - отложенная запись (`write-back`) 3 - объединенная запись (`write-combining`) 4 - сквозная запись (`write-through`);
3. **mode** (только для `vesafb-tng`) Установить разрешение, цветность и частоту кадров. Например, `1024x768-32@85` для разрешения 1024x768, 32-битного цвета и частоты кадров 85 Гц.

В результате должно получиться что-то вроде **`vga=0x318 video=vesafb:mtrr:3,ywrap`** или **`video=vesafb:mtrr:3,ywrap,1024x768-32@85`**. Запомните (или запишите) составленное значение, скоро оно понадобится.

Перейдем к установке GRUB или LILO.

10.b. По умолчанию: использование GRUB

Введение в терминологию GRUB

Самое сложное в освоении **GRUB** — освоиться с тем, как в нем именуются жесткие диски и разделы. Ваш Linux-раздел `/dev/hda1`, скорее всего, в **GRUB** станет называться **(hd0,0)**. Обратите внимание на круглые скобки вокруг **hd0,0** — они **обязательны**.

Жесткие диски нумеруются, начиная с нуля, а не с «а»; разделы — с нуля, а не с единицы. Нужно помнить, что в виде `hd` нумеруются только жесткие диски, но не устройства `atapi-ide`, такие как приводы компакт-дисков. Та же нумерация используется для устройств `SCSI` (обычно им присваиваются номера большие, чем устройствам `IDE`, кроме случаев, когда BIOS настроен на загрузку с устройства `SCSI`). Когда BIOS настроен на загрузку с другого жесткого диска (например, с первичного ведомого), именно этот жесткий диск и становится `hd0`.

Например, если у вас есть жесткий диск `/dev/hda`, привод CD-ROM `/dev/hdb`, записывающий CD `/dev/hdc`, второй жесткий диск `/dev/hdd`, а устройств `SCSI` нет, то разделу `/dev/hdd7` будет соответствовать **(hd1,6)**. Возможно, это покажется запутанным (так и есть), но, как мы увидим, в **GRUB** есть механизм автодополнения по `tab`, облегчающий жизнь обладателям множества жестких дисков и разделов, а также тем, кто теряется в схеме нумерации устройств **GRUB**.

Почувствовав, что к чему, пора установить **GRUB**.

Установка GRUB

Для установки **GRUB** сначала добавим его в систему:

Листинг 1: Установка **GRUB**
`# emerge grub`

Хотя **GRUB** уже установлен, нам еще потребуется подправить его файл конфигурации, и поместить **GRUB** в MBR, чтобы он автоматически загружал ядро. С помощью `nano` (или другого редактора) создайте `/boot/grub/grub.conf`:

Листинг 2: Создание `/boot/grub/grub.conf`
`# nano -w /boot/grub/grub.conf`

Теперь заполним `grub.conf` своими значениями. Ниже приведены два варианта `grub.conf` для показанного примера разбиения дисков. Первый вариант `grub.conf` мы подробно прокомментировали. Удостоверьтесь, что у себя вы указываете имя своего файла образа ядра, и при необходимости имя своего образа начального корневого диска (`initrd`).

первый вариант `grub.conf` — для тех, кто при сборке ядра обходился без `genkernel`
 второй вариант `grub.conf` — для тех, кто при сборке ядра пользовался `genkernel`

Примечание: Если ваша корневая файловая система — **JFS**, необходимо

добавить «ro» в строку kernel, поскольку JFS «накатывает» свой журнал перед тем, как разрешить монтирование раздела на чтение-запись.

Листинг 3: grub.conf для тех, кто обошелся без genkernel

какой пункт загружать по умолчанию: 0 - первый, 1 - второй и т.д.

default 0

сколько секунд ждать до начала загрузки пункта по умолчанию

timeout 30

симпатичная заставка, добавить по вкусу :)

закомментируйте, если у вас не установлена графическая видеоплата

splashimage=(hd0,0)/boot/grub/splash.xpm.gz

title=Gentoo Linux 2.6.12-r10

раздел с файлом образа ядра (или операционной системой)

root (hd0,0)

kernel /boot/kernel-2.6.12-gentoo-r10 root=/dev/hda3

следующие четыре строки нужны только для двойной загрузки с Windows

в этом примере Windows находится на /dev/hda6

title=Windows XP

rootnoverify (hd0,5)

makeactive

chainloader +1

Листинг 4: grub.conf для тех, кто пользовался genkernel

default 0

timeout 30

splashimage=(hd0,0)/boot/grub/splash.xpm.gz

title=Gentoo Linux 2.6.12-r10

root (hd0,0)

kernel /boot/kernel-genkernel-x86-2.6.12-gentoo-r10 root=/dev/ram0 init=/linuxrc ramdisk=8192

real_root=/dev/hda3 udev

initrd /boot/initramfs-genkernel-x86-2.6.12-gentoo-r10

нужно только для двойной загрузки

title=Windows XP

rootnoverify (hd0,5)

makeactive

chainloader +1

Примечание: Параметр udev, указанный в конце строки kernel, необходим для обхода ошибки в некоторых версиях genkernel, если вы вообще используете udev (по умолчанию — используется).

Примечание: Если вы разбили жесткий диск по-другому, или у вас другое ядро, внесите необходимые изменения. При этом убедитесь, что все пути, следующие за упоминанием устройства GRUB (например **(hd0,0)**), приведены относительно точки подключения, а не корня файловой системы. Другими словами, **(hd0,0)/grub/splash.xpm.gz** — на самом деле **/boot/grub/splash.xpm.gz**, так как **(hd0,0)** — это **/boot**.

Кроме того, если вы избрали другую схему разбиения диска, и не выделяли для /boot отдельный раздел, префикс /boot, использованный в примерах выше, обязателен. Если же вы следовали рекомендованному нами плану разбиения, префикс /boot не требуется, но все работает благодаря символической ссылке boot. Короче говоря, приведенные примеры должны работать независимо от того, есть у вас отдельный раздел для /boot или нет.

Если вам надо передать ядру дополнительные параметры, просто добавьте их в конец строки kernel. Один параметр мы уже передаем ядру (root=/dev/hda3 или real_root=/dev/hda3); можно добавлять и другие, например, такие как параметры video и/или vga для кадрового буфера, обсуждавшиеся выше.

Если вы используете ядро 2.6.7 или выше, а объем жесткого диска ограничили перемычками из-за того, что BIOS не в состоянии работать с дисками большого размера, вам потребуется добавить hdx=stroke.

Тем, кто использует **genkernel**, нужно помнить, что их ядро использует такие же загрузочные параметры, как на установочном компакт-диске. Например, если у вас есть устройства SCSI, следует передать ядру параметр doscsi.

Теперь сохраните grub.conf и выйдите из редактора. Вам по-прежнему необходимо записать GRUB в MBR (Master Boot Record), чтобы GRUB автоматически запускался при загрузке системы.

Разработчики GRUB рекомендуют использовать grub-install. Однако, на случай некорректной работы grub-install есть возможность записать GRUB вручную.

Переходите к разделу по умолчанию: установка GRUB с помощью grub-install или альтернатива: установка GRUB вручную.

По умолчанию: установка GRUB с помощью grub-install

Для установки GRUB вам надо выполнить команду grub-install. Однако, grub-install не заработает сам по себе, т.к. мы находимся в среде с измененным корневым каталогом. Нам нужно создать файл /etc/mtab, перечислив в нем все смонтированные файловые системы. К счастью, для этого есть очень легкий способ: просто скопируйте содержимое /proc/mounts поверх /etc/mtab, исключив строку rootfs, если вы не создавали отдельный загрузочный раздел. Следующая команда подойдет в обоих случаях:

Листинг 5: Создание /etc/mtab
grep -v rootfs /proc/mounts > /etc/mtab

Теперь мы можем установить GRUB, используя grub-install:

Листинг 6: Выполнение grub-install
grub-install /dev/hda

Если у вас есть вопросы о GRUB, пожалуйста, обратитесь к GRUB FAQ (англ.) или руководству по GRUB (англ.).

Переходите к перезагрузке системы.

Альтернатива: установка GRUB вручную

Для начала настройки, введите `grub`. Вы увидите приглашение `grub>` — это командная строка `grub`. Теперь потребуется набрать команды, нужные для установки загрузочной записи GRUB на ваш жесткий диск.

Листинг 7: Запуск оболочки GRUB
grub

Примечание: Если у вас нет приводов для дискет, к приведенной команде добавьте `--no-floppy`, чтобы `grub` зря не опрашивал несуществующие дисководы.

В приводимом примере мы хотим установить GRUB так, чтобы он считывал нужную информацию с загрузочного раздела `/dev/hda1`, а загрузочная запись GRUB находилась в MBR (Master Boot Record) жесткого диска, чтобы первое, что мы видели после включения компьютера — это приглашение GRUB. Естественно, если вы при установке отклонялись от предлагаемой схемы, внесите необходимые поправки.

Находясь в GRUB, можно использовать автодополнение по клавише TAB. К примеру, если ввести `<root>` (`<`), а затем TAB, появится список устройств (таких как `hd0`). Если ввести `<root (hd0,>` и нажать TAB, появится список для выбора раздела из возможных (такого как `hd0,0`).

Благодаря автодополнению установка GRUB не так сложна. Теперь приступим к настройке GRUB.

Листинг 8: Установка GRUB в MBR
grub> root (hd0,0) (указание расположения раздела с `/boot`)
grub> setup (hd0) (установка GRUB в MBR)
grub> quit (выход из оболочки GRUB)

Примечание: Если вы хотите установить GRUB в определенный раздел вместо MBR, команду `setup` потребуется исправить так, чтобы она указывала на нужный раздел. Например, команда для установки GRUB в `/dev/hda3` — `setup (hd0,2)`. Однако, так поступают немногие.

С дополнительными вопросами о GRUB, пожалуйста, обращайтесь к GRUB FAQ (англ.) или руководству по GRUB (англ.).

Примечание: В случае переустановки ядра в будущем, вам больше не потребуется копировать содержимое файлов. После компиляции ядра просто запускайте `make install`: копирование файлов и изменение конфигурации GRUB произойдет автоматически.

Переходите к перезагрузке системы.

10.c. Альтернатива: использование LILO

Установка LILO

LILO (сокращение от Linux LOader) — это проверенная временем рабочая лошадка среди загрузчиков Linux-систем. Но ей недостает ряда возможностей, которые есть в GRUB (и в том числе в этом заключается причина растущей популярности GRUB). LILO все еще используется, потому что на некоторых системах он работает, а GRUB — нет. Конечно же, он используется еще и потому, что многие просто знакомы с LILO и сроднились с ним. Так или иначе, в Gentoo поддерживаются оба загрузчика, и вы, видимо, решили использовать LILO.

Установка LILO в систему проста как пробка: просто используйте `emerge`.

Листинг 9: Установка LILO
`# emerge lilo`

Настройка LILO

Для настройки LILO нужно создать файл `/etc/lilo.conf`. Запустите свой любимый редактор (в руководстве мы для единообразия используем `nano`) и создайте файл.

Листинг 10: Создание `/etc/lilo.conf`
`# nano -w /etc/lilo.conf`

Несколькими разделами раньше мы попросили вас запомнить название созданного файла образа ядра. В следующем примере используется предложенная нами схема деления диска.

Пример разделен на две части:

- одна — для тех, кто не пользовался для сборки ядра `genkernel`;
- другая — для тех, кто при сборке ядра пользовался `genkernel`;

Удостоверьтесь, что у себя вы указываете имя своего файла образа ядра, и при необходимости имя своего образа начального корневого диска (`initrd`).

Примечание: Если ваша корневая файловая система — JFS, необходимо добавить «го» в строку `kernel`, поскольку JFS «накатывает» свой журнал перед тем, как разрешить монтирование раздела на чтение-запись.

Листинг 11: Пример `/etc/lilo.conf`
`boot=/dev/hda       # установка LILO в MBR`

```
prompt          # предоставление шанса выбора другого варианта
timeout=50       # ожидание пять секунд до загрузки варианта по умолчанию
default=gentoo   # по истечении времени загрузка варианта gentoo

# для тех, кто не использует genkernel
image=/boot/kernel-2.6.12-gentoo-r10
label=gentoo     # название этого варианта
read-only        # запуск с корневой ФС только для чтения; не менять!
root=/dev/hda3   # расположение корневой файловой системы

# для тех, кто использует genkernel
image=/boot/kernel-genkernel-x86-2.6.12-gentoo-r10
label=gentoo
read-only
root=/dev/ram0
append="init=/linuxrc ramdisk=8192 real_root=/dev/hda3 udev"
initrd=/boot/initramfs-genkernel-2.6.12-gentoo-r10

# следующие две строки нужны только для двойной загрузки с Windows
# в этом примере Windows находится на /dev/hda6
other=/dev/hda6
label=windows
```

Примечание: Параметр `udev`, указанный в конце строки `kernel`, необходим для обхода ошибки в некоторых версиях `genkernel`, если вы вообще используете `udev` (по умолчанию — используется).

Примечание: Если вы разбили жесткий диск по-другому, или у вас другое ядро, внесите необходимые изменения.

Если нужно передать ядру дополнительные параметры, добавьте в соответствующий раздел файла выражение `append`. Например, добавим параметр `video` для включения кадрового буфера:

Листинг 12: Использование `append` для добавления параметров ядра

```
image=/boot/kernel-2.6.12-gentoo-r10
label=gentoo
read-only
root=/dev/hda3
append="video=vesafb:mtrr,ywrap,1024x768-32@85"
```

Если вы используете ядро 2.6.7 или выше, а объем жесткого диска ограничили перемычками из-за того, что BIOS не в состоянии работать с дисками большого размера, вам потребуется добавить `hdx=stroke`.

Тем, кто использует `genkernel`, нужно помнить, что их ядро использует такие же загрузочные параметры, как на установочном компакт-диске. Например, если у вас есть устройства SCSI, следует передать ядру параметр `doscsi`.

Теперь сохраните файл и выйдите из редактора. Для окончания установки нужно запустить `/sbin/lilo`, чтобы LILO смог отразить настройки, сделанные в `/etc/lilo.conf`, в вашей системе (т.е. записался на диск). Имейте в виду, что при каждой

установке нового ядра или изменении меню вам потребуется выполнять `/sbin/lilo` заново.

Листинг 13: Завершение установки LILo
`# /sbin/lilo`

Примечание: При переустановке ядра вам больше не придется копировать файлы. Просто запустите `make install` после компиляции ядра; копирование файлов и изменение конфигурации LILo выполнится автоматически.

Переходите к перезагрузке системы.

10.d. Перезагрузка системы

Выйдите из изолированной среды и размонтируйте все смонтированные разделы. Затем введите ту самую волшебную команду, которую вы так жаждали: `reboot`.

Листинг 14: Размонтирование всех разделов и перезагрузка
`# exit`
`cdimage ~# cd`
`cdimage ~# umount /mnt/gentoo/boot /mnt/gentoo/dev /mnt/gentoo/proc /mnt/gentoo`
`cdimage ~# reboot`

Естественно, не забудьте вынуть загрузочный компакт-диск, иначе он загрузится сам вместо вашей новой системы Gentoo.

Загрузив вновь установленную систему, переходите к завершению установки Gentoo.

11. Завершение установки Gentoo

11.a. Управление учетными записями

Добавление учетной записи для повседневной работы

Работа в учетной записи `root` (администратора) в системе Unix/Linux опасна, и ее следует всячески избегать. Поэтому настоятельно рекомендуется добавить учетную запись пользователя для повседневной работы.

Членством пользователя в группах определяется, какие действия он сможет выполнять. В следующей таблице перечислено несколько важных групп, в которые вы, возможно, захотите включать пользователей.

Группа	Описание
audio	возможность доступа к аудиоустройствам
cdrom	возможность прямого доступа к оптическим накопителям
floppy	возможность прямого доступа к гибким дискам
games	возможность играть в игры
portage	возможность использования emerge --pretend с правами пользователя
usb	возможность доступа к устройствам USB
plugdev	возможность монтирования и использования подключаемых устройств типа камер и USB-брелков
video	возможность доступа к средствам видеозахвата и выполнения аппаратного ускорения видео
wheel	возможность использования команды su

Например, для создания учетной записи пользователя по имени john, входящего в группы wheel, users и audio, сначала войдите в систему как root (только root может создавать учетные записи пользователей), а затем запустите useradd:

Листинг 1: Создание учетной записи на каждый день

Login: root

Password: (ваш пароль root)

```
# useradd -m -G users,wheel,audio -s /bin/bash john
```

```
# passwd john
```

Password: (введите пароль для john)

Re-enter password: (повторно введите пароль для подтверждения)

Если пользователю потребуется выполнить задачу от имени root, для временного получения привилегий root можно использовать su -. Другой способ — пользоваться пакетом sudo, который при правильной настройке вполне безопасен.

12. Чем заняться дальше?

12.a. Документация

Примите поздравления! У вас теперь появилась работающая система Gentoo. И что же делать дальше? Какие у вас появились возможности? На что стоит взглянуть прежде всего? Gentoo дает своим пользователям богатый выбор возможностей, а следовательно — и множество документированных (или не очень) свойств.

Вам обязательно нужно прочитать следующую часть настольной книги Gentoo, работа в Gentoo, в которой рассказывается, как поддерживать программное обеспечение в актуальном состоянии, как до устанавливать программы, что такое «USE-флаги», как происходит инициализация в Gentoo и т.д.

Если вас интересует оптимизация системы с точки зрения пользователя, или вы

хотите узнать, как настроить полноценный «рабочий стол», обратитесь к более подробной документации по графической среде Gentoo. Кроме того, вы, возможно захотите прочитать наше руководство по локализации (англ.), что бы чувствовать себя более уютно.

Примечание: В настоящее время для русификации рекомендуется пользоваться альтернативными руководствами по локализации Gentoo. — прим. пер.

Также имеется настольная книга по безопасности в Gentoo (англ.), которую определенно стоит прочитать.

Полный список существующих материалов имеется на странице документации.

12.b. Gentoo в интернете

Естественно, мы всегда рады видеть вас на форумах Gentoo (англ.), как и на любом из множества IRC-каналов Gentoo (англ.).

Кроме того, мы можем предложить вашему вниманию несколько списков рассылки, открытых для всех наших пользователей. Сведения о порядке подписки находятся на той же странице.

На этом мы замолкаем, чтобы позволить вам в полной мере насладиться результатом установки :)

В. Работа с Gentoo

1. Введение в Portage

1.a. Добро пожаловать в Portage

Система Portage — вероятно, самое известное нововведение Gentoo в управлении программным обеспечением. Благодаря высокой гибкости и чрезвычайно богатым возможностям, она зачастую считается лучшим средством управления программным обеспечением из существующих в Linux.

Portage полностью написана на Python и Bash, и в результате полностью прозрачна для пользователей, поскольку оба — языки сценариев.

Большинство пользователей взаимодействует с Portage с помощью команды `emerge`. Эта глава не призвана заменить страницу справки `emerge`. Для просмотра всех возможных параметров команды `emerge`, обращайтесь к странице справки:

Листинг 1: Чтение страницы справки `emerge`
`$ man emerge`

1.b. Дерево портежей

Сборочные файлы `ebuild`

Говоря о пакетах, мы часто имеем в виду программы, доступные пользователям Gentoo через дерево портежей. Дерево портежей — это набор сборочных файлов `ebuild`, содержащих всю информацию, необходимую Portage для управления программным обеспечением (установки, поиска, извлечения и т.п.) По умолчанию сборочные файлы находятся в `/usr/portage`.

Когда Portage по вашему поручению выполняет любые действия над пакетами программ, эти действия опираются на сборочные файлы, имеющиеся в системе. Поэтому необходимо регулярно обновлять сборочные файлы, чтобы Portage знала о новых программах, обновлениях, связанных с безопасностью и т.д.

Обновление дерева портежей

Дерево портежей обычно обновляется с помощью **`rsync`** (англ.), средства быстрой разностной передачи файлов. Обновление выполнить довольно просто, так как запуск `rsync` обеспечивается командой `emerge` :

Листинг 2: Обновление дерева портежей

emerge --sync

Если rsync выполнить невозможно из-за ограничений межсетевого экрана, дерево портежей все-таки можно обновить из ежедневных «снимков», создаваемых нами. Для автоматического извлечения и установки в системе новейшего снимка служит утилита **emerge-webrsync**:

Листинг 3: Запуск emerge-webrsync
emerge-webrsync

1.с. Обслуживание программного обеспечения

Поиск программ

Для поиска программ в дереве портежей по названию можно использовать встроенные возможности команды **emerge**. По умолчанию команда **emerge --search** выдает названия пакетов, соответствующих (как полностью, так и частично) заданному условию поиска.

Например, чтобы найти все пакеты, содержащие «pdf» в названии:

Листинг 4: Поиск пакетов с pdf в названии
\$ emerge --search pdf

Для поиска пакетов еще и по тексту описания можно использовать параметр **--searchdesc** (или **-S**):

Листинг 5: Поиск пакетов, связанных с pdf
\$ emerge --searchdesc pdf

Посмотрев на сообщения команды, вы отметите, что вам дается множество информации. Поля четко обозначены, поэтому мы не будем вдаваться в подробности их значения:

Листинг 6: Пример вывода emerge --search

```
* net-print/cups-pdf
  Latest version available: 1.5.2
  Latest version installed: [ Not Installed ]
  Size of downloaded files: 15 kB
  Homepage:  http://cip.physik.uni-wuerzburg.de/~vrbehr/cups-pdf/
  Description: Provides a virtual printer for CUPS to produce PDF files.
  License:   GPL-2

(
* net-print/cups-pdf
  Последняя доступная версия: 1.5.2
  Последняя установленная версия: [ не установлен ]
  Размер загружаемых файлов: 15 kB
  Веб-страница: http://cip.physik.uni-wuerzburg.de/~vrbehr/cups-pdf/
```

Описание: Снабжает CUPS виртуальным принтером для печати PDF-файлов.
Лицензия: GPL-2)

Примечание (JohnBat26): Для более быстрого поиска программ используйте утилиту: **eix** (установка **emerge eix**). Она работает быстрее благодаря индексированию. Не забывайте запускать **update-eix** после **emerge —sync ;**)

Установка программ

После того, как вы нашли нужное программное обеспечение, его можно легко установить с помощью команды **emerge**. Вот пример установки пакета **gnumeric**:

Листинг 7: Установка gnumeric
emerge gnumeric

Так как множество приложений зависит друг от друга, любая попытка установить какой-либо пакет программ может повлечь за собой также установку дополнительных пакетов. Не беспокойтесь, Portage справится и с этим. Если вы захотите выяснить, что именно Portage собирается установить вместе с нужным вам пакетом, добавьте параметр **--pretend**. Например:

Листинг 8: Проверка зависимостей пакета gnumeric
emerge --pretend gnumeric

После команды на установку пакета, Portage загружает из интернета необходимый исходный код (при необходимости), и по умолчанию сохраняет его в каталоге **/usr/portage/distfiles**. После этого пакет распаковывается, компилируется и устанавливается. Если вы хотите, чтобы Portage только загрузила исходный код без его установки, добавьте к команде **emerge** параметр **--fetchonly**:

Листинг 9: Загрузка исходного кода пакета gnumeric
emerge --fetchonly gnumeric

Обнаружение документации к пакету

Многие пакеты содержат собственную документацию. Иногда USE-флаг **doc** определяет, следует ли устанавливать документацию к пакету. Проверить наличие USE-флага **doc** можно командой **emerge -vp <название пакета>**.

Листинг 10: Проверка наличия USE-флага doc
(alsa-lib - это всего лишь пример)
emerge -vp alsa-lib
[ebuild N] media-libs/alsa-lib-1.0.9_rc3 +doc -jack 674 kB

USE-флаг **doc** можно включить или отключить как глобально в файле **/etc/make.conf**, так и для отдельных пакетов в файле **/etc/portage/package.use**. Также можно, создав каталог с названием **/etc/portage/package.use**, указать флаг в

файле внутри этого каталога. В главе USE-флаги этот вопрос описывается более подробно.

Документация от вновь установленного пакета обычно находится в подкаталоге каталога `/usr/share/doc`, соответствующем названию пакета. Кроме того, можно вывести список всех установленных файлов утилитой `equery`, которая входит в пакет `gentoolkit` (англ.) — `app-portage/gentoolkit`.

Листинг 11: Обнаружение документации пакета

```
# ls -l /usr/share/doc/alsa-lib-1.0.9_rc3
total 28
-rw-r--r-- 1 root root 669 May 17 21:54 ChangeLog.gz
-rw-r--r-- 1 root root 9373 May 17 21:54 COPYING.gz
drwxr-xr-x 2 root root 8560 May 17 21:54 html
-rw-r--r-- 1 root root 196 May 17 21:54 TODO.gz
```

(или используйте для поиска интересных файлов команду `equery` :)

```
# equery files alsa-lib | less
media-libs/alsa-lib-1.0.9_rc3
* Contents of media-libs/alsa-lib-1.0.9_rc3:
/usr
/usr/bin
/usr/bin/alsalisp
(выдача обрезана)
```

Удаление пакета

Когда вы захотите удалить пакет из системы, используйте команду `emerge --unmerge`. Это приведет к удалению из системы всех файлов, установленных пакетом, кроме конфигурационных файлов приложения, изменявшихся после установки. Сохранение конфигурационных файлов позволяет вернуться к работе с пакетом, если вы когда-нибудь решите снова его установить.

Внимание: Portage не проверяет, зависят ли другие пакеты от удаляемого! Однако вы получите предупреждение, если удаление пакета приведет к неработоспособности системы.

Листинг 12: Удаление пакета `gnnumeric` из системы

```
# emerge --unmerge gnumeric
```

После удаления пакета из системы, пакеты, установленные автоматически, потому что от них зависел удаленный пакет, остаются. Чтобы Portage выявила все когда-то нужные пакеты, которые теперь можно удалить, используйте команду **`emerge --depclean`**. Мы вернемся к этому ниже.

Обновление системы

Чтобы система сохранялась в отличной форме (не говоря уже об установке свежайших обновлений, связанных с безопасностью), ее нужно регулярно обновлять. Так как Portage просматривает сборочные файлы только в локальном

дереве портежей, сперва потребуется обновить его. Обновив дерево портежей, вы сможете обновить систему командой `emerge --update world`. В следующем примере мы также пользуемся параметром `--ask`, который поручает Portage вывести список пакетов, которые она собирается обновить, и спросить вас, можно ли продолжать:

Листинг 13: Обновление системы
emerge --update --ask world

Portage будет искать более новые версии установленных приложений. Однако проверяется только версии приложений, явно установленных вами, а не тех, от которых они зависят. Если вы хотите обновить каждый пакет в системе, добавьте аргумент `--deep`:

Листинг 14: Обновление всей системы
emerge --update --deep world

Поскольку обновления, относящиеся к безопасности, случаются и в пакетах, явным образом не устанавливались (но были «подтянуты» из-за того, что он них зависят другие программы), рекомендуется изредка запускать эту команду.

Если вы меняли какие-либо из USE-флагов, возможно, потом вы также захотите добавить параметр `--newuse`. Тогда Portage проверит, требует ли изменение установки новых пакетов или перекомпиляции существующих:

Листинг 15: Выполнение полного обновления
emerge --update --deep --newuse world

Метапакеты

У некоторых пакетов в дереве портежей нет содержимого как такового, и они используются для установки набора других пакетов. Например, пакет **kde** полностью устанавливает среду KDE в вашей системе, привлекая различные KDE-пакетов в качестве зависимостей.

Если вы когда-либо захотите удалить из системы такой пакет, запуск **emerge--unmerge** не возымеет должного эффекта, так как пакеты, от которых он зависит, останутся в системе.

В Portage существует возможность удаления остаточных зависимостей, но так как зависимости программ меняются со временем, доступность программного обеспечения, прежде всего требуется полностью обновить всю систему, включая реализацию изменений, произведенных путем модификации USE-флагов. После этого можно запустить **emerge --depclean**, чтобы удалить остаточные зависимости. Когда это сделано, вам потребуется пересобрать приложения, ранее динамически связанные с удаленными пакетами, в которых они теперь не нуждаются.

Со всем этим управляются следующие три команды:

Листинг 16: Удаление ненужных зависимостей

```
# emerge --update --deep --newuse world
# emerge --depclean
# revdep-rebuild
```

revdep-rebuild входит в пакет **gentoolkit**; не забудьте сначала его установить:

Листинг 17: Установка пакета gentoolkit

```
# emerge gentoolkit
```

1.d. Когда Portage жалуется...

Слоты, виртуалы, ветви, архитектуры и профили

Как уже сказано, **Portage** — чрезвычайно мощная система, поддерживающая множество возможностей, не хватающих другим системам управления программами. Чтобы это стало понятно, разберем несколько аспектов Portage, не вникая в подробности.

С помощью Portage разные версии отдельного пакета могут сосуществовать в одной системе. В то время, как другие системы управления стремятся называть пакеты в соответствии с версией (например **freetype** и **freetype2**), **в Portage используется технология слотов (SLOT), или областей**. Пакет присваивает определенный слот своей версии. Пакеты с разными слотами способны сосуществовать в одной системе. Например, у пакета **freetype** есть ebuild как со **SLOT="1"**, так и со **SLOT="2"**.

Существуют также пакеты, выполняющие одни и те же функции, но отличающиеся в реализации. Например **metalogd**, **syslogd** и **syslog-ng** являются системными службами журналирования. Приложения, использующие «системный журнал», не могут зависеть от одной конкретной программы, например от **metalogd**, так как остальные программы ничем не хуже. В Portage предусмотрены виртуальные пакеты: каждая служба журналирования предоставляет **virtual/syslog**, и в результате в приложениях можно указывать зависимость от **virtual/syslog**.

Программное обеспечение может располагаться в различных ветвях дерева портежей. По умолчанию в системе разрешено только использование стабильных пакетов. Большинство новых программ при поступлении включаются в тестовую ветвь, что указывает на необходимость дополнительного тестирования перед тем, как включить их в стабильные. Хотя в дереве портежей и видны сборочные файлы для таких программ, Portage не станет обновлять их до тех пор, пока они не будут помещены в стабильную ветвь.

Некоторые программы имеются не для всех архитектур. Либо они не работают в определенных архитектурах, либо требуют дополнительного тестирования, или у разработчика нет возможности проверить, работает ли пакет в различных

архитектурах.

Каждая установка Gentoo придерживается определенного профиля, который содержит, помимо прочего, список пакетов, необходимых для работоспособности системы.

Блокировка пакетов

Листинг 18: Предупреждение о заблокированных пакетах (с --pretend)
[blocks B] mail-mta/ssmtp (is blocking mail-mta/postfix-2.2.2-r1)

Листинг 19: Предупреждение о заблокированных пакетах (без --pretend)
!!! Error: the mail-mta/postfix package conflicts with another package.
!!! both can't be installed on the same system together.
!!! Please use 'emerge --pretend' to determine blockers.

(!!! Ошибка: пакет mail-mta/postfix конфликтует с другим пакетом.
!!! оба не могут находиться в системе одновременно. Пожалуйста,
!!! запустите 'emerge --pretend' для выявления блокирующих пакетов.)

В файлах ebuild есть специальные поля, сообщающие Portage о зависимостях. Возможны два вида зависимости: зависимость сборки, объявленная в **DEPEND**, и зависимость выполнения, объявленная в **RDEPEND**. Когда одна из этих зависимостей явно указывает на несовместимость пакета или виртуального пакета, это вызывает блокировку.

Для разблокировки можно отказаться от установки пакета или предварительно удалить конфликтующего пакета. В данном примере можно отказаться от установки **postfix** или сначала удалить **ssmtp**.

Также возможно, что два пакета, подлежащие установке, блокируют друг друга. В этом редчайшем случае следует определить, зачем вам устанавливать оба пакета. В большинстве случаев можно обойтись одним. Если это окажется не так, то, пожалуйста, заведите отчет об ошибке в системе распределения запросов Gentoo.

Маскировка пакетов

Листинг 20: Предупреждение о замаскированных пакетах
!!! all ebuilds that could satisfy "boot splash" have been masked.

(!!! все сборки, удовлетворяющие "boot splash", замаскированы.)

Листинг 21: Предупреждение о замаскированных пакетах с указанием причины
!!! possible candidates are:

- gnome-base/gnome-2.8.0_pre1 (masked by: ~x86 keyword)
- lm-sensors/lm-sensors-2.8.7 (masked by: -sparc keyword)
- sys-libs/glibc-2.3.4.20040808 (masked by: -* keyword)
- dev-util/cvsd-1.0.2 (masked by: missing keyword)
- media-video/ati-gatos-4.3.0 (masked by: package.mask)

- sys-libs/glibc-2.3.2-r11 (masked by: profile)

(!!! возможные кандидаты:

- gnome-base/gnome-2.8.0_pre1 (маскировка: ключ ~x86)
- lm-sensors/lm-sensors-2.8.7 (маскировка: ключ -sparc)
- sys-libs/glibc-2.3.4.20040808 (маскировка: ключ -*)
- dev-util/cvsd-1.0.2 (маскировка: ключ отсутствует)
- media-video/ati-gatos-4.3.0 (маскировка: package.mask)
- sys-libs/glibc-2.3.2-r11 (маскировка: profile)

Когда вы собираетесь установить пакет, не предназначенный для вашей системы, выдается ошибка маскировки. Нужно попытаться установить другую программу, существующую для вашей системы, или дождаться, пока пакет станет доступным. Всегда есть причина, по которой пакет замаскирован:

- ключ **~arch**: пакет недостаточно проверен для помещения в стабильную ветвь. Подождите несколько дней или недель и попробуйте установить его еще раз;
- ключ **-arch** или ключ **-***: пакет не работоспособен в вашей архитектуре. Если вы полагаете, что он работает, сообщите об этом в **bugzilla**;
- ключ **отсутствует**: пакет еще не тестировался в вашей архитектуре. Попросите группу портирования в архитектуру проверить пакет, или протестируйте его за них и сообщите о своих изысканиях в bugzilla;
- **package.mask**: обнаружено повреждение пакета, нестабильность или что-то худшее, и пакет заблокирован специально;
- **profile**: пакет считается не предназначенным для вашего профиля. В случае установки приложение может вызвать сбой системы или просто несовместимо с используемым профилем.

Отсутствие нужных пакетов

Листинг 22: Предупреждение об отсутствии пакета
 emerge: there are no ebuids to satisfy ">=sys-devel/gcc-3.4.2-r4".

```
!!! Problem with ebuild sys-devel/gcc-3.4.2-r2
!!! Possibly a DEPEND/*DEPEND problem.
```

(emerge: нет сборок, удовлетворяющих ">=sys-devel/gcc-3.4.2-r4".

```
!!! Проблема с ebuild sys-devel/gcc-3.4.2-r2
!!! Возможно, ошибка в DEPEND/*DEPEND. )
```

Приложение, которое вы пытаетесь установить, зависит от другого пакета, недоступного вашей системе. Пожалуйста, проверьте, есть ли такой запрос в **bugzilla**, а если нет, сообщите об ошибке. Если вы не смешиваете ветви, такого не должно происходить, и это — явная ошибка.

Неоднозначность названия пакета

Листинг 23: Предупреждение о повторяющихся именах ebuild
!!! The short ebuild name "aterm" is ambiguous. Please specify
!!! one of the following fully-qualified ebuild names instead:

```
dev-libs/aterm
x11-terms/aterm
```

(!!! Короткое название ebuild "aterm" неоднозначно. Пожалуйста,
!!! вместо него укажите одно из полных названий ebuild:

```
dev-libs/aterm
x11-terms/aterm )
```

Название приложения, которое вы собираетесь установить, соответствует более чем одному пакету. Требуется также указать название категории. Portage предложит вам возможные варианты.

Циклические зависимости

Листинг 24: Предупреждение Portage о циклических зависимостях
!!! Error: circular dependencies:

```
ebuild / net-print/cups-1.1.15-r2 depends on ebuild /
app-text/ghostscript-7.05.3-r1
ebuild / app-text/ghostscript-7.05.3-r1 depends on ebuild /
net-print/cups-1.1.15-r2
```

(!!! Ошибка: циклические зависимости:

```
ebuild / net-print/cups-1.1.15-r2 зависит от ebuild /
app-text/ghostscript-7.05.3-r1
ebuild / app-text/ghostscript-7.05.3-r1 зависит от ebuild /
net-print/cups-1.1.15-r2 )
```

Два или более пакета, которые вы хотите установить, взаимно зависимы, и в результате их установка невозможна. Скорее всего, это ошибка в дереве портежей. Пожалуйста, выждав время, обновите дерево портежей, и попробуйте снова. Вы можете также проверить, есть ли эта ошибка в bugzilla, и если нет, сообщить о ней.

Ошибка извлечения

Листинг 25: Предупреждение Portage об ошибке извлечения
!!! Fetch failed for sys-libs/ncurses-5.4-r5, continuing...

(...)

!!! Some fetch errors were encountered. Please see above for details.

(!!! Ошибка при извлечении sys-libs/ncurses-5.4-r5, продолжение...

(...)

!!! При извлечении произошли ошибки. Подробности выше.)

Portage не смогла загрузить исходный код данного приложения и попытается продолжить установку других приложений (если запланирована). Эта ошибка может произойти из-за неправильно синхронизированного зеркала, или из-за того, что ebuild указывает на неверное место. Сервер, где находятся исходные коды, также может почему-либо не работать.

Повторите действие через час, чтобы посмотреть, повторится ли эта ошибка.

Защита системного профиля

Листинг 26: Предупреждение Portage о пакете, защищенном профилем
 !!! Trying to unmerge package(s) in system profile. 'sys-apps/portage'
 !!! This could be damaging to your system.

(!!! Попытка удаления пакетов из системного профиля. 'sys-apps/portage'
 !!! Это может повредить вашей системе.)

Вы попросили удалить пакет, входящий в состав базовых пакетов вашей системы. Он отмечен в вашем профиле как обязательный, и его не следует удалять из системы.

2. USE-флаги

2.а. Что такое USE-флаги?

Смысл USE-флагов

Устанавливая Gentoo (или любой другой дистрибутив, или даже операционную систему вообще), вы выбираете те или иные возможности в зависимости от среды, с которой работаете. Установка сервера отличается от установки рабочей станции, а установка игровой станции — от платформы 3D-рендеринга.

Это касается не только того, какие пакеты устанавливать, но и какие функции определенных пакетов должны поддерживаться. Если вам не нужен OpenGL, то зачем вам его ставить и встраивать поддержку OpenGL в большинство программ? Если вы не собираетесь использовать KDE, зачем собирать пакеты с его поддержкой, если они работают и без этого?

Чтобы помочь пользователям в выборе того, что устанавливать/активировать, а что — нет, мы захотели дать им простой способ описания рабочей среды. Это позволяет пользователю решить, что же ему на самом деле надо, и облегчить работу с Portage, нашей системой управления пакетами.

Определение USE-флагов

Рассмотрим USE-флаги. USE-флаг — это ключевое слово, включающее сведения о поддержке и зависимостях определенного понятия или функции. При определении какого-либо USE-флага, Portage узнает, что вам нужна поддержка соответствующей функции. Конечно, это также влияет на сведения о зависимостях пакета.

Давайте рассмотрим конкретный пример — ключевое слово **kde**. Если в вашей переменной USE нет этого слова, то все пакеты, где поддержка KDE является необязательной, собираются без нее. Все пакеты, где зависимость от KDE является необязательной, устанавливаются без установки библиотек KDE (по зависимости). Если же вы определите ключевое слово **kde**, то эти пакеты будут собираться с поддержкой KDE, а KDE будет установлен в качестве необходимого.

Правильно определяя ключевые слова, вы создаете систему, подогнанную специально для ваших нужд.

Какие USE-флаги существуют?

Есть два типа USE-флагов: **глобальные** и **локальные**.

Глобальный USE-флаг используется несколькими пакетами и является системным. Это то, что большинство видит в качестве USE-флагов.

Локальный USE-флаг используется единственным пакетом для настройки определенных параметров самого пакета.

Список доступных глобальных USE-флагов можно найти в сети или локально в **/usr/portage/profiles/use.desc**.

Список локальных USE-флагов находится в вашей системе в **/usr/portage/profiles/use.local.desc**.

Примечание (JohnBat26): Для удобной установки и чтения описаний USE флагов используйте программу: **Use Flag EDitor** или **ufed (emerge ufed)**

2.b. Использование USE-флагов

Объявление постоянных USE-флагов

В надежде, что вы убедились в важности USE-флагов, теперь мы расскажем, как их объявлять.

Как сказано ранее, все USE-флаги объявляются в переменной USE. Чтобы упростить пользователям поиск и выбор флагов, мы предлагаем значение USE по умолчанию, которое представляют собой список USE-флагов, как нам кажется, наиболее часто используемых пользователями Gentoo. Это значение приведено в файле **make.defaults** вашего профиля.

Профиль, на который ориентируется ваша система, указывается символьной ссылкой **/etc/make.profile**. Каждый профиль основывается на предыдущем, более

крупном, а итоговый складывается из всех профилей. Верхним является базовый профиль (/usr/portage/profiles/base).

Давайте взглянем на значение по умолчанию для профиля 2004.3:

Листинг 1: Итоговая переменная USE make.defaults для профиля 2004.3
(в этом примере объединяются значения из base, default-linux, default-linux/x86 и default-linux/x86/2004.3)

```
USE="x86 oss apm arts avi berkdb bitmap-fonts crypt cups encode fortran f77
foomaticdb gdbm gif gpm gtk imlib jpeg kde gnome libg++ libwww mad
mikmod motif mpeg ncurses nls oggvorbis opengl pam pdflib png python qt
quicktime readline sdl spell ssl svga tcpd truetype X xml2 xmms xv zlib"
```

Как видите, эта переменная уже содержит достаточно много ключевых слов. Не меняйте файл make.defaults для настройки переменной USE под свои нужды: изменения в этом файле аннулируются при обновлении Portage!

Для изменения значения по умолчанию, нужно добавлять или удалять ключевые слова из переменной USE. Это делается глобально, определением переменной USE в /etc/make.conf. В эту переменную можно добавить нужные вам USE-флаги, или удалить ненужные. Для удаления флага, его надо указывать со знаком минус в виде приставки («-»).

Например, чтобы убрать поддержку **KDE** и **QT**, но добавить поддержку **Idap**, можно определить в /etc/make.conf переменную USE следующего вида:

Листинг 2: Пример значения USE в /etc/make.conf

```
USE="-kde -qt Idap"
```

Объявление USE-флагов для отдельных пакетов

Иногда нужно определить некоторые USE-флаги только для одного или нескольких пакетов, не трогая системных настроек. Для этого необходимо создать каталог /etc/portage (если его еще нет) и отредактировать файл /etc/portage/package.use.

Например, вам не нужна глобальная поддержка berkdb, но она необходима в mysql:

Листинг 3: Пример /etc/portage/package.use

```
dev-db/mysql berkdb
```

Естественно, можно в явном виде отключить USE-флаги для определенного пакета. Например, если вам не нужна поддержка java в PHP:

Листинг 4: Второй пример /etc/portage/package.use

```
dev-php/php -java
```

Объявление временных USE-флагов

Иногда необходимо установить какой-то USE-флаг только на один раз. Вместо того, чтобы дважды редактировать `/etc/make.conf` (сначала добавить изменения USE, а потом удалить), можно просто объявить USE как переменную среды. Помните, что при переустановке или обновлении приложения (явном или в составе обновления системы) ваши изменения будут утеряны!

Например, уберем **java** из значения USE на время установки mozilla.

Листинг 5: Использование USE в виде переменной среды
USE="-java" emerge mozilla

Наследование

Конечно же, существует определенная последовательность формирования значения USE. Вы же не хотите объявить USE="-java" только для того, чтобы узнать, что java все еще включена из-за значения с более высоким приоритетом. Последовательность установки значения USE в порядке приоритета (от меньшего к большему) такова:

- значение USE по умолчанию, объявленное в файлах `make.defaults` в составе вашего профиля;
- значение, определенное пользователем в `/etc/make.conf` ;
- значение, указанное пользователем в `/etc/portage/package.use` ;
- значение, определенное пользователями в переменной среды.

Чтобы узнать, какие же настройки USE в конечном счете видит Portage, запустите `emerge --info`. Эта команда выводит значения всех переменных (включая USE), используемые Portage.

Листинг 6: Запуск `emerge --info`
emerge --info

Адаптация всей системы под новые USE-флаги

Если вы изменили свои USE-флаги и хотите обновить всю систему в соответствии с новым значением USE, запустите `emerge` с параметром **--newuse**:

Листинг 7: Пересборка всей системы
emerge --update -deep --newuse world

Теперь запустите функцию Portage **depclean**, чтобы удалить условные зависимости, присутствующие в «старой» системе, но больше не нужные при новом составе USE-флагов.

Предупреждение: Запуск **emerge --depclean** является опасной операцией, которую следует использовать с осторожностью. Дважды проверьте список «ненужных» пакетов и убедитесь, что не удалятся нужные пакеты. В следующем примере мы добавляем ключ `-p`, чтобы `depclean` только перечислил пакеты, не

удаляя их.

Листинг 8: Удаление ненужных пакетов
`# emerge -p --depclean`

Когда **depclean** закончит свою работу, запустите **revdep-rebuild**, чтобы пересобрать программы, динамически связанные с библиотеками, входящими в потенциально удаленные пакеты. **revdep-rebuild** входит в пакет **gentoolkit**, так что не забудьте сначала установить его.

Листинг 9: Запуск **revdep-rebuild**
`# revdep-rebuild`

После выполнения всех этих действий, ваша система будет полностью использовать новые значения USE-флагов.

2.с. USE-флаги отдельных пакетов

Просмотр доступных USE-флагов

Возьмем, к примеру, **mozilla** — какие USE-флаги она может использовать? Чтобы это выяснить, запустим **emerge** с параметрами **--pretend** и **--verbose**:

Листинг 10: Просмотр используемых USE флагов
`# emerge --pretend --verbose mozilla`
 These are the packages that I would merge, in order:

```
Calculating dependencies ...done!
[ebuild R ] www-client/mozilla-1.7.12-r2 USE="crypt gnome java mozsvg ssl
truectype xprint -debug -ipv6 -ldap -mozcalendar -mozdevelop -moznocompose
-moznoirc -moznomail -moznoxft -postgres -xinerama" 0 kB
```

emerge — не единственное средство для решения этой задачи. Существует программа, специально предназначенная для вывода информации о пакетах. Она называется **equery** и находится в пакете **gentoolkit**. Для начала установим этот пакет:

Листинг 11: Установка **gentoolkit**
`# emerge gentoolkit`

Теперь для просмотра USE-флагов какого-нибудь пакета запустим **equery** с аргументом **uses**. Пусть это будет пакет **gnunumeric**:

Листинг 12: Запуск **equery** для просмотра доступных USE-флагов
`# equery uses =gnunumeric-1.6.3 -a`

```
[ Searching for packages matching =gnumeric-1.6.3... ]
[ Colour Code : set unset ]
[ Legend      : Left column (U) - USE flags from make.conf ]
[              : Right column (I) - USE flags packages was installed with ]
[ Found these USE variables for app-office/gnumeric-1.6.3 ]
U I
-- debug      : Tells configure and the makefiles to build for debugging.
                Effects vary across packages, but generally it will at
                least add -g to CFLAGS. Remember to set FEATURES=nostrip too
-- gnome      : Adds GNOME support
+ + python    : Adds support/bindings for the Python language
-- static     : !!do not set this during bootstrap!! Causes binaries to be
                statically linked instead of dynamically
```

3. Возможности Portage

3.a. Возможности Portage

В Portage есть несколько дополнительных возможностей (features), которые значительно улучшат ваше впечатление от Gentoo. Многие из этих возможностей полагаются на определенные программы, повышающие производительность, надежность, безопасность и т.п.

Для включения и выключения определенных возможностей Portage нужно редактировать в файле `/etc/make.conf` переменную **FEATURES**, в которой перечислены ключевые слова, разделенные пробелами, обозначающие различные возможности. Иногда для использования соответствующих возможностей потребуется установка дополнительных утилит.

Здесь перечислены не все возможности, поддерживаемые Portage. Полный перечень представлен на странице справки `make.conf`:

Листинг 1: Вызов страницы справки `make.conf`
\$ `man make.conf`

Чтобы узнать, какие возможности включены по умолчанию, запустите `emerge --info` и найдите переменную **FEATURES** (или отфильтруйте ее с помощью `grep`):

Листинг 2: Выявление уже включенных возможностей
\$ `emerge --info | grep FEATURES`

3.b. Распределенная компиляция

Использование `distcc`

distcc — программа, распределяющая компиляцию по нескольким, не

обязательно одинаковым, машинам в сети. Клиент **distcc** посылает всю необходимую информацию на доступные серверы **distcc** (на которых выполняется **distccd**), чтобы они могли компилировать для клиента части исходного кода. Чистый выигрыш — более быстрая компиляция.

Подробная информация о **distcc** (и как заставить его заработать в Gentoo) находится в нашем описании **distcc** в Gentoo.

Установка **distcc**

Distcc поставляется с графическим монитором (средством контроля), позволяющим отслеживать задачи, которые ваш компьютер отправляет на компиляцию. Если вы используете **Gnome**, тогда добавьте «**gnome**» к переменной **USE**. А если вы не пользуетесь **Gnome**, но при этом хотите пользоваться монитором, добавьте «**gtk**» к переменной **USE**.

Листинг 3: Установка **distcc**
emerge distcc

Подключение поддержки **Portage**

Добавьте **distcc** к переменной **FEATURES** в файле **/etc/make.conf**. Затем отредактируйте переменную **MAKEOPTS**, как вам нравится. Известная рекомендация — указывать директиву «**-jX**», где **X** — число центральных процессоров, на которых работает **distccd** (включая текущий компьютер) плюс один; у вас могут получиться лучшие результаты и с другими значениями.

Теперь запустите **distcc-config** и введите список доступных серверов **distcc**. Для простоты примера, предположим, что доступные серверы **DistCC** — **192.168.1.102** (текущий компьютер), **192.168.1.103** и **192.168.1.104** (два «удаленных» компьютера):

Листинг 4: Настройка **distcc** для использования трех доступных серверов **distcc**
distcc-config --set-hosts "192.168.1.102 192.168.1.103 192.168.1.104"

Не забудьте также запустить демон **distccd**:

Листинг 5: Запуск демонов **distccd**
rc-update add distccd default
/etc/init.d/distccd start

3.с. Кэширование компиляции

О средстве **ccache**

ccache — это быстрый кэш компилятора. Когда вы компилируете программу,

он кэширует промежуточные результаты так, что всякий раз, когда вы перекомпилируете ту же самую программу, время компиляции значительно сокращается. В типичных случаях общее время компиляции может сокращаться в 5—10 раз.

Если вы интересуетесь подробностями `ccache`, пожалуйста, посетите домашнюю страницу `ccache`.

Установка `ccache`

Для установки `ccache`, выполните **`emerge ccache`**:

Листинг 6: Установка `ccache`
`emerge ccache`

Подключение поддержки Portage

Откройте `/etc/make.conf` и добавьте `ccache` к переменной `FEATURES`. Затем добавьте новую переменную по имени `CCACHE_SIZE` (размер кэша), и установите её равной «2G»:

Листинг 7: Редактирование `CCACHE_SIZE` в `/etc/make.conf`
`CCACHE_SIZE="2G"`

Для проверки работоспособности `ccache`, запросите статистику `ccache`. Из-за того, что Portage использует другой домашний каталог `ccache`, вам также потребуется установить переменную `CCACHE_DIR`:

Листинг 8: Просмотр статистики `ccache`
`CCACHE_DIR="/var/tmp/ccache" ccache -s`

Домашний каталог `ccache` по умолчанию — `/var/tmp/ccache`; изменить это назначение можно, определив переменную `CCACHE_DIR` в `/etc/make.conf`.

Однако, при запуске `ccache` используется каталог по умолчанию, `$ {HOME}/.ccache`, вот почему при запросе статистики (Portage) `ccache` требуется определять переменную `CCACHE_DIR`.

Использование `ccache` для компиляции Си не в Portage

Если вы хотите использовать `ccache` для компиляций не в Portage, добавьте `/usr/lib/ccache/bin` в начало вашей переменной `PATH` (перед `/usr/bin`). Это можно сделать, отредактировав `/etc/env.d/00basic`, который является первым файлом среды, где определяется переменная `PATH`:

Листинг 9: Редактирование `/etc/env.d/00basic`
`PATH="/usr/lib/ccache/bin:/opt/bin"`

3.d. Поддержка двоичных пакетов

Создание готовых (заранее собранных) пакетов

Portage поддерживает установку заранее собранных готовых пакетов. Несмотря на то, что в саму Gentoo не входят заранее собранные пакеты (за исключением снимков GRP), Portage можно настроить на полноценное управление готовыми пакетами.

Чтобы создать двоичный пакет, можно использовать **quickpkg**, если пакет уже установлен в вашей системе, или **emerge** с параметрами **--buildpkg** или **--buildpkgonly**.

Если вы хотите, чтобы Portage создавал двоичные пакеты из каждого пакета, который вы будете устанавливать, добавьте **buildpkg** к переменной **FEATURES**.

Расширенная поддержка создания наборов готовых пакетов имеются в **catalyst**. Для получения подробной информации о **catalyst**, пожалуйста, прочитайте справочное руководство по **catalyst** (англ.) и распространенные вопросы о **catalyst** (англ.).

Установка двоичных пакетов

Хотя в Gentoo такого хранилища нет, вы можете создать централизованное хранилище для заранее скомпилированных двоичных пакетов. Чтобы использовать такое хранилище, потребуется указать Portage путь к нему с помощью переменной **PORTAGE_BINHOST**. Например, если двоичные пакеты находятся на **ftp://buildhost/gentoo**:

Листинг 10: Установка **PORTAGE_BINHOST** в **/etc/make.conf**
PORTAGE_BINHOST="ftp://buildhost/gentoo"

При установке двоичных пакетов, указывайте в команде **emerge** параметр **--getbinpkg** вместе с параметром **--usepkg**. Первый указывает **emerge** загрузить двоичный пакет с сервера, определенного раньше, а второй сообщает **emerge**, что до загрузки исходных кодов и их компиляции сначала нужно попытаться установить этот двоичный пакет.

Например, чтобы установить **gnuplot** из двоичных пакетов:

Листинг 11: Установка двоичного пакета **gnuplot**
emerge --usepkg --getbinpkg gnuplot

Подробную информацию о параметрах установки двоичных пакетов можно найти на странице справки **emerge**:

Листинг 12: Чтение справки по **emerge**
\$ man emerge

4. Сценарии инициализации

4.а. Уровни запуска

Процесс загрузки системы

При загрузке вашей системы по экрану пробегают много текста. Если присмотреться, заметно, что этот текст не меняется от загрузки к загрузке. Последовательность всех этих действий называется последовательностью загрузки и в той или иной степени постоянна.

Во-первых, загрузчик размещает в памяти образ ядра, который вы указали в файле его конфигурации. После этого ядро запускается. Когда ядро загружено и запущено, оно инициализирует относящиеся к ядру структуры и задания, и запускает процесс **init**.

Этот процесс удостоверяется, что все файловые системы (определенные в **/etc/fstab**) смонтированы и готовы к использованию. Затем он выполняет несколько сценариев, находящихся в каталоге **/etc/init.d**, которые запускают службы, необходимые для нормального запуска системы.

И, наконец, когда все сценарии выполнены, **init** подключает терминалы (чаще всего просто виртуальные консоли, которые видны при нажатии **ALT+F1**, **ALT+F2** и т.д.), прикрепляя к каждой консоли специальный процесс под названием **agetty**. Этот процесс впоследствии обеспечивает возможность входа в систему с помощью **login**.

Сценарии инициализации

Сейчас процесс **init** запускает сценарии из каталога **/etc/init.d** не просто в случайном порядке. Более того, запускаются не все сценарии из **/etc/init.d**, а только те, которые предписано исполнять. Решение о запуске сценария принимается в результате просмотра каталога **/etc/runlevels**.

Во-первых, **init** запускает все сценарии из **/etc/init.d**, на которые есть символичные ссылки из **/etc/runlevels/boot**. Обычно сценарии запускаются в алфавитном порядке, но в некоторых сценариях имеется информация о зависимостях от других сценариев, указывающая системе на необходимость их предварительного запуска.

Когда все сценарии, указанные в **/etc/runlevels/boot**, будут выполнены, **init** переходит к запуску сценариев, на которые есть символичные ссылки из **/etc/runlevels/default**. И снова запуск происходит в алфавитном порядке, пока в сценарии не встретится информация о зависимостях; тогда порядок изменяется для обеспечения правильного порядка запуска.

Как работает `init`

Конечно, `init` не принимает решений сам по себе. Ему необходим конфигурационный файл, где описаны необходимые действия. Этот файл — `/etc/inittab`.

Если вы запомнили последовательность загрузки, описанную чуть ранее, вы вспомните, что первое действие `init` — это монтирование всех файловых систем. Это определяется в строке `/etc/inittab`, приведенной ниже:

Листинг 1: Строка инициализации системы из `/etc/inittab`
`si::sysinit:/sbin/rc sysinit`

Этой строкой процессу `init` предписывается выполнить `/sbin/rc sysinit` для инициализации системы. Самой инициализацией занимается сценарий `/sbin/rc`, так что можно сказать, что `init` делает не слишком много — он просто делегирует задачу по инициализации системы другому процессу.

Во-вторых, `init` выполняет все сценарии, на которые есть символьные ссылки из `/etc/runlevels/boot`. Это определяется следующей строкой:

Листинг 2: Инициализация системы, продолжение
`rc::bootwait:/sbin/rc boot`

И снова все необходимые действия выполняются сценарием `rc`. Заметьте, что параметр, переданный `rc` (`boot`), совпадает с названием используемого подкаталога в `/etc/runlevels`.

Теперь `init` проверяет свой конфигурационный файл, чтобы определить, какой уровень запуска использовать. Для этого из `/etc/inittab` считывается строка:

Листинг 3: Строка `initdefault`
`id:3:initdefault:`

В приведенном примере (который подходит для подавляющего большинства пользователей Gentoo) номер уровня запуска — 3. Пользуясь этой информацией, `init` проверяет, что нужно выполнить для запуска уровня запуска 3:

Листинг 4: Определение уровней запуска
`I0:0:wait:/sbin/rc shutdown`
`I1:S1:wait:/sbin/rc single`
`I2:2:wait:/sbin/rc nonetwork`
`I3:3:wait:/sbin/rc default`
`I4:4:wait:/sbin/rc default`
`I5:5:wait:/sbin/rc default`
`I6:6:wait:/sbin/rc reboot`

В строке, определяющей уровень 3, для запуска служб снова используется сценарий **rc** (на этот раз с аргументом **default**). Опять-таки, обратите внимание, что аргумент, передаваемый сценарию **rc**, совпадает с названием подкаталога из **/etc/runlevels**.

По окончании работы **rc**, **init** принимает решение о том, какие виртуальные консоли включить и какие команды выполнить в каждой из них:

Листинг 5: Определение виртуальных консолей

```
c1:12345:respawn:/sbin/agetty 38400 tty1 linux
c2:12345:respawn:/sbin/agetty 38400 tty2 linux
c3:12345:respawn:/sbin/agetty 38400 tty3 linux
c4:12345:respawn:/sbin/agetty 38400 tty4 linux
c5:12345:respawn:/sbin/agetty 38400 tty5 linux
c6:12345:respawn:/sbin/agetty 38400 tty6 linux
```

Что такое уровень запуска?

Как вы заметили, **init** применяет нумерацию для определения уровня запуска, который надо использовать. Уровень запуска — это то состояние, в котором запускается ваша система, он содержит набор сценариев (сценариев уровня запуска или сценариев инициализации [initscript]), которые следует выполнять, при входе и выходе из определенного уровня запуска.

В Gentoo определено семь уровней запуска: три служебных и четыре определяемых пользователем. Служебные называются **sysinit**, **shutdown** и **reboot**. Действия, совершаемые ими, в точности соответствуют их названиям: инициализация системы, выключение системы и ее перезагрузка.

Определяемые пользователем уровни — это те, которым соответствуют подкаталоги в **/etc/runlevels**: **boot**, **default**, **nonetwork** и **single**. Уровень **boot** запускает все службы, необходимые системе и используемые всеми остальными уровнями. Остальные уровни отличаются друг от друга запускаемыми службами: **default** используется для повседневной работы, **nonetwork** — для тех случаев, когда не требуется сеть, а **single** — при необходимости восстановления системы.

Работа со сценариями инициализации

Сценарии, запускаемые процессом **rc**, называются сценариями инициализации. Каждый сценарий из **/etc/init.d** может запускаться с аргументами **start**, **stop**, **restart**, **pause**, **zap**, **status**, **ineed**, **iuse**, **needsme**, **usesme** и **broken**.

Для запуска, остановки или перезапуска службы (и всех, зависящих от нее) следует использовать **start**, **stop** и **restart**:

Листинг 6: Запуск postfix
/etc/init.d/postfix start

Примечание: Останавливаются или перезапускаются только те службы, которым необходима данная служба. Остальные зависимые службы (те, которые используют службу, но не нуждаются в ней) эта операция не затрагивает.

Если вы хотите остановить службу, но оставить зависимые от нее работающими, можно использовать аргумент **pause**:

Листинг 7: Остановка postfix без остановки зависимых служб
`# /etc/init.d/postfix pause`

Чтобы узнать текущее состояние службы (запущена, остановлена, приостановлена и т.д.), можно использовать аргумент **status**:

Листинг 8: Информация о состоянии postfix
`# /etc/init.d/postfix status`

Если указано, что служба работает, но вы знаете, что это не так, можно сбросить состояние на **stopped** (остановлена), используя аргумент **zap**:

Листинг 9: Сброс информации о состоянии postfix
`# /etc/init.d/postfix zap`

Для того, чтобы выяснить зависимости службы, можно использовать аргументы **iuse** или **ineed**. С помощью **ineed** вы увидите те службы, которые действительно необходимы для правильного функционирования интересующей вас службы. С другой стороны, **iuse** покажет те службы, которые могут использоваться нашей службой, но не обязательны для ее работы.

Листинг 10: Запрос списка всех необходимых служб, от которых зависит Postfix
`# /etc/init.d/postfix ineed`

Аналогично вы можете узнать, какие службы нуждаются в данной службе (**needsme**) или могут ее использовать (**usesme**):

Листинг 11: Запрос списка всех служб, которым необходим Postfix
`# /etc/init.d/postfix needsme`

Наконец, можно просмотреть список служб, требующихся для данной, но отсутствующих в системе:

Листинг 12: Запрос списка служб, необходимых Postfix, но отсутствующих
`# /etc/init.d/postfix broken`

4.b. Использование rc-update

Что такое rc-update?

Система инициализации Gentoo использует дерево зависимостей для определения служб, которые запускаются в первую очередь. Т. к. это очень утомительное занятие, и мы не хотели, чтобы пользователь занимался этим вручную, мы разработали инструменты, упрощающие управление уровнями запуска и сценариями инициализации.

Используя **rc-update**, можно включать и исключать сценарии инициализации из уровней запуска. Из **rc-update** автоматически запускается сценарий **depscan.sh** для перестроения дерева зависимостей.

Добавление и удаление служб

В процессе установки Gentoo вы уже добавляли сценарии инициализации в уровень запуска «default». В тот момент вы, возможно, не имели понятия, что такое «default» и зачем он нужен, но теперь вы все это знаете. Сценарию **rc-update** требуется второй аргумент, определяющий действие: **add** (добавить), **del** (удалить) или **show** (показать).

Для того, чтобы добавить или удалить сценарий, просто введите **rc-update** с аргументом **add** или **del**, затем название сценария и уровня запуска. Например:

Листинг 13: Удаление Postfix из уровня запуска default
rc-update del postfix default

По команде **rc-update show** выводится список всех доступных сценариев с указанием соответствующих уровней запуска:

Листинг 14: Получение информации о сценариях инициализации
rc-update show

4.c. Настройка служб

Почему нужна дополнительная настройка?

Сценарии инициализации могут быть весьма сложны. Поэтому нежелательно допускать непосредственное редактирование сценария пользователями, т.к. это может привести в систему множество ошибок. Но, с другой стороны, необходимо правильно настроить службу. Например, может понадобиться передать службе дополнительные параметры.

Вторая причина, по которой настройки хранятся отдельно от самого сценария —

это возможность обновления сценария без опасения, что все ваши настройки будут утеряны.

Каталог `/etc/conf.d`

В Gentoo предусмотрен очень простой способ настройки служб: для каждого сценария, предполагающего настройку, в каталоге `/etc/conf.d` есть конфигурационный файл. Например, у сценария, запускающего **apache2** (под названием `/etc/init.d/apache2`) есть конфигурационный файл `/etc/conf.d/apache2`, где могут храниться нужные вам параметры, передаваемые серверу Apache 2 при запуске:

Листинг 15: Переменная, определенная в `/etc/conf.d/apache2`
`APACHE2_OPTS="-D PHP4"`

Такие файлы настроек содержат одни переменные (наподобие `/etc/make.conf`), облегчая настройку служб. Это также позволяет нам давать больше информации о переменных (в комментариях).

4.d. Написание сценариев инициализации

Мне тоже придется?..

Нет, написание сценариев инициализации обычно не требуется, т.к. Gentoo содержит готовые сценарии для всех поддерживаемых служб. Однако, вы можете установить какую-либо службу, не используя систему Portage; в таком случае, вероятно, вам придется создавать сценарий инициализации самостоятельно.

Внимание: не используйте сценарий, идущий со службой, если он не написан специально для Gentoo: сценарии инициализации Gentoo не совместимы со сценариями, используемыми в других дистрибутивах!

Структура

Основная структура сценария инициализации показана ниже.

Листинг 16: Основная структура сценария

```
#!/sbin/runscript

depend() {
    (информация о зависимостях)
}

start() {
    (команды, необходимые для запуска службы)
}
```

```
stop() {  
    (команды, необходимые для остановки службы)  
}  
  
restart() {  
    (команды, необходимые для перезапуска службы)  
}
```

В любом сценарии должна быть определена функция `start()`. Все остальные разделы необязательны.

Зависимости

Можно определять два типа зависимостей: **use** (использую) и **need** (нуждаюсь). Как упоминалось ранее, **need**-зависимость более строга, чем **use**-зависимость. Вслед за типом зависимости указывается название службы, от которой существует зависимость, или ссылка на виртуальную (**virtual**) зависимость.

Виртуальная зависимость — это зависимость от функций, предоставляемых службой, но не какой-то единственной службой. Сценарий может зависеть от службы системного журнала, но таких достаточно много (**metalogd**, **syslog-ng**, и и т.п.). Поскольку нельзя нуждаться в каждой из них (ни в одной вразумительной системе они не запущены все сразу), мы обеспечили предоставление виртуальной зависимости всеми этими службами.

Давайте взглянем на информацию о зависимостях postfix.

Листинг 17: Информация о зависимостях Postfix

```
depend() {  
    need net  
    use logger dns  
    provide mta  
}
```

Как можно увидеть, postfix:

- требует сеть (net): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/net.eth0;
- использует журнал (logger): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/syslog-ng;
- использует службу имен (dns): виртуальная зависимость, удовлетворяемая, например, /etc/init.d/named);
- предоставляет почтовый агент (mta): виртуальная зависимость, общая для всех программ — почтовых серверов.

Порядок запуска

Иногда вам нужна не сама служба, а запуск вашей службы до (или после) другой службы, если та присутствует в системе (обратите внимание на условие: это уже не зависимость) и запускается на том же уровне запуска (отметьте условие: это относится только к службам из одинакового уровня запуска). Такую очередность

можно указать, используя значения **before** (до) или **after** (после).

Например, рассмотрим значения для службы Portmap:

Листинг 18: Функция `depend()` службы Portmap

```
depend() {
    need net
    before inetd
    before xinetd
}
```

Также можно использовать знак «*», чтобы охватить все службы данного уровня запуска, хотя это не рекомендуется.

Листинг 19: Запуск сценария первым на уровне запуска

```
depend() {
    before *
}
```

Стандартные функции

Следом за разделом **depend()** вам потребуется определить функцию **start()**. В ней содержатся все команды, необходимые для запуска вашей службы. Рекомендуется применять функции **ebegin** и **end** для сообщений пользователю о том, что происходит:

Листинг 20: Пример функции `start()`

```
start() {
    ebegin "Запуск - моя_служба"
    start-stop-daemon --start --quiet --exec /path/to/my_service
    end $?
}
```

Если вам нужны дополнительные примеры функции **start()**, пожалуйста, прочитайте исходные коды сценариев инициализации, находящиеся в каталоге `/etc/init.d`. Что касается команды **start-stop-daemon**, то на случай, если вам нужны дополнительные сведения, есть превосходная страница справки:

Листинг 21: Вызов страницы справки по `start-stop-daemon`

```
# man start-stop-daemon
```

Другими функциями, которые можно определить — `stop()` и `restart()`. От вас не требуется определение этих функций! Система инициализации, применяемая нами, достаточно развита и в состоянии самостоятельно заполнить эти функции, если вы используете **start-stop-daemon**.

Синтаксис сценариев инициализации, применяемых в Gentoo, основан на оболочке Борна (Bourne Again Shell — `bash`), поэтому вы можете свободно использовать внутри своих сценариев `bash`-совместимые конструкции.

Добавление дополнительных параметров

Если вы хотите ввести в сценарий дополнительные параметры, кроме упоминавшихся, нужно добавить к переменной `opts` название параметра и создать функцию с названием, соответствующим параметру. Например, для поддержки параметра `restartdelay`:

Листинг 22: Создание дополнительной функции `restartdelay`
`opts="${opts} restartdelay"`

```
restartdelay() {  
    stop  
    sleep 3 # пауза в 3 секунды перед повторным запуском  
    start  
}
```

Переменные для настройки служб

Для поддержки конфигурационного файла в каталоге `/etc/conf.d` ничего дополнительно делать не нужно: при запуске вашего сценария инициализации автоматически включаются следующие файлы (т.е., переменные из них становятся доступны):

```
/etc/conf.d/<ваш сценарий инициализации>  
/etc/conf.d/basic  
/etc/rc.conf
```

Если ваш инициализационный сценарий предоставляет виртуальную зависимость (например, `net`), то также включается файл, соответствующий этой зависимости (например, `/etc/conf.d/net`).

4.e. Изменение поведения уровней запуска

Кто от этого выиграет?

Большинству пользователей ноутбуков знакома ситуация: дома вам нужен запуск **net.eth0**, и наоборот, в дороге запуск **net.eth0** не нужен (так как сеть недоступна). В Gentoo можно изменять поведение уровней запуска по своему усмотрению.

Например вы можете создать второй загружаемый уровень запуска «по умолчанию», в котором будут другие сценарии. Затем при загрузке вы сможете выбрать, какой из уровней по умолчанию следует использовать.

Использование программного уровня (softlevel)

Прежде всего, создайте каталог для своего второго уровня запуска «по умолчанию». Например, создадим уровень запуска `offline`:

Листинг 23: Создание каталога уровня запуска
`# mkdir /etc/runlevels/offline`

Добавьте необходимые сценарии инициализации в только что созданный уровень запуска. Например, чтобы получить точную копию уровня **default**, за исключением **net.eth0**:

Листинг 24: Добавление нужных сценариев инициализации
 (копирование всех служб с уровня default в уровень offline)
`# cd /etc/runlevels/default`
`# for service in *; do rc-update add $service offline; done`
 (удаление ненужных сценариев с уровня offline)
`# rc-update del net.eth0 offline`
 (просмотр сценариев, запускаемых на уровне offline)
`# rc-update show offline`
 (часть выведенного списка)
 acpid | offline
 domainname | offline
 local | offline
 net.eth0 |

Теперь необходимо отредактировать конфигурацию загрузчика, добавив запись об уровне **offline**. Например, в файле **/boot/grub/grub.conf**:

Листинг 25: Добавление записи об уровне offline
 title Автономное использование Gentoo Linux
 root (hd0,0)
 kernel (hd0,0)/kernel-2.4.25 root=/dev/hda3 softlevel=offline

Вуаля, все готово. Теперь, если при загрузке вы выберете вновь созданную запись, то вместо default будет использоваться уровень offline.

Использование загрузочного уровня (bootlevel)

Использование загрузочного уровня полностью аналогично использованию программного уровня. Единственная разница состоит в том, что вы определяете второй уровень «boot» вместо «default».

5. Переменные среды

5.а. Переменные среды

Что это такое?

Переменная среды — это именованный объект, который содержит информацию, используемую одним или несколькими приложениями.

Многие пользователи (особенно новички в Linux) находят этот подход несколько странным или неуправляемым. Но это впечатление ошибочно: используя переменные среды, можно очень легко изменить настройку разнообразных программ.

Важные примеры

В следующей таблице описывается ряд переменных, используемых в системе Linux. Примеры их значений приведены далее.

Таблица 1. Переменные среды Linux

Переменная	Описание
PATH	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых система ищет исполняемые файлы. Если вы вводите имя исполняемого файла например, ls , rc-update или emerge), который не находится ни в одной из перечисленных здесь каталогов, этот файл не запустится (если, конечно, вы не указали полный путь, например, /bin/ls).
ROOTPATH	У этой переменной такое же значение, что и у PATH , но в ней перечисляются только те каталоги, которые нужно просматривать при вводе команды пользователем с правами root .
LDPATH	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых динамический компоновщик ищет библиотеки.
MANPATH	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых команда man ищет страницы справки.
INFODIR	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых команда info ищет info -страницы.
PAGER	В этой переменной содержится путь к программе, позволяющей постранично просматривать содержимое файлов, например less или more .
EDITOR	В этой переменной содержится путь к программе, используемой для изменения файлов, например vi или nano .
KDEDIRS	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых находятся ресурсы KDE.

CLASSPATH	В этой переменной содержится список каталогов, разделенных двоеточиями, в которых находятся классы Java.
CONFIG_PROTECT	В этой переменной содержится список каталогов, защищаемых Portage при обновлении, разделенных пробелами.
CONFIG_PROTECT_MASK	В этой переменной содержится список каталогов, исключаемых из защиты Portage при обновлении, разделенных пробелами

Ниже представлен пример определения всех этих переменных:

Листинг 1: Пример определения

```
PATH="/bin:/usr/bin:/usr/local/bin:/opt/bin:/usr/games/bin"
ROOTPATH="/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin"
LDPATH="/lib:/usr/lib:/usr/local/lib:/usr/lib/gcc-lib/i686-pc-linux-gnu/3.2.3"
MANPATH="/usr/share/man:/usr/local/share/man"
INFODIR="/usr/share/info:/usr/local/share/info"
PAGER="/usr/bin/less"
EDITOR="/usr/bin/vim"
KDEDIRS="/usr"
CLASSPATH="/opt/blackdown-jre-1.4.1/lib/rt.jar:."
CONFIG_PROTECT="/usr/X11R6/lib/X11/xkb /opt/tomcat/conf \
    /usr/kde/3.1/share/config /usr/share/texmf/tex/generic/config/ \
    /usr/share/texmf/tex/latex/config/ /usr/share/config"
CONFIG_PROTECT_MASK="/etc/gconf"
```

5.b. Глобальное определение переменных

Каталог `/etc/env.d`

Для того, чтобы определить эти переменные централизованно, в Gentoo появился каталог `/etc/env.d`. В нём находится ряд файлов, например, **00basic**, **05gcc** и так далее, в которых определяются переменные, необходимые программам, указанным в названии файлов.

Например, при установке **gcc ebuild** создает файл **05gcc**, содержащий следующие определения переменных:

Листинг 2: `/etc/env.d/05gcc`

```
PATH="/usr/i686-pc-linux-gnu/gcc-bin/3.2"
ROOTPATH="/usr/i686-pc-linux-gnu/gcc-bin/3.2"
MANPATH="/usr/share/gcc-data/i686-pc-linux-gnu/3.2/man"
INFOPATH="/usr/share/gcc-data/i686-pc-linux-gnu/3.2/info"
CC="gcc"
CXX="g++"
```

```
LDPATH="/usr/lib/gcc-lib/i686-pc-linux-gnu/3.2.3"
```

В других дистрибутивах вам предлагается изменять или добавлять определения переменных среды в **/etc/profile** или где-нибудь еще. Gentoo, с другой стороны, облегчает вам (и Portage) поддержку и управление переменными среды, избавляя от необходимости уделять внимание многочисленным файлам, содержащим определения переменных.

Например, когда обновляется **gcc**, также без малейшего участия пользователя обновляется и **/etc/env.d/05gcc**.

От этого выигрывает не только Portage, но и вы, пользователь. Иногда от вас может потребоваться глобальная установка какой-нибудь переменной. Возьмем, к примеру, переменную **http_proxy**. Вместо того, чтобы возиться с **/etc/profile**, теперь можно просто создать файл (**/etc/env.d/99local**) и добавить нужные определения туда:

Листинг 3: **/etc/env.d/99local**
`http_proxy="proxy.server.com:8080"`

Используя один и тот же файл для всех своих переменных, вы можете быстро увидеть все определенные вами переменные вместе.

Сценарий **env-update**

Переменная **PATH** определяется в нескольких файлах в **/etc/env.d**. Нет, нет это не ошибка: при запуске **env-update** различные определения объединяются перед обновлением переменных среды, позволяя пакетам (или пользователям) добавлять собственные значения переменных, не влияя на уже существующие.

Сценарий **env-update** объединяет значения переменных из файлов, находящихся в **/etc/env.d**, в алфавитном порядке. Имена файлов должны начинаться с двух десятичных цифр.

Листинг 4: Порядок обновления, используемый **env-update**

```
00basic    99kde-env    99local
+-----+-----+-----+
PATH="/bin:/usr/bin:/usr/kde/3.2/bin:/usr/local/bin"
```

Внимание: объединение выполняется не всегда, а только для следующих переменных: **KDEDIRS**, **PATH**, **CLASSPATH**, **LDPATH**, **MANPATH**, **INFODIR**, **INFOPATH**, **ROOTPATH**, **CONFIG_PROTECT**, **CONFIG_PROTECT_MASK**, **PRELINK_PATH** и **PRELINK_PATH_MASK**. Для всех остальных переменных используется значение, определенное в последнем из файлов (по алфавиту в каталоге **/etc/env.d**).

При запуске сценария **env-update** создаются все переменные среды, и помещаются в **/etc/profile.env** (используемый файлом **/etc/profile**). Кроме того, на

основе значения **LDPATH** создается **/etc/ld.so.conf**. После этого запускается **ldconfig**, чтобы вновь создать файла **/etc/ld.so.cache**, используемый динамическим компоновщиком.

Если вы хотите, чтобы результаты работы **env-update** вступили в силу немедленно, для обновления среды выполните следующую команду. Пользователи, самостоятельно устанавливавшие Gentoo, возможно, помнят ее из указаний по установке:

Листинг 5: Обновление среды
env-update && source /etc/profile

Примечание: Эта команда обновляет переменные только в текущем терминале, в новых консолях и их потомках. То есть, если вы работаете в X11, потребуется или набирать **source /etc/profile** в каждом открываемом терминале, или перезапустить X, чтобы все новые терминалы обращались к новым переменным. Если вы используете диспетчер входа в систему, станьте пользователем с правами **root** и наберите **/etc/init.d/xdm restart**. Если нет, вам придется выйти и снова войти в систему, чтобы X порождала потомков, использующих новые значения переменных.

5.с. Локальное определение переменных

Пользовательские переменные

Далеко не всегда нужно определять переменные глобально. Например, вам может понадобиться добавить **/home/my_user/bin** и текущий рабочий каталог (где вы находитесь) к переменной **PATH**, но при этом не нужно, чтобы это добавление появилось и в переменной **PATH** у всех остальных пользователей. Если вы хотите определить переменную среды локально, используйте **~/.bashrc** или **~/.bash_profile**:

Листинг 6: Расширение **PATH** в **~/.bashrc** для локальных нужд
 (двоеточие без последующего указания каталога означает текущий рабочий каталог)
PATH="\${PATH}:/home/my_user/bin:"

Обновление вашей переменной **PATH** произойдет, когда вы выйдете и снова войдете в систему.

Сеансовые переменные

Иногда нужны еще более жесткие ограничения. Вам может потребоваться возможность запуска исполняемых файлов из специально созданного временного каталога без указания полного пути к ним, и без изменения файла **~/.bashrc** ради нескольких минут.

В этом случае можно просто определить переменную **PATH** для текущего сеанса

командой **export**. Переменной будет присвоено временное значение до тех пор, пока вы не завершите сеанс.

Листинг 7: Определение сеансовой переменной среды
export PATH="\${PATH}:/home/my_user/tmp/usr/bin"

С. Работа с Portage

1. Файлы и каталоги

1.а. Файлы Portage

Директивы настройки

Настройки Portage по умолчанию хранятся в **/etc/make.globals**. Когда вы откроете этот файл, вы увидите, что все настройки представляют собой переменные. Что означает каждая из переменных, описано ниже.

Так как многие директивы отличаются в зависимости от используемой архитектуры, к Portage прилагаются настройки по умолчанию, которые входят в ваш профиль. На ваш профиль указывает символическая ссылка **/etc/make.profile**. Настройка Portage выполняется с помощью файлов **make.defaults** вашего профиля и всех родительских профилей. Более подробно о профилях и каталоге **/etc/make.profile** мы расскажем позже.

Если вы планируете вносить изменения в конфигурационные переменные, не изменяйте **/etc/make.globals** или **make.defaults**. Вместо этого пользуйтесь файлом **/etc/make.conf**, который имеет приоритет перед вышеуказанными файлами. Вы также обнаружите файл **/etc/make.conf.example**. Как понятно из его названия, это просто пример — Portage не использует этот файл.

Переменные Portage также можно устанавливать как переменные среды, но мы не рекомендуем этого делать.

Конфигурация, определяемая профилем

Мы уже встречались с каталогом **/etc/make.profile**. На самом деле это не каталог, а символическая ссылка на профиль, по умолчанию на тот, что содержится в **/usr/portage/profiles**, однако вы можете создавать свои собственные профили где угодно и ссылаться на них. Профиль, указанный ссылкой, является профилем, к которому принадлежит ваша система.

В профиле содержатся сведения для Portage, специфичные для архитектуры, такие как список пакетов, принадлежащих соответствующей системе, список неработоспособных (или замаскированных) пакетов, и т.д.

Конфигурация, задаваемая пользователем

Если вам необходимо изменить поведение Portage относительно установки программного обеспечения, вам потребуется отредактировать файлы,

находящиеся в `/etc/portage`. Мы настоятельно рекомендуем вам пользоваться файлами из `/etc/portage`, всеми силами отговариваем от настройки поведения Portage через переменные среды!

Внутри `/etc/portage` вы можете создать следующие файлы:

- **package.mask**, в котором перечислены пакеты, которые Portage никогда не следует устанавливать;
- **package.unmask**, со списком пакетов, для которых вы хотите иметь возможность установки, даже если разработчики Gentoo отговаривают вас от этого;
- **package.keywords**, где перечислены пакеты, которые должны быть доступны для установки, несмотря на то, что они не подходят для вашей системы или архитектуры (пока);
- **package.use**, где перечислены значения USE-флагов, которые необходимо указывать для конкретных пакетов, а не для всей системы.

Дополнительные сведения о каталоге **`/etc/portage`**, а также список всех файлов, которые там можно создавать, находятся на справочной странице Portage:

Листинг 1: Вызов справки по Portage

```
$ man portage
```

Изменение файлов Portage и размещения каталогов

Ранее упомянутые конфигурационные файлы нельзя хранить где угодно — Portage всегда ищет свои настроечные файлы в строго определенных местах. Однако Portage также использует множество каталогов для других целей: каталог для сборки, место для хранения исходных кодов, место для дерева Portage, и т.д.

Для этих целей существуют хорошо известные каталоги по умолчанию, положение которых можно изменить на свой вкус, внося изменения в `/etc/make.conf`. Оставшаяся часть этой главы посвящена описанию того, какие специальные места Portage использует для своих целей, и как изменить их расположение в файловой системе.

Этот документ не претендует на статус справочника. Если вам необходим полный объем информации, пожалуйста, обратитесь к страницам справки по Portage и `make.conf`:

Листинг 2: Вызов справки по Portage и `make.conf`

```
$ man portage  
$ man make.conf
```

1.b. Хранение файлов

Дерево Portage

Дерево Portage размещается, по умолчанию, в **`/usr/portage`**. Это

определяется значением переменной **PORTDIR**. Когда вы храните дерево Portage где-либо в другом месте (изменив эту переменную), не забывайте соответственно изменить символическую ссылку **/etc/make.profile**.

Если вы измените переменную **PORTDIR**, вам может потребоваться изменить и следующие переменные: **PKGDIR**, **DISTDIR**, **RPMDIR**, так как они не замечают изменений **PORTDIR**. Это связано с особенностями их обработки Portage.

Двоичные пакеты

Несмотря на то, что Portage по умолчанию не использует прекомпилированное программное обеспечение, для него предусмотрена очень мощная поддержка. Если вы укажете Portage работать с прекомпилированными пакетами, они будут разыскиваться в **/usr/portage/packages**. Это расположение определяется переменной **PKGDIR**.

Исходные коды

Исходные коды приложений хранятся в **/usr/portage/distfiles** по умолчанию. Это определяется переменной **DISTDIR**.

Файлы RPM

Несмотря на то, что Portage не может использовать RPM-файлы, есть возможность их создания командой **ebuild** (см. Приложение Ebuild). По умолчанию Portage хранит RPM файлы в каталоге **/usr/portage/rpm**, как определяется переменной **RPMDIR**.

База данных Portage

Portage хранит состояние вашей системы (какие пакеты установлены, какие файлы относятся к определенным пакетам и т. п.) в **/var/db/pkg**. Не изменяйте эти файлы вручную! Это может разрушить знание вашей системы Portage.

Кэш Portage

Кэш Portage (включая сведения о времени изменения, виртуальные пакеты, информацию дерева зависимостей и т. д.) хранится в **/var/cache/edb**. Это место действительно является кэшем: вы можете его очистить в любой момент, когда не запущены приложения, связанные с Portage.

1.с. Сборка программного обеспечения

Временные файлы Portage

По умолчанию Portage хранит временные файлы в **/var/tmp**. За это отвечает переменная **PORTAGE_TMPDIR**.

Если вы измените переменную **PORTAGE_TMPDIR**, вам может потребоваться изменить и переменную **BUILD_PREFIX**, так как она не замечает изменений **PORTAGE_TMPDIR**. Это связано с особенностями ее обработки Portage.

Каталог сборки

Portage создает специфичные каталоги сборки для каждого пакета внутри **/var/tmp/portage**. Это расположение задается переменной **BUILD_PREFIX**.

Размещение «живой файловой системы»

По умолчанию Portage устанавливает все файлы в текущую файловую систему (/), но это можно изменить, установив переменную окружения **ROOT**. Это может оказаться полезным при построении новых образов системы.

1.d. Ведение журнала

Журнал Ebuild

Portage может создавать отдельные файлы журнала для каждого файла **ebuild**, но только тогда, когда переменная **PORT_LOGDIR** указывает на место, доступное для записи для Portage (пользователя portage). По умолчанию эта переменная не установлена.

2. Настройка с помощью переменных

2.a. Настройка Portage

Как отмечалось ранее, Portage настраивается с помощью множества переменных, которые задаются в файле **/etc/make.conf**. За более полной и подробной информацией обращайтесь к странице справки по **make.conf**:

Листинг 1: Чтение страницы справки по make.conf
\$ man make.conf

2.b. Параметры сборки

Параметры конфигурирования и компиляции

Когда Portage собирает приложения, компилятору и сценарию конфигурации передаются значения следующих переменных:

- **CFLAGS** и **CXXFLAGS** определяют желаемые флаги компилятора для C и C++;

- **CHOST** определяет информацию об используемой платформе для сценария конфигурации приложения;
- **MAKEOPTS** передается команде make и обычно применяется для установки степени распараллеливания компиляции. Более подробная информация о параметрах команды make находится на странице справки по make.

Переменная USE также используется при конфигурировании и компиляции, но о ней уже много и подробно говорилось в предыдущих главах.

Параметры установки

Когда Portage устанавливает (**merge**) новую версию программного продукта, файлы более старых версий удаляются из системы. Portage дает пользователю 5-ти секундную задержку перед стиранием старых версий. Эти 5 секунд задаются переменной **CLEAN_DELAY**.

2.с. Защита конфигурационных файлов

Места, защищаемые Portage

Portage записывает файлы, предоставляемые новой версией программы, поверх старых, если только эти файлы не расположены в защищенном месте. Защищенные каталоги определяются переменной **CONFIG_PROTECT**. Обычно, это места расположения файлов конфигурации. Каталоги в списке разделяются пробелами.

Файл, который должен быть записан в такой защищенный каталог, переименовывается, а пользователь получает предупреждение о наличии новой версии (обычно) файла конфигурации.

Узнать текущее значение **CONFIG_PROTECT** можно из сообщений emerge --info:

Листинг 2: Получение значения CONFIG_PROTECT
`$ emerge --info | grep 'CONFIG_PROTECT='`

Более подробная информация о защите конфигурационных файлов, осуществляемой системой Portage, доступна по команде emerge:

Листинг 3: Подробная информация о защите конфигурационных файлов
`$ emerge --help config`

Исключение каталогов

Чтобы снять защиту с определенных подкаталогов защищенного каталога, можно использовать переменную **CONFIG_PROTECT_MASK**.

2.d. Параметры скачивания

Расположение сервера

Если запрошенная информация или данные отсутствуют в вашей системе, Portage обращается за ними в интернет. Расположение серверов для различных каналов получения информации задается следующими переменными:

- **GENTOO_MIRRORS** определяет список адресов серверов, содержащих исходный код (distfiles);
- **PORTAGE_BINHOST** указывает расположение определенного сервера, содержащего двоичные пакеты (prebuilt packages) для вашей системы;
- Третья переменная содержит расположение сервера **rsync**, который используется при обновлении вашего дерева портежей:
SYNC указывает сервер, с которого Portage извлекает дерево портежей.

Переменные **GENTOO_MIRRORS** и **SYNC** можно установить автоматически программой **mirrorselect**. Перед тем, как использовать, ее нужно установить, выполнив **emerge mirrorselect**. За дополнительной информацией обращайтесь к оперативной справке **mirrorselect**:

Листинг 4: Дополнительные сведения о mirrorselect
mirrorselect --help

Если вы вынуждены использовать прокси-сервер, для его указания можно использовать переменные **HTTP_PROXY**, **FTP_PROXY** и **RSYNC_PROXY**.

Команды для извлечения

Когда Portage требуется извлечь исходный код, по умолчанию используется **wget**. Вы можете это изменить с помощью переменной **FETCHCOMMAND**.

Portage может возобновлять скачивание частично загруженного исходного кода. По умолчанию используется **wget**, но это можно переопределить переменной **RESUMECOMMAND**.

Удостоверьтесь, что ваши команды **FETCHCOMMAND** и **RESUMECOMMAND** сохраняют исходный код в нужном месте. Внутри этих переменных следует использовать **\\${URI}** и **\\${DISTDIR}**, для указания расположения исходных кодов и distfiles, соответственно.

Также существует возможность определить индивидуальные настройки для различных протоколов, используя **FETCHCOMMAND_HTTP**, **FETCHCOMMAND_FTP**, **RESUMECOMMAND_HTTP**, **RESUMECOMMAND_FTP**, и т.п.

Настройки rsync

Вы не можете заменить команду `rsync`, которую Portage использует для обновления дерева портежей, но можно установить несколько переменных, определяющих ее поведение:

- **RSYNC_EXCLUDEFROM** указывает на файл, где перечислены пакеты и/или категории, которые `rsync` должна игнорировать во время обновления.
- **RSYNC_RETRIES** определяет, сколько раз `rsync` должна пытаться соединиться с зеркалом, на которое указывает переменная `SYNC`. По умолчанию равна 3.
- **RSYNC_TIMEOUT** определяет количество секунд, в течение которого `rsync` соединение может бездействовать, перед тем как `rsync` сочтет его превысившим время ожидания. По умолчанию равна 180, но если вы используете соединение по модему или у вас медленный компьютер, возможно, следует установить значение этой переменной равным 300 или большим.

2.e. Настройка Gentoo

Выбор ветви

Используемую ветвь можно изменить переменной **ACCEPT_KEYWORDS**. По умолчанию используется стабильная ветвь для вашей архитектуры. Дополнительная информация о ветвях Gentoo находится в следующей главе.

Возможности Portage

Вы можете включить отдельные функции Portage с помощью переменной **FEATURES**. Возможности Portage рассматривались в предыдущих главах.

2.f. Поведение Portage

Распределение ресурсов

С помощью переменной **PORTAGE_NICENESS** можно увеличивать или уменьшать значение **nice**, с которым выполняется Portage. Значение **PORTAGE_NICENESS** прибавляется к текущему значению **nice**.

Более подробно о значениях **nice** написано в странице справки:

Листинг 5: Дополнительные сведения о **nice**
\$ man nice

Настройки вывода

Переменная **NOCOLOR** (по умолчанию «false») определяет, следует ли Portage

отключить цветовую раскраску своих сообщений.

3. Смешение ветвей программного обеспечения

3.а. Использование одной ветви

Стабильная ветвь

Переменная **ACCEPT_KEYWORDS** определяет, какую из ветвей использовать в вашей системе. По умолчанию используется стабильная ветвь для вашей архитектуры, например **x86**

Мы рекомендуем использовать только стабильную ветвь. Однако, если для вас стабильность не критична и вы хотите помочь Gentoo, отсылая отчеты об ошибках на <http://bugs.gentoo.org>, читайте дальше.

Тестовая ветвь

Если вы желаете использовать наиболее свежее ПО, подумайте над использованием тестовой ветви. Чтобы Portage начала использовать тестовую ветвь, добавьте «~» перед названием вашей архитектуры.

Тестовая ветвь полностью соответствует своему названию: для тестирования. Если пакет находится в стадии тестирования, это означает, что разработчики считают, что пакет работоспособен, но тщательно он не протестирован. Вы можете оказаться первым, кто столкнется с какой-либо ошибкой. В этом случае вы можете создать отчет об ошибке, чтобы разработчики узнали о ней.

Однако будьте готовы к тому, что могут возникнуть проблемы со стабильностью, неудовлетворительной поддержкой пакетов (например неправильные/отсутствующие зависимости), слишком частыми обновлениями (а в результате — частыми сборками) или невозможностью собрать пакет. Если вы не знаете, как работает Gentoo и как разрешать возникающие проблемы, мы рекомендуем не отходить от стабильной и оттестированной ветви.

К примеру, для выбора тестовой ветви на архитектуре x86, отредактируйте `/etc/make.conf` и укажите в нем:

Листинг 1: Установка значения переменной `ACCEPT_KEYWORDS`
`ACCEPT_KEYWORDS="~x86"`

Если вы запустите обновление системы, то увидите, что многие пакеты нуждаются в обновлении. Обратите внимание, что после перехода на тестовую ветвь и обновления системы, как правило, нет простого пути назад к стабильной официальной ветви (конечно, кроме использования резервной копии).

3.b. Одновременное использование стабильной и тестовой ветвей

Местоположение `package.keywords`

Вы можете указать, чтобы Portage использовала тестовую ветвь только для определенных пакетов, а для остальной системы — стабильную ветвь. Для этого добавьте категорию и имя пакета, для которого вы желаете использовать тестовую ветвь, в файл `/etc/portage/package.keywords`. Вместо этого можно создать каталог (с таким же именем) и указывать пакеты в файлах, находящихся внутри этого каталога. Например, для использования тестовой ветви для **gnumeric**:

Листинг 2: Настройка `/etc/portage/package.keywords` для `gnumeric`, вся строка
`app-office/gnumeric ~x86`

Тестирование определенных версий

Если вы желаете использовать конкретную версию ПО из тестовой ветви, но не хотите, чтобы Portage использовала тестовую ветвь для последующих версий этого ПО, можно указать в местоположении `package.keywords` номер необходимой версии. В этом случае вы обязаны использовать оператор `=`. Также можно указать диапазон версий, используя операторы `<=`, `<`, `>` или `>=`.

В любом случае, добавляя информацию о версии, вы должны использовать один из этих операторов. Если вы не указываете версию, эти операторы использовать нельзя.

В следующем примере мы просим Portage разрешить установку `gnumeric-1.2.13`:

Листинг 3: Использование конкретной тестовой версии `gnumeric`
`=app-office/gnumeric-1.2.13 ~x86`

3.c. Использование заблокированных пакетов

Расположение `package.unmask`

Разработчики Gentoo не поддерживают использование этого места расположения. Пожалуйста, используйте их на свой страх и риск. Просьбы о помощи, связанные с использованием **`package.unmask`** и/или **`package.mask`**, останутся без ответа. Вы предупреждены.

Если использование пакета было заблокировано разработчиками Gentoo, но вы желаете его использовать несмотря на причины блокировки, указанные в файле `package.mask` (по умолчанию он находится в `/usr/portage/profiles`), добавьте для него точно такую же строку в файл `/etc/portage/package.unmask` (или в файл в этом каталоге, если это каталог).

Например, если `=net-mail/hotwayd-0.8` заблокирован, то разблокировать его можно, прописав в `package.unmask` точно такую же строчку:

Листинг 4: `/etc/portage/package.unmask`
`=net-mail/hotwayd-0.8`

Местоположение `package.mask`

Если вы не хотите, чтобы Portage использовала какое-то конкретное ПО или конкретные версии ПО, вы можете его самостоятельно заблокировать, добавив соответствующую запись в `/etc/portage/package.mask` (в такой файл либо в файл внутри такого каталога).

Если, к примеру, вы не хотите, чтобы Portage устанавливала исходные коды ядра новее, чем `gentoo-sources-2.6.8.1`, добавьте такую строку в местоположение `package.mask`:

Листинг 5: Пример использования файла `/etc/portage/package.mask`
`>sys-kernel/gentoo-sources-2.6.8.1`

4. Дополнительные средства Portage

4.a. `etc-update`

etc-update — это утилита, предназначенная для обновления в системе файлов `._cfg0000_<имя>`. Она обеспечивает интерактивную настройку установки и может также автоматически устанавливать тривиальные изменения. Файлы создаются `._cfg0000_<имя>` Portage, когда нужно заменить файл в каталоге, защищенном переменной **CONFIG_PROTECT**.

Выполнить **etc-update** довольно просто:

Листинг 1: Запуск `etc-update`
`# etc-update`

После выполнения тривиальных обновлений, вы увидите запрос со списком защищенных файлов, ожидающих обновления. Внизу вам предложат следующие варианты:

Листинг 2: Запрос `etc-update`
Please select a file to edit by entering the corresponding number.
(-1 to exit) (-3 to auto merge all remaining files)
(-5 to auto-merge AND not use 'mv -i'):

(Пожалуйста, выберите файл для правки, введя соответствующее число.
(-1 - выход) (-3 - автоустановка всех оставшихся файлов)

(-5 для автоустановки БЕЗ использования 'mv -i'):)

При вводе -1, etc-update выходит, прекращая последующие изменения. Если вы введете -3 или -5, все перечисленные файлы конфигурации заменяются более новыми версиями. Следовательно, очень важно сначала отобрать файлы, которые не следует автоматически обновлять. Для этого надо только вводить номер, указанный слева от файлов.

Например, выбираем файл конфигурации /etc/pear.conf:

Листинг 3: Обновление конкретного конфигурационного файла
 Beginning of differences between /etc/pear.conf and /etc/._cfg0000_pear.conf
 [...]
 End of differences between /etc/pear.conf and /etc/._cfg0000_pear.conf
 1) Replace original with update
 2) Delete update, keeping original as is
 3) Interactively merge original with update
 4) Show differences again

Теперь можно увидеть различия между двумя файлами. Если вы считаете, что обновленный файл конфигурации можно использовать без проблем, введите 1. Если вы считаете, что обновленный файл конфигурации не нужен, или не содержит новую или полезную информацию, введите 2. Если вы хотите обновить текущий файл в интерактивном режиме, введите 3.

Нет никакого смысла в подробном описании интерактивного обновления. Для полноты изложения, мы перечислим возможные команды, которые можно использовать при интерактивном слиянии двух файлов. Вас встречают две строки (одна исходная, вторая измененная) и запрос, в ответ на который можно ввести одну из следующих команд:

Листинг 4: Команды, доступные при интерактивном слиянии
 ed: редактировать и использовать оба варианта, каждый пометить заголовком
 eb: редактировать и использовать оба варианта
 el: редактировать и использовать левый вариант
 er: редактировать и использовать правый вариант
 e: редактировать новую версию
 l: использовать левую версию
 r: использовать правую версию
 s: молча включить общие строки
 v: включить общие строки, сообщив подробности
 q: выход

Завершив обновление важных файлов конфигурации, вы можете автоматически обновить оставшиеся файлы конфигурации. etc-update выйдет, если не найдет других файлов, подлежащих обновлению.

4.b. dispatch-conf

С помощью **dispatch-conf** можно обновлять файлы конфигурации, сохраняя при этом историю изменений. **dispatch-conf** хранит различия между файлами конфигурации в виде заплаток или в системе управления версиями RCS.

Как и с **etc-update**, вы можете попросить сохранить файл конфигурации как есть, использовать новый файл конфигурации, редактировать текущий или объединить изменения интерактивно. Однако, у **dispatch-conf** также есть приятные дополнительные возможности:

- автоматическое обновление файлов, в которых обновились только комментарии;
- автоматическое обновление файлов, которые отличаются только количеством пробелов.

Убедитесь, что вы сначала отредактировали **/etc/dispatch-conf.conf** и создали каталог, прописанный в **archive-dir**.

За дополнительными сведениями обращайтесь к странице справки **dispatch-conf**:

Листинг 5: Чтение справки по **dispatch-conf**
\$ man dispatch-conf

4.c. quickpkg

С **quickpkg** вы можете создавать архивы пакетов, уже установленных в системе. Эти архивы можно использовать в качестве двоичных пакетов. Запуск **quickpkg** прост: только укажите имена пакетов, которые нужно заархивировать.

Например, чтобы поместить в архив **curl**, **arts** и **procps**:

Листинг 6: Пример использования **quickpkg**
quickpkg curl arts procps

Двоичные пакеты будут храниться в **\$PKGDIR/All** (по умолчанию — **/usr/portage/packages/All**). Символьные ссылки, указывающие на эти пакеты, помещаются в **\$PKGDIR/<категория>**.

5. Отступление от официального дерева

5.a. Использование собственного дерева Portage

Исключение пакета/категории

Вы можете выборочно обновлять определенные категории/пакеты, игнорируя обновление других категорий/пакетов. Это достигается путем исключения таких категорий/пакетов программой **rsync** на этапе выполнения **emerge --sync**.

Вам потребуется определить имя файла, содержащего шаблоны исключаемых пакетов, в переменной **RSYNC_EXCLUDEFROM** в своем файле `/etc/make.conf`.

Листинг 1: Указание файла исключаемых пакетов в `/etc/make.conf`
RSYNC_EXCLUDEFROM=/etc/portage/rsync_excludes

Листинг 2: Исключение всех игр в файле `/etc/portage/rsync_excludes`
games-*/*

Заметьте, однако, что это может привести к проблемам с зависимостями, так как новые разрешенные пакеты могут зависеть от других новых, но исключенных из обновления пакетов.

5.b. Добавление неофициального сборочного файла ebuild

Определение оверлейного каталога портежей

Вы можете указать Portage использовать сборочные файлы, не входящие в официальное дерево Portage. Создайте новый каталог (к примеру, `/usr/local/portage`), в котором будут находиться файлы **ebuild** сторонних разработчиков. Используйте в точности такую же структуру каталогов, как и в официальном дереве портежей!

Затем определите переменную **PORTDIR_OVERLAY** в `/etc/make.conf`, чтобы она указывала на ранее созданный каталог. Теперь при использовании Portage, эти сборочные файлы будут рассматриваться как часть системы, и не будут удаляться/перезаписываться при последующих запусках **emerge --sync**.

Работа с несколькими оверлейными каталогами

Для продвинутых пользователей, ведущих разработку в нескольких оверлейных каталогах, тестирующих пакеты перед включением в основное дерево портежей или просто желающих использовать неофициальные сборочные файлы ebuild из разных источников, в пакете `app-portage/gentoolkit-dev` есть утилита `gensync`, которая поможет поддерживать ваши оверлейные репозитории в актуальном состоянии.

Используя **gensync**, вы можете обновить сразу все репозитории или выбрать для обновления только некоторые из них. В каждом репозитории в каталоге `/etc/gensync/` должен находиться файл **.syncsource**, в котором содержится информация о местоположении репозитория, его имени, идентификаторе и т.д.

Предположим, что у вас есть два дополнительных репозитория с названиями `java` (для сборочных файлов разработок, ведущихся на `java`) и `entapps` (для внутренних приложений, разработанных на вашем предприятии). Вы можете обновить эти репозитории следующей командой:

Листинг 3: Запуск `gensync` для обновления нескольких репозиториях
`# gensync java entapps`

5.с. Программы, поддерживаемые не Portage

Использование Portage с пакетами самостоятельной сборки

Иногда вам может потребоваться сконфигурировать, установить и поддерживать программное обеспечение самостоятельно, без автоматизации со стороны Portage, не смотря на то, что оно поддерживается Portage. Наиболее известные случаи — это исходные коды ядра и драйверы от nVidia. Вы можете настроить Portage так, чтобы системе стало известно, что определенные пакеты установлены вручную. Этот процесс называется внедрение, и поддерживается Portage посредством файла `/etc/portage/profile/package.provided`.

Например, если вы захотите сообщить Portage, что пакет `vanilla-sources-2.6.11.6` установлен вручную, нужно добавить следующую строку в `/etc/portage/profile/package.provided`:

Листинг 4: Пример строки из файла `package.provided`
`sys-kernel/vanilla-sources-2.6.11.6`

6. Использование *ebuild*

6.a. Emerge и Ebuild

Программа **ebuild** — это низкоуровневый интерфейс системы Portage. С ее помощью можно выполнять определенные действия над заданными сборками **ebuild**. Например, вы можете самостоятельно выполнить отдельные этапы установки.

Программа **ebuild** предназначена в основном для разработчиков, поэтому более подробная информация находится в настольной книге разработчика (англ.). Однако, мы расскажем, какие экземпляры **ebuild** вызываются системой Portage на разных этапах установки, и как выполнить пост-конфигурационные шаги, которые допускаются некоторыми пакетами.

6.b. Ручная установка программ

Извлечение исходных кодов и проверка контрольных сумм

Каждый раз, когда вы вызываете **ebuild** для какого-то **ebuild**-файла, проверяется совпадение контрольной суммы всех задействованных файлов с указанной в файлах **Manifest** или `files/digest-<имя>-<версия>`. Проверка выполняется после загрузки исходных кодов.

Чтобы загрузить исходные коды с помощью **ebuild**, запустите:

Листинг 1: Загрузка исходных кодов
ebuild путь/к/файлу-ebuild fetch

Если контрольная сумма md5 сборочного файла не совпадает с той, что указана в файле **Manifest**, или же один из загруженных файлов не совпадает с описанием в файле `files/digest<пакет>`, вы получите сообщение об ошибке, похожее на такое:

Листинг 2: Ошибка контрольной суммы **ebuild**
 !!! File is corrupt or incomplete. (Digests do not match)
 >>> our recorded digest: db20421ce35e8e54346e3ef19e60e4ee
 >>> your file's digest: f10392b7c0b2bbc463ad09642606a7d6

(!!! Файл поврежден или усечен. (Контрольные суммы не совпадают))

На следующей строке указывается проблемный файл.

Если вы абсолютно уверены, что загруженные исходные коды и сам сборочный файл **ebuild** именно те, что вам нужны, можете пересоздать файлы **Manifest** и `digest-<пакете>`, используя функцию `digest` программы **ebuild**:

Листинг 3: Создание новых файлов **Manifest** и `digest`
ebuild путь/к/файлу-ebuild digest

Распаковка исходных кодов

Чтобы распаковать исходные коды в `/var/tmp/portage` (или любой другой каталог, указанный в `/etc/make.conf`), запустите функцию `unpack` программы **ebuild**:

Листинг 4: Распаковка исходных кодов
ebuild путь/к/файлу-ebuild unpack

Эта команда выполнит функцию `src_unpack()` программы **ebuild** (которая по умолчанию просто выполняет распаковку, если функция `src_unpack()` не определена). Все необходимые заплатки накладываются также на этом этапе.

Компиляция исходных кодов

Следующий шаг в процессе установки — компиляция исходных кодов. Для этого выполняется функция `src_compile()` вашего сборочного файла. Если нужно, заодно выполняется конфигурация.

Листинг 5: Компиляция исходных кодов
ebuild путь/к/файлу-ebuild compile

Если вы хотите изменить инструкции компиляции, советуем отредактировать функцию `src_compile()`. Однако, вы можете также обмануть Portage, заставив ее поверить, что программа `ebuild` уже завершила компиляцию. Запустите нужные команды самостоятельно и создайте пустой файл `.compile` в рабочем каталоге.

Листинг 6: Сообщение Portage о завершении задания компиляции
touch .compiled

Установка файлов во временное место

Следующий шаг — установка всех необходимых файлов во временный каталог. В него помещаются все файлы, подлежащие включению в рабочую файловую систему. Вы можете выполнить этот этап, запустив функцию установки программы `ebuild`, которая исполняет функцию `src_install()` сборочного файла.

Листинг 7: Установка файлов
ebuild путь/к/файлу-ebuild install

Помещение файлов в рабочую файловую систему

Последний этап — перенос всех файлов в рабочую файловую систему и их регистрация в системе Portage. В `ebuild` этот этап называется «`qmerge`», и включает следующие действия:

- выполняется функция `pkg_preinst()`, если она определена;
- все файлы копируются в рабочую файловую систему;
- файлы регистрируются в системе Portage;
- выполняется функция `pkg_postinst()`, если она определена;

Запустите функцию **`qmerge`** программы **`ebuild`**, чтобы выполнить этот этап:

Листинг 8: Помещение файлов в рабочую файловую систему
ebuild путь/к/файлу-ebuild qmerge

Очистка временного каталога

Наконец, можно очистить временный каталог, используя команду `clean` программы `ebuild`:

Листинг 9: Очистка временного каталога
ebuild путь/к/файлу-ebuild clean

6.c. Дополнительные возможности Ebuild

Запуск всех команд установки

С помощью функции **merge** программы **ebuild**, можно запустить команды извлечения, распаковки, компиляции, установки и помещения за один раз:

Листинг 10: Установка программы
ebuild путь/к/файлу-ebuild merge

Выполнение действий по настройке

В некоторых приложениях содержатся инструкции по дальнейшей настройке установленного пакета. Эти инструкции могут потребовать участия пользователя, и, следовательно, не выполняться автоматически. Для запуска шагов настройки, указанных в необязательной функции **config()** сборочного файла, используйте команду **config** программы **ebuild**:

Листинг 11: Настройка пакета
ebuild путь/к/файлу-ebuild config

Сборка пакета (RPM)

Вы можете попросить Portage создать двоичный пакет или даже RPM из вашего сборочного файла, воспользовавшись командами **package** и **rpm**, соответственно. Эти команды несколько различаются:

- команда **package** во многом похожа на **merge**, выполняя все необходимые шаги (извлечение, распаковку, компиляцию, установку) перед созданием пакета;
- команда **rpm** собирает пакет RPM из файлов созданных после запуска окончания функции **install** программы **ebuild**.

Листинг 12: Создание пакетов
 (создание двоичного пакета, совместимого с Portage)
ebuild путь/к/файлу-ebuild package

(создание пакета RPM)
ebuild путь/к/файлу-ebuild rpm

Созданный RPM, однако, не будет содержать информацию о зависимостях из сборочного файла **ebuild**.

6.d. Дополнительная информация

За дополнительными сведениями о системе **Portage**, программе **ebuild** и сценариях **ebuild** обращайтесь к следующим страницам справки **man**:

Листинг 13: Страницы справки

```
$ man portage  (сама система Portage)
$ man emerge  (команда emerge)
$ man ebuild  (команда ebuild)
$ man 5 ebuild (синтаксис файлов ebuild)
```

Кроме того, дополнительные сведения, относящиеся к разработке, находятся в настольной книге разработчика (англ.).

D. Настройка сети в Gentoo

1. Начальная настройка

1.a. Приступаем к настройке

Примечание: В документе предполагается, что вы правильно сконфигурировали свое ядро и модули для оборудования, и вам известно интерфейсное имя устройств. Мы также предполагаем, что вы настраиваете **eth0**, хотя на самом деле это может оказаться **eth1**, **wlan0** и т.д.

Примечание: Требуется, чтобы у вас использовался baselayout-1.11.11 или более свежий.

Для начала настройки своей сетевой платы, нужно рассказать о ней системе Gentoo RC. Это делается созданием символической ссылки с net.lo на net.eth0 в /etc/init.d.

Листинг 1: Создание символической ссылки с net.lo на net.eth0

```
# cd /etc/init.d
# ln -s net.lo net.eth0
```

Теперь система **Gentoo** RC знает об этом интерфейсе. Ей также нужно знать, как настраивать новый интерфейс. Конфигурация всех сетевых интерфейсов находится в **/etc/conf.d/net**. Вот простая настройка для использования DHCP или статического адреса.

Листинг 2: Примеры для /etc/conf.d/net

```
# использование DHCP
config_eth0=( "dhcp" )

# статический IP-адрес, используется запись CIDR
config_eth0=( "192.168.0.7/24" )
routes_eth0=( "default via 192.168.0.1" )

# статический IP-адрес, запись с маской подсети
config_eth0=( "192.168.0.7 netmask 255.255.255.0" )
routes_eth0=( "default gw 192.168.0.1" )
```

Примечание: Если конфигурация для интерфейса не указывается, предполагается использование DHCP.

Примечание: CIDR расшифровывается как Classless InterDomain Routing (бесклассовая междоменная маршрутизация). Первоначально, адреса IPv4 были разделены на классы A, B и C. Ранняя система классификации не была рассчитана на массовую популярность интернета, и попала под угрозу исчерпания новых уникальных адресов. CIDR — это схема адресации, позволяющая одному

IP-адресу обозначать множество IP-адресов. IP-адрес CIDR выглядит как обычный IP-адрес с добавлением косой черты и числа; например, 192.168.0.0/16. CIDR описывается в RFC 1519.

Теперь, настроив интерфейс, мы можем запускать и останавливать его следующими командами:

Листинг 3: Сценарии запуска и остановки сети

```
# /etc/init.d/net.eth0 start
# /etc/init.d/net.eth0 stop
```

Важно: При поиске неисправностей сети рекомендуется установить `RC_VERBOSE="yes"` в `/etc/conf.d/rc` для получения более подробной информации о происходящем.

Теперь, успешно запустив и остановив сетевой интерфейс, вы можете захотеть, чтобы он запускался при каждой загрузке Gentoo. Вот как это сделать. Последняя команда «rc» указывает Gentoo, что нужно запускать в текущем уровне запуска любые еще не запущенные сценарии.

Листинг 4: Настройка запуска сетевого интерфейса при загрузке

```
# rc-update add net.eth0 default
# rc
```

2. Расширенная настройка

2.а. Расширенная настройка

Переменная `config_eth0` служит основой конфигурации интерфейса. Она содержит список высокоуровневых инструкций по настройке интерфейса (в данном случае, `eth0`). Все команды списка выполняются последовательно. Интерфейс считается работоспособным, если хотя бы одна команда выполнена успешно.

Вот список встроенных инструкций:

Команда	Описание
null	Не выполнять никаких действий
noop	Если интерфейс включен и существует адрес, успешно завершить настройку.
an IPv4 or IPv6 address	Добавить адрес к интерфейсу
dhcpr, adsl or apipa(или команда запуска модуля стороннего изготовителя)	Запустить модуль, реализующий команду. Например, <code>dhcpr</code> запускает модуль, реализующий команду. Например, <code>dhcpr</code> запускает модуль, реализующий DHCP, которым может быть <code>dhcpcd</code> , <code>udhcprc</code> , <code>dhclient</code> или <code>pump</code> .

На случай неудачного выполнения команды можно указать запасную команду. Запасной вариант должен строго соответствовать структуре конфигурации.

Команды можно сцеплять. Вот несколько практических примеров.

Листинг 1: Примеры настройки

Задание трех адресов IPv4

```
config_eth0=(
  "192.168.0.2/24"
  "192.168.0.3/24"
  "192.168.0.4/24"
)
```

Задание одного адреса IPv4 и двух адресов IPv6

```
config_eth0=(
  "192.168.0.2/24"
  "4321:0:1:2:3:4:567:89ab"
  "4321:0:1:2:3:4:567:89ac"
)
```

Сохранять адрес, присвоенный ядром, до отключения интерфейса.

При этом назначить другой через DHCP. Если DHCP не работает,

задать статический адрес, определяемый APIPA

```
config_eth0=(
  "noop"
  "dhcp"
)
fallback_eth0=(
  "null"
  "apipa"
)
```

Примечание: При использовании модуля `ifconfig` для назначения нескольких адресов, для каждого дополнительного адреса создаются псевдонимы интерфейса. Так, в двух примерах, приведенных выше, создаются интерфейсы `eth0`, `eth0:1` и `eth0:2`. С этими интерфейсами нельзя сделать ничего особенного, так как и ядро, и другие программы обрабатывают `eth0:1` и `eth0:2` просто как `eth0`.

Важно: Порядок настройки запасного режима имеет значение! Если бы мы не указали инструкцию `null`, то команда `apipa` запускалась бы только при неудачном выполнении команды `noop`.

Примечание: APIPA и DHCP обсуждаются позже.

2.b. Сетевые зависимости

Сценарии инициализации в `/etc/init.d` могут находиться в зависимости от определенного сетевого интерфейса или просто от службы сети (`net`). Определив переменную `RC_NET_STRICT_CHECKING` в `/etc/conf.d/rc`, службе `net` можно придать различный смысл.

Значение	Описание
none	Служба <code>net</code> считается всегда работающей
no	В основном это означает, что по крайней мере одна служба <code>net.*</code> , кроме <code>net.lo</code> , должна работать. Это может пригодиться пользователям ноутбуков, у которых есть WIFI и статическое проводное подключение, когда нужно, чтобы при включении хотя бы одного интерфейса служба сети выглядела включенной.
lo	То же, что и <code>no</code> , но с учетом <code>net.lo</code> . Может быть полезно для тех, кого не волнует, чтобы определенный интерфейс включался при загрузке.
yes	В этом случае ВСЕ сетевые интерфейсы ДОЛЖНЫ работать, чтобы служба <code>net</code> считалась работающей.

Но как насчет `net.br0`, зависящего от `net.eth0` и `net.eth1`? `net.eth1` может быть беспроводным или PPP-устройством, требующим предварительной настройки для возможности включения в мост. Это невозможно сделать в `/etc/init.d/net.br0`, так как он является символьной ссылкой на `net.lo`.

Ответом является создание своей собственной функции `depend()` в `/etc/conf.d/net`.

Листинг 2: Зависимость `net.br0` в `/etc/conf.d/net`
 # Можно использовать любую зависимость (use, after, before),
 # как видно в текущих сценариях
`depend_br0() {`
 `need net.eth0 net.eth1`
`}`

Более подробно зависимости обсуждаются в разделе Написание сценариев инициализации Настольной книги Gentoo.

2.с. Имена и значения переменных

Имена переменных являются динамическими. Обычно они следуют структуре `variable_${interface|mac|ssid|apmac}`. Например, значение переменной `dhcpcd_eth0` хранит параметры `dhcpcd` для `eth0`, а переменной `dhcpcd_essid` — параметры `dhcpcd`, используемые при подключении любого интерфейса к ESSID «`essid`».

Однако, не существует твердого простого правила, устанавливающего, что интерфейсы должны называться `ethx`. На деле, имена многих беспроводных выглядят как `wlanx`, `rax` и `ethx`. Кроме того, некоторые пользовательские

интерфейсы, например, мосты, можно называть как угодно, например, foo. Для пущего разнообразия, в именах беспроводных точек доступа также допускаются знаки, не входящие в алфавитно-цифровые; это имеет значение, потому что есть возможность настройки сетевых параметров для отдельных ESSID.

Оборотная сторона всего этого в том, что для настройки сети в Gentoo используются переменные bash, а bash не в состоянии использовать что-либо кроме знаков английского алфавита и цифр. Чтобы обойти такое ограничение, мы заменяем каждый символ, не являющийся английским буквенно-цифровым, на знак подчеркивания: `_`.

Другая особенность bash — это значения переменных: некоторые символы требуют специальной записи, перед ними помещается знак `\`. Им необходимо предварять следующие символы: `"`, `'` и `\`.

В следующем примере мы используем беспроводные ESSID, так как в них может содержаться самое широкое множество символов. Мы воспользуемся ESSID `My "\ NET`:

Листинг 3: Пример имени переменной

```
# Этот пример работает, но домен не существует
dns_domain_My____NET="My \"\\ NET"
```

```
# Предыдущая строка устанавливает домен dns в My "\ NET при
# подключении беспроводной платы к точке доступа с ESSID My "\ NET.
```

3. Модульное построение сети

3.a. Сетевые модули

Сейчас мы поддерживаем модульные сетевые сценарии. Это значит, что мы можем легко добавлять поддержку для новых типов интерфейсов и конфигурационных модулей, сохраняя совместимость с существующими.

По умолчанию, модули загружаются только если пакет, нужный им, установлен. Если отметить модуль, для которого не установлен пакет, будет выдана ошибка с указанием, какой пакет нужно установить. В идеале, настройка модулей требуется только тогда, когда установлено несколько пакетов, представляющих одну и ту же службу, а вам установить приоритет одного из них.

Примечание: Все обсуждаемые значения хранятся в `/etc/conf.d/net`, если явно не указано иное.

Листинг 1: Предпочтение модуля

```
# выбор iproute2, а не ifconfig
modules=( "iproute2" )
```

```
# можно также указать другие модули для отдельного интерфейса
```

```
# здесь мы выбираем udhcpc, а не dhcpcd
modules_eth0=( "udhcpc" )
```

```
# также можно указать, какие модули не надо использовать: например,
# возможно, вы используете supplicant или linux-wlan-ng для управления
# параметрами беспроводной сети, но при этом желаете настраивать сетевые
# параметры отдельно для каждого связанного ESSID
modules=( "!iwconfig" )
```

3.b. Обработчики интерфейса

Сейчас мы предоставляем два обработчика интерфейса: **ifconfig** и **iproute2**. Для настройки сети вам нужен только один из них.

- **ifconfig** в текущем Gentoo используется по умолчанию, и включен в системный профиль.
- **iproute2** — более мощный и гибкий пакет, который не включен в системный профиль по умолчанию.

Листинг 2: Установка iproute2
emerge sys-apps/iproute2

```
# выбор iproute2, а не ifconfig, когда установлены оба
modules=( "iproute2" )
```

Так как и **ifconfig** и **iproute2** делают очень сходные вещи, то мы сделали их базовую настройку взаимозаменяемой. Например, оба приведенных ниже примера работают не зависимо от того, какой модуль используется.

Листинг 3: Примеры ifconfig и iproute2
`config_eth0=( "192.168.0.2/24" )`
`config_eth0=( "192.168.0.2 netmask 255.255.255.0" )`

```
# также можно указать широковещательный адрес
config_eth0=( "192.168.0.2/24 brd 192.168.0.255" )
config_eth0=( "192.168.0.2 netmask 255.255.255.0 broadcast 192.168.0.255" )
```

3.c. DHCP

DHCP — это способ получения сетевой информации (адреса IP, сервера DNS, шлюза и т.д.) с сервера. Это значит, что если в сети запущен сервер DHCP, вам остается только сказать каждому клиенту, чтобы он использовал DHCP, и сеть настроится сама собой. Конечно, вам придется настраивать все остальное (беспроводную сеть, подключение точка-точка и т.д.), если они должны работать до использования DHCP.

Поддержка DHCP обеспечивается **dhclient**, **dhcpcd**, **pump** или **udhcpc**. У каждого модуля DHCP есть свои плюсы и минусы: здесь мы быстро рассмотрим их.

Модуль DHCP	Пакет	Плюсы	Минусы
dhclient	net-misc/dhcp	Сделан ISC, теми же людьми, кто делает BIND DNS. Гибок в настройке.	Настройка чрезмерно сложна, программа довольно «распухшая», не может получать данные о серверах NTP с DHCP, по умолчанию не отправляет имя узла.
dhcpcd	net-misc/dhcpcd	Давно в Gentoo по умолчанию, не зависит от внешних утилит.	Более не поддерживается разработчиком, может быть временами медленным, не становится демоном при неограниченном сроке аренды адреса.
pump	net-misc/pump	Компактный, не зависит от внешних утилит.	Более не поддерживается разработчиком, ненадежен, особенно по модему, не может получать данные о серверах NIS по DHCP.
udhcp	net-misc/udhcp	Компактный; наименьший существующий клиент DHCP, сделан для встроенных систем.	Не зарекомендовал себя — ни в одном дистрибутиве не используется по умолчанию; не поддерживает длительность таймаута более 3 секунд.

Если у вас установлено больше одного DHCP клиента, вам нужно указать, какой использовать; иначе по умолчанию используется **dhcpcd**, если есть.

Чтобы передать определенные параметры модулю DHCP, используйте `модуль_eth0="..."` (замените модуль на имя используемого модуля DHCP, например, `dhcpcd_eth0`).

Мы попытались сделать DHCP относительно агностиком: по существу, мы поддерживаем следующие команды, с использованием переменной `dhcp_eth0`. По умолчанию не включена ни одна из них.

release — освобождать IP-адрес для повторного использования

nodns — не замещать `/etc/resolv.conf`

nontp — не замещать /etc/ntp.conf

nonis — не замещать /etc/yp.conf

Листинг 4: Простая настройка DHCP в /etc/conf.d/net
требуется только если у вас несколько модулей DHCP
modules=("dhcpcd")

```
config_eth0=( "dhcp" )  
dhcpcd_eth0="-t 10" # прекращение после 10 секунд  
dhcpcd_eth0="release nodns nontp nonis" # только получать адрес
```

Примечание: По умолчанию, dhcpcd, udhcpc и pump передают текущее узла на сервер DHCP, поэтому его больше не требуется указывать.

3.d. Модем ADSL

Сначала нужно установить программное обеспечение для ADSL.

Листинг 5: Установка пакета gr-pppoe
emerge net-dialup/gr-pppoe

Предупреждение: В baselayout-1.11.x поддерживается только PPPoE. Надеемся, что в будущих версиях появится поддержка PPPoA.

Сейчас нам нужно указать, что на eth0 будет ADSL-интерфейс, и ввести наше имя пользователя, обновив /etc/conf.d/net.

Листинг 6: Настройка eth0 для ADSL в /etc/conf.d/net

```
config_eth0=( "adsl" )  
adsl_user_eth0="имя-пользователя"
```

Наконец, нужно указать ваше имя и пароль в /etc/ppp/pap-secrets.

Листинг 7: Пример /etc/ppp/pap-secrets
* обязательна
"пользователь" * "пароль"

3.e. APIPA (автоматическая частная IP-адресация)

2APIPA пытается найти свободный адрес в диапазоне 169.254.0.0-169.254.255.255, проверяя отклик на интерфейсе произвольного адреса из этого диапазона по протоколу arp. Если отклика нет, адрес назначается интерфейсу.

Это полезно только в локальных сетях, где нет сервера DHCP, нет прямого подключения к интернету, и все другие компьютеры используют APIPA.

Для поддержки APIPA установите **net-misc/iputils** или **net-analyzer/arping**.

Листинг 8: Настройка APIPA в /etc/conf.d/net
 # сначала пробуем DHCP, при неудаче переходим на APIPA
 config_eth0=("dhcp")
 fallback_eth0=("apiipa")

 # использование только APIPA
 config_eth0=("apiipa")

3.f. Объединение интерфейсов

Для объединения каналов в ствол (bonding) установите **net-misc/ifenslave**.

Объединение используется для повышения пропускной способности сети. Если у вас есть две сетевых карты, выходящих в одну и ту же сеть, можно объединить их, так что ваши приложения увидят только один интерфейс, но реально будут пользоваться двумя сетевыми платами.

Листинг 9: Настройка объединения в /etc/conf.d/net
 # объединение интерфейсов
 slaves_bond0="eth0 eth1 eth2"

 # вы можете не захотеть назначать адрес IP объединенному интерфейсу
 config_bond0=("null")

 # указание зависимости от eth0, eth1 и eth2, так как им может требоваться
 # дополнительная настройка
 depend_bond0() {
 need net.eth0 net.eth1 net.eth2
 }

3.g. Образование моста (поддержка 802.1d)

Для поддержки мостов установите **net-misc/bridge-utils**.

Мосты используются для объединения сетей. Например, у вас может быть сервер, подсоединенный к интернету через ADSL-модем, и плата беспроводного доступа для предоставления доступа в интернет через ADSL модем другим компьютерам. Чтобы соединить оба интерфейса, можно создать «мост».

Листинг 10: Настройка моста в /etc/conf.d/net
 # настройка моста: подробности в "man brctl"
 brctl_br0=("setfd 0" "sethello 0" "stp off")

 # включаем порты в мост br0
 bridge_br0="eth0 eth1"

```
# устанавливаем порты в "null", чтобы не запускался dhcp
config_eth0=( "null" )
config_eth1=( "null" )

# наконец, даем мосту адрес; можно использовать и DHCP
config_br0=( "192.168.0.1/24" )

# указываем зависимость от eth0 и eth1, так как им может потребоваться
# дополнительная настройка
depend_br0() {
    need net.eth0 net.eth1
}
```

Важно: Для использования некоторых вариантов моста вам может потребоваться обращение к документации по именам переменных.

3.h. MAC-адрес

Для изменения MAC-адреса своего интерфейса вам не потребуется ничего устанавливать, если у вас **sys-apps/baselayout-1.11.14** или новее, и вы собираетесь сменить MAC-адрес на какой-то определенный. Однако, если вам нужно сменить MAC-адрес на случайный, или ваш baselayout старше указанной версии, для смены адреса потребуется установить пакет командой **emerge net-analyzer/macchanger**.

Листинг 11: Пример изменения MAC-адреса

```
# установка MAC-адреса интерфейса
mac_eth0="00:11:22:33:44:55"

# случайная установка последних 3 байт адреса
mac_eth0="random-ending"

# установка случайного адреса из диапазона для физического соединения
# того же типа (оптического, медного, беспроводного) любого изготовителя
mac_eth0="random-samekind"

# установка случайного адреса из диапазона для любого типа физического
# соединения (оптического, медного, беспроводного) любого изготовителя
mac_eth0="random-anykind"

# полностью случайный; ВНИМАНИЕ, некоторые MAC-адреса, сгенерированные
# таким образом, могут вести себя НЕ ТАК, как предполагается
mac_eth0="random-full"
```

3.i. Образование туннеля

Для образования туннеля вам не нужно ничего устанавливать, поскольку на это способен сам обработчик интерфейса.

Листинг 12: Настройка тоннеля в /etc/conf.d/net

```
# для тоннелей GRE
iptunnel_vpn0="mode gre remote 207.170.82.1 key 0xffffffff ttl 255"

# для тоннелей IPIP
iptunnel_vpn0="mode ipip remote 207.170.82.2 ttl 255"

# для настройки интерфейса
config_vpn0=( "192.168.0.2 peer 192.168.1.1" )
```

3.j. Виртуальные сети (поддержка 802.1q)

Для поддержки VLAN, установите **net-misc/vconfig**.

Виртуальная локальная сеть (VLAN) — это группа сетевых устройств, которые ведут себя, как будто подключены к одному сегменту сети, даже когда это не так. Членам VLAN видны только члены той же VLAN даже если в той же физической сети присутствуют другие.

Листинг 13: VLAN configuration in /etc/conf.d/net

```
# указание номеров VLAN для интерфейса
# пожалуйста, убедитесь, что ваши номера VLAN НЕ дополнены нулем
vlans_eth0="1 2"

# можно также настроить VLAN
# за подробностями обращайтесь к man vconfig
vconfig_eth0=( "set_name_type VLAN_PLUS_VID_NO_PAD" )
vconfig_vlan1=( "set_flag 1" "set_egress_map 2 6" )

# настройка интерфейса как обычно
config_vlan1=( "172.16.3.1 netmask 255.255.254.0" )
config_vlan2=( "172.16.2.1 netmask 255.255.254.0" )
```

Важно: Для использования некоторых вариантов VLAN вам может потребоваться обращение к документации по именам переменных.

4. Беспроводная сеть

4.a. Введение

В настоящее время поддерживается подключение к беспроводной сети с помощью **wireless-tools** или **wpa_supplicant**. Важно помнить, что подключение к беспроводным сетям настраивается глобально, а не для определённого интерфейса.

- **wpa_supplicant** — лучший выбор, но он поддерживает не все драйверы. Список поддерживаемых драйверов находится на сайте [wpa_supplicant](http://wpa_supplicant.org). Кроме того, сейчас **wpa_supplicant** может подключаться только к тем сетям, на SSID которых настроен.

- **wireless-tools** поддерживает практически все платы и драйверы, но не способен подключаться к точкам доступа, работающим исключительно с WPA.

Предупреждение: Драйвер linux-wlan-ng в данный момент не поддерживается в baselayout. Это из-за того, что в linux-wlan-ng своя собственная программа установки и настройки, которая ни на что не похожа. Разработчики linux-wlan-ng, по слухам, собираются перейти на установку как в wireless-tools; когда это произойдет, вы сможете использовать linux-wlan-ng с baselayout.

4.b. Запросчик WPA

Запросчик **WPA (WPA Supplicant)** — пакет, позволяющий подсоединяться к точкам доступа с протоколом WPA. Его настройка проходит достаточно гладко, и пакет работает достаточно стабильно, хотя находится на стадии бета-тестирования.

Листинг 1: Установка wpa_supplicant
emerge net-wireless/wpa_supplicant

Важно: Для работы wpa_supplicant в ядре должен быть включен параметр **CONFIG_PACKET**.

Теперь нам необходимо настроить **/etc/conf.d/net** для предпочтения **wpa_supplicant** по отношению к **wireless-tools** (по умолчанию, если обе программы установлены, работает **wireless-tools**).

Листинг 2: Настройка /etc/conf.d/net для wpa_supplicant
выбор wpa_supplicant
modules=("wpa_supplicant")

важно указать wpa_supplicant, какой драйвер нужно использовать,
так как программа пока не слишком хорошо угадывает сама

wpa_supplicant_eth0="-Дбезумный-wifi"

Примечание: Если вы используете драйвер **host-ap**, то вам потребуется перевести плату в ведомый режим (managed mode), прежде чем она сможет правильно работать с **wpa_supplicant**. Для этого можно указать iwconfig_eth0="mode managed" в /etc/conf.d/net.

Это довольно просто, не так ли? Однако, нужно настроить саму программу wpa_supplicant, что значительно сложнее. Сложность зависит от степени защиты точек доступа, к которым вы собираетесь подключаться. Следующий упрощенный пример взят из /etc/wpa_supplicant.conf.example, поставляемого в составе wpa_supplicant.

Листинг 3: Пример /etc/wpa_supplicant.conf

следующую строку нельзя изменять, иначе программа не сможет работать
ctrl_interface=/var/run/wpa_supplicant

ограничим доступ к настройкам WPA только для root
ctrl_interface_group=0

пусть wpa_supplicant заботится о сканировании и выборе точки доступа
ap_scan=1

простой случай: WPA-PSK, согласованный ключ - текстовая строка,
принимать любой допустимый шифр
network={
 ssid="просто"
 psk="очень тайный пароль"
 # чем выше приоритет, тем скорее выбор
 priority=5
}

как в предыдущем, но с запросом сканирования по определенному SSID
(для точек доступа, отклоняющих широковещательный SSID)
network={
 ssid="второй ssid"
 scan_ssid=1
 psk="очень тайный пароль"
 priority=2
}

использовать только WPA-PSK; принимать любое допустимое сочетание шифров

network={
 ssid="пример"
 proto=WPA
 key_mgmt=WPA-PSK
 pairwise=CCMP TKIP
 group=CCMP TKIP WEP104 WEP40
 psk=06b4be19da289f475aa46a33cb793029d4ab3db7a23ee92382eb0106c72ac7bb
 priority=2
}

открытое подключение без шифрования (не WPA, не IEEE #802.1X)

network={
 ssid="тест-открытого-текста"
 key_mgmt=NONE
}

подключение с общим ключом WEP (не WPA, не IEEE #802.1X)

network={
 ssid="тест-статического-wep"
 key_mgmt=NONE
 wep_key0="abcde"
 wep_key1=0102030405
 wep_key2="1234567890123"
 wep_tx_keyidx=0
 priority=5
}

```
# подключение с общим ключом WEP (не WPA, не IEEE #802.1X),
# допуск с использованием общего ключа IEEE 802.11
network={
    ssid="тест2-статического-wep"
    key_mgmt=NONE
    wep_key0="abcde"
    wep_key1=0102030405
    wep_key2="1234567890123"
    wep_tx_keyidx=0
    priority=5
    auth_alg=SHARED
}

# сеть IBSS/ad-hoc с WPA-None/TKIP
network={
    ssid="тест adhoc"
    mode=1
    proto=WPA
    key_mgmt=WPA-NONE
    pairwise=NONE
    group=TKIP
    psk="тайный пароль"
}
```

4.с. Утилиты Wireless tools

Начальная установка и режим ведомого

Wireless Tools обеспечивают общий способ настройки базовых беспроводных интерфейсов, вплоть до защиты **WEP**. Хотя **WEP** является слабым методом защиты, он наиболее распространен.

Для настройки **Wireless Tools** служат несколько основных переменных. В примере файла настроек, приведенном ниже, описано все, что вам потребуется. Нужно помнить, что отсутствие настройки означает «подключаться к нешифрующей точке доступа с самым сильным сигналом». Программа будет всегда пытаться подключить вас к чему-нибудь.

Листинг 4: Установка wireless-tools
emerge net-wireless/wireless-tools

Примечание: Хотя вы можете хранить свои параметры настройки беспроводной сети в `/etc/conf.d/wireless`, это руководство рекомендует хранить их в `/etc/conf.d/net`.

Важно: Вам понадобится обратиться к документации по именам переменных.

Листинг 5: Пример настройки iwconfig /etc/conf.d/net
приоритет использования iwconfig над wpa_supplicant
modules=("iwconfig")

```
# Настройка ключей WEP для точек доступа ESSID1 и ESSID2
# Можно указывать до 4 ключей WEP, но только 1 может работать в каждый
# момент, поэтому мы указываем индекс по умолчанию [1], чтобы сделать ключ [1],
# а впоследствии снова, чтобы изменить активный ключ на [1].
# Это нужно, если вы настраиваете другие ESSID на использование WEP-ключей,
# отличающихся от [1].
#
# Приставка s: перед ключом означает, что ключ текстовый, иначе -
# шестнадцатичный
#
# enc open указывает открытую защиту (более безопасно)
# enc restricted указывает ограниченную защиту (менее безопасно)
key_ESSID1="[1] s:ваш-ключ-здесь key [1] enc open"
key_ESSID2="[1] aaaa-bbbb-cccc-dd key [1] enc restricted"
#
# Нижеследующее работает только при поиске доступных точек доступа.

# Иногда видны несколько точек доступа, и требуется задать
# предпочтительный порядок подключения
preferred_aps=( "ESSID1" "ESSID2" )
```

Настройка порядка выбора точки доступа

Можно указать несколько дополнительных параметров для уточнения порядка выбора точки доступа, но обычно этого не требуется.

Вам решать, подключаться ли только к указанным точкам доступа, или нет. По умолчанию, если подключение ко всем настроенным точкам доступа не удалось, и есть возможность подключиться к не шифрующей точке доступа, такое соединение произойдет. Этот порядок зависит от переменной **associate_order**. Ниже приводится таблица значений и с описанием их действия:

Значение	Описание
any	поведение по умолчанию
preferredonly	соединяться только с видимыми точками доступа из списка
forcepreferred	принудительно подключаться к точкам доступа в заданной последовательности, если они не обнаружены при сканировании
forcepreferredonly	не сканировать точки доступа, просто пытаться подключиться к каждой по списку
forceany	так же, как в forcepreferred + подключаться к любой доступной точке доступа

Наконец, мы можем указать **blacklist_aps** и **unique_ap**. **blacklist_aps** работает подобно **preferred_aps**. **unique_ap** устанавливается в **yes** или **no**, указывая, можно ли подключать второй беспроводной интерфейс к той же точке доступа, что и первый.

Листинг 6: Пример **blacklist_aps** и **unique_ap**

```
# иногда требуется полностью исключить возможность подключения
# к определенным точкам доступа
blacklist_aps=( "ESSID3" "ESSID4" )

# если у вас несколько беспроводных плат, можно указать, можно ли им
# подключаться к одной и той же точке доступа
# значение - "yes" или "no"
# по умолчанию - "yes"
unique_ap="yes"
```

Режим отдельного и ведущего узла

Если вы хотите становиться отдельным узлом (ad hoc), когда не удается подключиться ни к какой точке доступа в ведомом режиме, это тоже возможно.

Листинг 7: Откат к режиму отдельного узла

```
adhoc_essid_eth0="Этот отдельный узел"
```

Как насчет подключения к сетям Ad-Hoc или запуска в режиме ведущего (master), чтобы стать точкой доступа? Есть конфигурация и для такой работы! Вам может потребоваться определить WEP-ключи, как показано выше.

Листинг 8: Пример настройки ad-hoc/master

```
# установка режима: допускается managed (ведомый, по умолчанию),
# ad-hoc (отдельный) или master (ведущий). Не все драйверы поддерживают
# каждый режим
mode_eth0="ad-hoc"
```

```
# установка ESSID интерфейса
# в ведомом режиме заставляет интерфейс пытаться подключиться к указанному
# ESSID, и больше ничего
essid_eth0="Этот отдельный узел"
```

```
# если не указан, используется канал 3
channel_eth0="9"
```

Важно: Следующий текст взят дословно из документации BSD wavelan, входящей в документацию NetBSD. «Существуют 14 каналов. Нам сообщили, что использование каналов с 1 по 11 является законным в Северной Америке, каналов с 1 по 13 — в большинстве стран Европы, каналов с 10 по 13 — во Франции, и только канала 14 — в Японии. Если у вас есть сомнения, обратитесь к документации от вашей платы или точки доступа. Убедитесь что выбранный канал совпадает с каналом точки доступа (или другой платы в сети ad-hoc). По умолчанию на платах, продаваемых в Северной Америке и большинстве стран Европы, настроен канал 3; на платах, продаваемых во Франции — канал 11; на платах, продаваемых в Японии — канал 14.»

Устранение неполадок в wireless tools

Существуют дополнительные переменные, которые можно использовать для запуска своего беспроводного оборудования и устранения неполадок, возникших

из-за драйвера или проблем с сетевым окружением. Ниже приведена таблица прочих функций, которые можно перепробовать:

Переменная	Значение по умолчанию	Описание
iwconfig_eth0		За подробными сведениями о параметрах iwconfig обращайтесь к странице справки iwconfig.
iwpriv_eth0		За подробными сведениями о параметрах iwpriv обращайтесь к странице справки iwpriv
sleep_scan_eth0	0	Время задержки в секундах перед попыткой сканирования. Требуется, когда драйверу или прошивке нужно дополнительное время для перехода в рабочий режим.
sleep_associate_eth0	5	Время ожидания связи интерфейса с точкой доступа (в секундах) перед переходом к опросу следующей
associate_test_eth0	MAC	Некоторые драйверы не сбрасывают MAC-адрес, связанный с недоступной точкой доступа, при потере или попытке связи. Некоторые драйверы не сбрасывают значение качества сигнала при потере или попытке соединения. Допустимые значения: MAC , quality и all .
scan_mode_eth0		Некоторым драйверам необходимо сканировать в режиме ad-hoc. Если сканирование не удастся, попробуйте указать здесь ad-hoc.
iwpriv_scan_pre_eth0		Посылать интерфейсу некоторые команды iwpriv перед сканированием. За дополнительными сведениями обращайтесь к странице справки iwpriv.
iwpriv_scan_post_eth0		Посылать интерфейсу некоторые команды iwpriv после сканирования. За дополнительными сведениями обращайтесь к странице справки iwpriv.

4.d. Раздельная настройка сети по ESSID

Иногда необходим статический IP при соединении с ESSID1, и DHCP при соединении с ESSID2. На деле, большинство переменных модуля можно определять раздельно по ESSID. Вот как это сделать:

Примечание: Это работает при использовании WPA Supplicant или Wireless Tools.

Важно: Вам потребуется свериться с документацией по именам переменных.

Листинг 9: Назначение сетевых настроек для ESSID

```
config_ESSID1=( "192.168.0.3/24 brd 192.168.0.255" )
routes_ESSID1=( "default via 192.168.0.1" )

config_ESSID2=( "dhcp" )
fallback_ESSID2=( "192.168.3.4/24" )
fallback_route_ESSID2=( "default via 192.168.3.1" )

# можно также указать сервера имен и др.
# ПРЕДУПРЕЖДЕНИЕ: DHCP переопределит настройки, если не указано иное
dns_servers_ESSID1=( "192.168.0.1" "192.168.0.2" )
dns_domain_ESSID1="some.domain"
dns_search_domains_ESSID1="search.this.domain search.that.domain"

# перенастройка производится по MAC-адресу точки доступа;
# это полезно, когда в разных местах есть точки доступа с одинаковым ESSID
config_001122334455=( "dhcp" )
dhcpcd_001122334455="-t 10"
dns_servers_001122334455=( "192.168.0.1" "192.168.0.2" )
```

5. Дополнительные возможности

5.а. Стандартные функции-обработчики

Можно определить четыре функции, которые вызываются при операциях запуска (start) и останова (stop). При вызове функциям передается название интерфейса, так что одна и та же функция может управлять несколькими адаптерами.

Для указания на то, что запуск или останов интерфейса может продолжаться, возвращаемое значение функций **preup()** и **predown()** должно быть нулевым (успешным). Если **preup()** возвращает ненулевое значение, запуск интерфейса прерывается. Если **predown()** возвращает ненулевое значение, не допускается продолжение останова интерфейса.

Возвращаемые значение функций **postup()** и **postdown()** игнорируется, так как показываемая ими ошибка не обрабатывается.

- **\${IFACE}** присваивается название запускаемого/останавливаемого интерфейса.
- **\${IFVAR}** — это значение **\${IFACE}**, преобразованное в имя переменной, разрешенное в **bash**.

Листинг 1: Примеры функций до/после запуска/останова

```
preup() {
```

```

# Проверка соединения интерфейса перед его запуском. Она
# работает лишь с некоторыми сетевыми адаптерами и требует наличия
# установленного пакета mii-diag.
if mii-tool ${IFACE} 2> /dev/null | grep -q 'no link'; then
    ewarn "Интерфейс ${IFACE} не подключен, прерывание запуска"
    return 1
fi

# Проверка соединения интерфейса перед его запуском. Она
# работает лишь с некоторыми сетевыми адаптерами и требует наличия
# установленного пакета ethtool.
if ethtool ${IFACE} | grep -q 'Link detected: no'; then
    ewarn "Интерфейс ${IFACE} не подключен, прерывание запуска"
    return 1
fi

# Не забываем вернуть 0 при успехе
return 0
}

preardown() {
# Назначение этого сценария - проверить наличие корня NFS
# и в этом случае предотвратить останов интерфейсов. Заметьте, что
# определяя функцию preardown(), вы отменяете существующую логику.
# Вот она, на случай если все же понадобится...
if is_net_fs /; then
    eerror "Корневая ФС смонтирована в сети - останов ${IFACE} невозможен"
    return 1
fi

# Не забываем вернуть 0 при успехе
return 0
}

postup() {
# Эту функцию можно использовать, например, для регистрации в
# службе динамического DNS. Другой пример - отправка/прием почты после
# запуска интерфейса.
return 0
}

postdown() {
# Эта функция приводится в основном для полноты... Я не придумал,
# что бы ценное в нее поместить ;-)
return 0
}

```

5.b. Функции-обработчики wireless tools

Примечание: Это не работает вместе с WPA Supplicant, но переменные `${ESSID}` и `${ESSIDVAR}` доступны в функции `postup()`.

Можно определить две функции, вызываемые до и после функции подключения (`associate`). При вызове им сначала передается название интерфейса, так что одна и та же функция может управлять несколькими адаптерами.

Для указания на то, что запуск или останов интерфейса можно продолжать, возвращаемое значение функции `preassociate()` должно быть нулевым (успешным). Если `preassociate()` возвращает ненулевое значение, запуск интерфейса прерывается.

Возвращаемое значение функции `postassociate()` игнорируется, так как показываемая ей ошибка не обрабатывается.

`${ESSID}` присваивается точный **ESSID** точки доступа, к которой вы подключаетесь. **`${ESSIDVAR}`** — это **`${ESSID}`**, преобразованный в имя переменной, разрешенное в `bash`.

Листинг 2: Функции до/после соединения

```
preassociate() {
# Ниже добавляются две конфигурационных переменных, leap_user_ESSID
# и leap_pass_ESSID. Когда они обе настроены на подключаемый ESSID,
# мы запускаем сценарий CISCO LEAP

local user pass
eval user="\${leap_user_${ESSIDVAR}}\"
eval pass="\${leap_pass_${ESSIDVAR}}\"

if [[ -n ${user} && -n ${pass} ]]; then
if [[ ! -x /opt/cisco/bin/leapscript ]]; then
eend "Для поддержки LEAP, выполните emerge net-misc/cisco-aironet-client-utils"
return 1
fi
einfo "Ожидание допуска LEAP на \"${ESSID}/\\\\/\""
if /opt/cisco/bin/leapscript ${user} ${pass} | grep -q 'Login incorrect'; then
ewarn "Вход пользователя ${user} не удался"
return 1
fi
fi

return 0
}

postassociate() {
# Эта функция приводится в основном для полноты... Я не придумал,
# что бы ценное в нее поместить ;- )

return 0
}
```

Примечание: `${ESSID}` и `${ESSIDVAR}` недоступны в функциях `predown()` и `postdown()`.

6. Управление сетью

6.а. Управление сетью

Если вы часто берете компьютер в поездки, то у вас не всегда может быть

возможность подключения к сети Ethernet или к беспроводной точке доступа. Но мы можем захотеть, чтобы сеть заработала автоматически, как только к компьютеру подключен кабель Ethernet или найдена беспроводная точка доступа.

Здесь вы найдете некоторые инструменты, которые помогут это организовать.

Примечание: В этом документе рассказывается только о **ifplugd**, но есть и альтернативные решения, например, **quickswitch**.

6.b. ifplugd

ifplugd — это программа, которая запускает и останавливает интерфейс при подключении или отключении кабеля к сети Ethernet. Также она может обработать подключение к беспроводной точке доступа или появление новых точек доступа.

Листинг 1: Установка ifplugd
emerge sys-apps/ifplugd

Настройка **ifplugd** — довольно простая задача. Файл конфигурации расположен по адресу: **/etc/conf.d/ifplugd**. Для просмотра подробного описания переменных запустите **man ifplugd**.

Листинг 2: Пример конфигурации ifplug
назначение интерфейса для слежения
INTERFACES="eth0"

```
AUTO="no"
BEEP="yes"
IGNORE_FAIL="yes"
IGNORE_FAIL_POSITIVE="no"
IGNORE_RETVAL="yes"
POLL_TIME="1"
DELAY_UP="0"
DELAY_DOWN="0"
API_MODE="auto"
SHUTDOWN="no"
WAIT_ON_FORK="no"
MONITOR="no"
ARGS=""
```

```
# дополнительные параметры ifplugd для указанного интерфейса.
# учтите, установки глобальных переменных игнорируются, если указаны значения
# для конкретного интерфейса
MONITOR_wlan0="yes"
DELAY_UP_wlan0="5"
DELAY_DOWN_wlan0="5"
```

Настольная книга по безопасности Gentoo

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/security/>

С версии: 1.3

Введение

Эта настольная книга предназначена для людей, использующих Gentoo Linux в качестве сервера или чувствующих, что они нуждаются в повышении безопасности системы.

Если после прочтения этой книги вы заинтересуетесь дальнейшим усилением защищенности системы Gentoo, то обратите внимание на проект укрепленной системы Gentoo — [Hardened Gentoo Project](#) (англ.).

А. Безопасность системы

1. Соображения перед началом установки

1.a. Физическая безопасность

Сколько преград не установи, злоумышленник, имея физический доступ к компьютеру, сможет легко их обойти. Несмотря на это, кое-какие меры для относительной защиты от злоумышленника с физическим доступом к вашей машине принять можно. Поместив оборудование в кладовку под замок, вы помешаете злоумышленнику попросту отключить его и унести с собой. Стоит запереть корпус компьютера на замок, чтобы не дать злоумышленнику просто вынуть из него жесткий диск. Чтобы предотвратить загрузку с альтернативного диска, благодаря чему можно просто обойти порядок входа в систему и права доступа, попробуйте установить в BIOS жесткий диск в качестве первого загрузочного устройства и запаролить BIOS. Также важно установить пароль для загрузчика LILO или GRUB, чтобы не дать недобросовестному пользователю загрузить систему в однопользовательском режиме, получив к ней полный доступ. Это подробно описано в третьей главе, в разделах [защита GRUB паролем](#) и [защита LILO паролем](#).

1.b. Планирование демонов/служб

Для начала выпишите службы, которые должны работать на машине. Это поможет вам выбрать наилучшую схему разбивки разделов в системе и позволит лучше спланировать меры безопасности. Конечно же, в этом нет необходимости, если ваш компьютер служит для одной определенной задачи, например в качестве рабочей станции или выделенного межсетевого экрана. В таких случаях не должны запускаться никакие службы, за исключением, возможно, sshd.

Составленный список также может пригодиться для администрирования системы. Вы увидите, что, поддерживая перечень текущих версий, значительно легче

удерживать все в актуальном состоянии, когда в каких-либо из ваших демонов обнаруживаются уязвимости к удаленному доступу.

1.c. Схемы создания разделов

Правила разбиения разделов:

- любое дерево каталогов, в которое пользователь должен иметь возможность записи (например, /home, /tmp), должно размещаться в отдельном разделе с использованием дисковых квот. Это снижает риск переполнения всей файловой системы каким-либо пользователем. Portage использует /var/tmp для компиляции, поэтому этот раздел должен быть большим
- любое дерево каталогов, в которое вы планируете устанавливать программное обеспечение не средствами управления пакетами дистрибутива, следует размещать в отдельном разделе. Согласно [стандарту иерархии файлов](#) (англ.), это /opt или /usr/local. Являясь отдельными разделами, они не удаляются при переустановке системы
- для лишней защиты можно поместить статические данные в отдельный раздел, монтируемый только для чтения. Если вы настоящий параноик, можете пользоваться непerezаписываемым носителем, например, компакт-диском

1.d. Пользователь root

Администратор системы (пользователь root) является наиболее важным пользователем, и его права не следует использовать без крайней необходимости. Если злоумышленник получит доступ с правами администратора, единственным способом восстановить доверие к системе будет переустановить ее.

Золотые правила для root

- всегда создавайте пользователя для каждодневных работ, а если этому пользователю понадобятся права администратора, добавьте его в группу «wheel». Это дает возможность обычным пользователям выполнять команды с правами администратора с помощью su
- никогда не запускайте X или другое пользовательское приложение под root. Права администратора следует использовать только при крайней необходимости: если уязвимость есть в приложении, запущенном от имени пользователя, злоумышленник может получить права этого пользователя. А если приложение запущено с правами администратора, то злоумышленник получит их.
- войдя под учетной записью администратора, всегда указывайте абсолютные пути (или всегда используйте команду su -, которая заменяет переменные среды пользователя на администраторские, обеспечивая наличие в PATH администратора только защищенных каталогов, например /bin и /sbin). Иначе можно обмануть администратора и заставить его выполнить совершенно другое приложение. Если PATH администратора защищен или он использует только абсолютные пути, то можно быть уверенным, что это не произойдет
- если пользователю нужно выполнять лишь несколько команд с правами администратора, то вместо использования учетной записи root, рекомендуется использовать команду sudo. Просто также задумывайтесь,

кому вы даете доступ к этой команде!

- никогда не оставляйте без присмотра терминал, в котором вы зарегистрированы в качестве root

В Gentoo по умолчанию есть некоторая защита от обычных пользователей, пытающихся с помощью `su` получить права администратора. По умолчанию настройки PAM требуют, чтобы пользователь был членом группы «wheel» для того, чтобы запускать команду `su`.

1.е. Правила безопасности

Есть ряд причин набросать правила безопасности для своей системы или сети.

- хорошие правила безопасности позволяют вам наметить подход к безопасности системно, а не сваливать в кучу разрозненные меры. Например, без правил администратор может решить отключить telnet, так как в нем пароли передаются не зашифрованными, оставив доступ по FTP, обладающему таким же недостатком. Хорошие правила безопасности позволяют выявить, какие меры безопасности имеют смысл, а какие — нет
- чтобы выявлять проблемы, проводить аудит или отслеживать злоумышленников, может потребоваться перехват данных, передаваемых по сети, проверка истории входа пользователей и выполнявшихся ими команд, а также просмотр их домашних каталогов. Без указания об этом в документации и явного предупреждения пользователей такие действия могут оказаться незаконными и привести к привлечению вас к ответственности по закону
- похищенные учетные записи пользователей являются одной из самых распространенных угроз безопасности системы. Не объясняя пользователям, почему безопасность важна, и как ее поддерживать (например, не записывать пароли на клочках бумаги на рабочем месте), можно и не надеяться на защищенность пользовательских данных
- хорошо задокументированная схема сети и системы поможет как вам, так и, при необходимости, суду и следствию, отслеживать проникновение злоумышленника и выявлять слабые места по факту взлома. Уведомление правил безопасности о том, что ваша система является частной сетью и несанкционированный доступ запрещен, также способствует надлежащему преследованию нарушителя по закону после его поимки

Надеемся, что теперь необходимость в хороших правилах безопасности более чем ясна.

Сами по себе правила — это документ или набор документов, раскрывающих функции сети или системы (например, какие именно услуги предоставляются), допустимые и запрещенные действия, применение «передового опыта» обеспечения безопасности, и т. д. Все пользователи должны быть знакомы как с правилами безопасности, так и с изменениями, которые вы вносите для поддержания их актуальности. Важно уделять время, чтобы добиваться понимания правил пользователями, объяснять, зачем знакомится под роспись, и что будет, если пользователь явно нарушит их требования (в правилах безопасности об этом должно быть ясно указано). Необходимо повторять ознакомление по крайней мере раз в год, так как правила подвержены изменениям (а также для напоминания пользователям).

Примечание: Создавайте правила, понятные для чтения и не допускающие разночтений в любой ситуации.

Правила безопасности должны охватывать по крайней мере следующие темы:

- допустимые действия
 - экранные заставки
 - обращение с паролями
 - скачивание и установка программ
 - предупреждение, если действия пользователей отслеживаются
 - использование антивирусных программ
- обращение с ценными сведениями (в любом письменном виде, на электронном или бумажном носителе)
 - чистый рабочий стол и хранение сведений ДСП под замком
 - отключение компьютера перед уходом
 - использование средств шифрования
 - передача ключей доверенным сотрудникам
 - обращение с конфиденциальными данными в поездках
- обращение с компьютерным оборудованием в поездках
 - обращение с ноутбуком в поездках и в гостиницах

Для разных пользователей могут требоваться разные типы или уровни доступа, и содержание ваших правил может отличаться, чтобы охватить их все.

В распухших правилах безопасности легко упустить важные моменты. В правилах для ИТ-персонала могут содержаться сведения, закрытые для обычных пользователей. Поэтому есть смысл разделить их на несколько небольших разделов, например, «допустимые действия», «использование паролей», «правила использования электронной почты», «правила удаленного доступа».

Примеры правил безопасности приведены на сайте [проекта правил безопасности SANS](#) (англ.). Если у вас небольшая сеть, и вы считаете, что эти примеры слишком велики, можете обратиться к [руководству по объектовой безопасности \(Site Security Handbook\)](#) (англ.).

2. Закручивание гаек

2.a. USE-флаги

В Gentoo Linux файл `make.conf` содержит USE-флаги, определенные пользователем, а `/etc/make.profile/make.defaults` — USE-флаги по умолчанию. Для данного руководства важны флаги `pam` (Pluggable Authentication Modules — подключаемые модули опознания), `tcpd` (Упаковщики TCP) и `ssl` (Secure Socket Layer). Все они включены в USE-флаги по умолчанию.

2.b. Защита GRUB паролем

В GRUB можно защитить загрузчик паролем двумя различными способами. В первом используется открытый пароль, а во втором — шифрование с использованием `md5+salt`.

Листинг 1: `/boot/grub/grub.conf`

```
timeout 5
password changeme
```

Благодаря этим строкам был добавлен пароль `changeme`. Если во время загрузки не ввести пароль, то GRUB просто загружает вариант по умолчанию.

При добавлении пароля MD5 вы должны преобразовать открытый пароль в зашифрованный в том же формате, что и в файле `/etc/shadow`. За дополнительными сведениями обращайтесь к `man crypt`. Зашифрованный пароль, например `changeme`, может выглядеть вот так: `$1$T7/dgdIJ$dJM.n2wZ8RG.oEiIOWJUs`.

Зашифровать пароль можно прямо в оболочке GRUB:

Листинг 2: `md5crypt` в оболочке `grub`

```
#!/sbin/grub

GRUB version 0.92 (640K lower / 3072K upper memory)

[ Minimal BASH-like line editing is supported. For the first word, TAB
lists
  possible command completions. Anywhere else TAB lists the possible
  completions of a device/filename. ]

grub> md5crypt

Password: *****
(было набрано слово changeme)
Encrypted: $1$T7/dgdIJ$dJM.n2wZ8RG.oEiIOWJUs.

grub> quit
```

Затем скопируйте полученный пароль в `/boot/grub/grub.conf`.

Листинг 3: `/boot/grub/grub.conf`

```
timeout 5
password --md5 $1$T7/dgdIJ$dJM.n2wZ8RG.oEiIOWJUs.
```

Задержка в 5 секунд пригодится в случае, когда система физически недоступна, и нужна возможность перезагрузки без использования клавиатуры. Более подробно о паролях в GRUB вы можете узнать, набрав `info grub`.

2.с. Защита LILO паролем

В LILO также поддерживается два способа защиты с помощью пароля: общий и для отдельного образа, в обоих случаях пароль хранится в открытом виде.

Общий пароль устанавливается в начале файла конфигурации и относится к каждому загрузочному образу:

Листинг 4: `/etc/lilo.conf`

```
password=changeme
restricted
delay=3
```

Пароль для отдельного образа устанавливается следующим образом:

Листинг 5: `/etc/lilo.conf`

```
image=/boot/bzImage
  read-only
  password=changeme
```

```
restricted
```

Если параметр `restricted` не введен, то пароль будет запрашиваться при каждой загрузке.

Чтобы сохранить новые сведения в файле `lilo.conf`, нужно запустить `/sbin/lilo`.

2.d. Ограничение доступа к консоли

С помощью файла `/etc/securetty` можно указать, с каких терминальных устройств (tty) можно входить в систему администратору.

Рекомендуется закомментировать все строки, кроме `vc/1` (если у вас `devfs`), или `tty1` (если у вас `udev`). Благодаря этому суперпользователь сможет входить в систему одновременно только один раз и только с одного терминала.

Примечание: Пользователи, входящие в группу «wheel» по-прежнему смогут воспользоваться `su -`, чтобы стать администратором, работая с других терминалов.

Листинг 6: `/etc/securetty`

```
(для devfs)
vc/1
(для udev)
tty1
```

3. Журналирование

3.a. Введение

Для перехвата предупреждений и ошибок, которые могут свидетельствовать о происходящей атаке или успешном взломе, следует подключать дополнительное журналирование. Часто злоумышленники сканируют систему и «проверяют ее на зуб» перед атакой.

Также немаловажно, чтобы файлы журналов были легко читаемыми и обозримыми. В Gentoo Linux при установке на выбор предлагается 3 разных средства журналирования.

3.b. Журналирование: *Syslogd*

Syslogd является самым распространенным средством журналирования для Linux и Unix. Он способен делать ротацию журналов, однако использование `/usr/sbin/logrotate` с вызовом по расписанию (*logrotate* настраивается с помощью файла `/etc/logrotate.conf`) может оказаться полезней, так как у *logrotate* есть много дополнительных возможностей. Частота ротации подбирается в зависимости от загрузки системы.

Ниже приведен стандартный файл `syslog.conf` с некоторыми добавлениями. Мы раскомментировали строки `cron` и `tty`, и добавили удаленный сервер журналирования. Для дальнейшего усиления журналирования можно настроить ведение журнала в двух местах.

Листинг 1: /etc/syslog.conf

```

# /etc/syslog.conf      Файл настройки syslogd.
#
#                       За дополнительной информацией обращайтесь к
#                       странице справки syslog.conf(5).
#                       Взято из Debian, пока пользуемся им
#                       Daniel Robbins, 5/15/99
#
# Сначала стандартные файлы журналов. Журналирование по подсистемам.
#
auth,authpriv.*          /var/log/auth.log
*.*;auth,authpriv.none   -/var/log/syslog
cron.*                   /var/log/cron.log
daemon.*                 -/var/log/daemon.log
kern.*                   -/var/log/kern.log
lpr.*                    -/var/log/lpr.log
mail.*                   /var/log/mail.log
user.*                   -/var/log/user.log
uucp.*                   -/var/log/uucp.log
local6.debug             /var/log/imapd.log
#
# Журналирование почтовой системы. Разбито на части, чтобы
# легко писать сценарии для разбора этих файлов.
#
mail.info                 -/var/log/mail.info
mail.warn                 -/var/log/mail.warn
mail.err                  /var/log/mail.err
#
# Журналирование новостной системы INN
#
news.crit                  /var/log/news/news.crit
news.err                  /var/log/news/news.err
news.notice               -/var/log/news/news.notice
#
# Журналы, куда может попадать "все подряд".
#
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none   -/var/log/debug
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none        -/var/log/messages
#
# Предупреждения и чрезвычайные сообщения, посылаемые всем вошедшим.
#
*.emerg                   *
*.=alert                  *
#
# Мне нравится вывод сообщений на консоль, но только на ту виртуальную
# консоль, что я обычно оставляю простаивать.
#

```

```

daemon,mail.*;\
    news.=crit;news.=err;news.=notice;\
    *.*=debug;*.*=info;\
    *.*=notice;*.*=warn          /dev/tty8

# Настройка сервера удаленного журналирования
*.* @logserver

# Именованный канал /dev/xconsole - для утилиты `xconsole'. Чтобы
использовать,
# следует запускать `xconsole' с параметром `-file':
#
#   $ xconsole -file /dev/xconsole [...]
#
# ПРИМЕЧАНИЕ: измените список, данный ниже, или вы с ума сойдете, если
# у вас относительно загруженная площадка..
#
#daemon.* ,mail.*;\
#   news.crit;news.err;news.notice;\
#   *.*=debug;*.*=info;\
#   *.*=notice;*.*=warn          | /dev/xconsole

local2.*          --/var/log/ppp.log

```

Злоумышленники, скорее всего, будут стараться уничтожать следы своего пребывания, исправляя или удаляя файлы журналов. Можно затруднить им работу, ведя журналирование на одном или нескольких удаленных серверах журнала, расположенных на других компьютерах. Узнать больше о syslogd можно, запустив `man syslog`.

3.c. Metalog

[Metalog](#), написанный Frank Dennis, не может отправлять журнал на удаленный сервер, но дает преимущество, когда речь о скорости и гибкости журналирования. Он может вести журналы по названию программы, срочности, подсистеме (как и syslogd), и комплектуется разборщиком регулярных выражений, с помощью которых можно запускать внешние сценарии при обнаружении заданных шаблонов. При необходимости это может оказаться очень полезным.

Стандартной настройки обычно достаточно. Если необходимо, чтобы вам отправлялось письмо при любом неверно введенном пароле, используйте один из следующих сценариев.

Для postfix:

Листинг 2: /usr/local/sbin/mail_pwd_failures.sh для postfix

```

#!/bin/sh
echo "$3" | mail -s "Warning (program : $2)" root

```

Для netqmail:

Листинг 3: /usr/local/sbin/mail_pwd_failures.sh for netqmail

```

#!/bin/sh
echo "To: root
Subject:Failure (Warning: $2)
$3

```

```
" | /var/qmail/bin/qmail-inject -f root
```

Не забудьте сделать сценарий исполняемым командой `/bin/chmod +x /usr/local/sbin/mail_pwd_failures.sh`.

Затем в файле `/etc/metalog/metalog.conf` раскомментируйте строку под «Password failures»:

Листинг 4: `/etc/metalog/metalog.conf`

```
command = "/usr/local/sbin/mail_pwd_failures.sh"
```

3.d. Syslog-ng

Syslog-ng, за некоторыми отличиями, обладает теми же возможностями, что и syslog и metalog. Он может фильтровать сообщения по уровню и содержанию (как metalog), вести удаленное журналирование, как syslog, обрабатывать журналы, получаемые от syslogd (даже потоки от Solaris), записывать в TTY, запускать программы, и служить сервером журналирования. То есть, он сочетает лучшие черты обоих средств журналирования с возможностью расширенной настройки.

Ниже приведен слегка подправленный классический конфигурационный файл.

Листинг 5: `/etc/syslog-ng/syslog-ng.conf`

```
options { chain_hostnames(off); sync(0); };

#источник, откуда читать журнал
source src { unix-stream("/dev/log"); internal(); };
source kernsrc { file("/proc/kmsg"); };

#получатели
destination authlog { file("/var/log/auth.log"); };
destination syslog { file("/var/log/syslog"); };
destination cron { file("/var/log/cron.log"); };
destination daemon { file("/var/log/daemon.log"); };
destination kern { file("/var/log/kern.log"); };
destination lpr { file("/var/log/lpr.log"); };
destination user { file("/var/log/user.log"); };
destination mail { file("/var/log/mail.log"); };

destination mailinfo { file("/var/log/mail.info"); };
destination mailwarn { file("/var/log/mail.warn"); };
destination mailerr { file("/var/log/mail.err"); };

destination newscrit { file("/var/log/news/news.crit"); };
destination newserr { file("/var/log/news/news.err"); };
destination newsnotice { file("/var/log/news/news.notice"); };

destination debug { file("/var/log/debug"); };
destination messages { file("/var/log/messages"); };
destination console { usertty("root"); };
destination console_all { file("/dev/tty12"); };
destination xconsole { pipe("/dev/xconsole"); };

#создание фильтров
filter f_authpriv { facility(auth, authpriv); };
filter f_syslog { not facility(authpriv, mail); };
```

```

filter f_cron { facility(cron); };
filter f_daemon { facility(daemon); };
filter f_kern { facility(kern); };
filter f_lpr { facility(lpr); };
filter f_mail { facility(mail); };
filter f_user { facility(user); };
filter f_debug { not facility(auth, authpriv, news, mail); };
filter f_messages { level(info..warn)
    and not facility(auth, authpriv, mail, news); };
filter f_emergency { level(emerg); };

filter f_info { level(info); };
filter f_notice { level(notice); };
filter f_warn { level(warn); };
filter f_crit { level(crit); };
filter f_err { level(err); };
filter f_failed { match("failed"); };
filter f_denied { match("denied"); };

#связывание фильтров с получателями
log { source(src); filter(f_authpriv); destination(authlog); };
log { source(src); filter(f_syslog); destination(syslog); };
log { source(src); filter(f_cron); destination(cron); };
log { source(src); filter(f_daemon); destination(daemon); };
log { source(kernsrc); filter(f_kern); destination(kern); };
log { source(src); filter(f_lpr); destination(lpr); };
log { source(src); filter(f_mail); destination(mail); };
log { source(src); filter(f_user); destination(user); };
log { source(src); filter(f_mail); filter(f_info); destination(mailinfo); };
log { source(src); filter(f_mail); filter(f_warn); destination(mailwarn); };
log { source(src); filter(f_mail); filter(f_err); destination(mailerr); };

log { source(src); filter(f_debug); destination(debug); };
log { source(src); filter(f_messages); destination(messages); };
log { source(src); filter(f_emergency); destination(console); };

#журнал по умолчанию
log { source(src); destination(console_all); };

```

Syslog-ng очень легко настраивается, но с той же легкостью можно что-либо упустить в файле настройки, так как он очень велик. Автор также обещает дополнительные возможности, например, шифрование, опознание, сжатие и обязательную проверку прав доступа (MAC — mandatory access control). С такими возможностями программа станет незаменимой для журналирования в сети, так как злоумышленник не сможет перехватывать журнал.

К тому же у syslog-ng есть еще одно достоинство: он не требует прав администратора для запуска!

3.e. Анализ журналов с помощью Logcheck

Конечно же, ведение журналов — это только полдела. Приложение наподобие Logcheck может значительно облегчить регулярный анализ журналов. Logcheck — это сценарий, который вместе с двоичным файлом logtail, запускается службой cron и сверяет журналы с заданными правилами на предмет подозрительной деятельности. При совпадении он отправляет электронное письмо с соответствующими записями на почтовый ящик администратора.

Logcheck и logtail входят в состав пакета app-admin/logsentry.

Logcheck использует четыре файла для выделения важных событий из обычных сообщений. Этими файлами являются logcheck.hacking, содержащий известные сообщения, встречающиеся при взломе, logcheck.violations, содержащий шаблоны, сигнализирующие о нарушениях безопасности, logcheck.violations.ignore, содержащий ключевые слова, совпадающие с шаблонами в файле нарушений, но которые можно игнорировать, и logcheck.ignore, содержащий записи, которые можно игнорировать.

Предупреждение: Не оставляйте logcheck.violations.ignore пустым. Для обработки журналов Logcheck использует грер, некоторые версии которого считают пустой файл знаком подстановки (wildcard). В таком случае все нарушения будут игнорироваться.

4. Монтирование разделов

4.а. Монтирование разделов

При монтировании разделов с файловой системой ext2, ext3 или reiserfs можно указать различные параметры в файле /etc/fstab. Среди них:

- `nosuid` — игнорировать установленный для файла бит SUID, уподобляя его обычному файлу
- `noexec` — предотвращать запуск файлов с данного раздела
- `nodev` — игнорировать файлы устройств

К сожалению, эти настройки можно легко обойти, задействовав косвенные пути. Тем не менее, установкой `noexec` для `/tmp` можно воспрепятствовать использованию большинства вредоносных программ, разработанных для запуска напрямую из каталога `/tmp`.

Листинг 1: /etc/fstab

```
/dev/sda1 /boot ext2 noauto,noatime 1 1
/dev/sda2 none swap sw 0 0
/dev/sda3 / reiserfs notail,noatime 0 0
/dev/sda4 /tmp reiserfs notail,noatime,nodev,nosuid,noexec 0 0
/dev/sda5 /var reiserfs notail,noatime,nodev 0 0
/dev/sda6 /home reiserfs notail,noatime,nodev,nosuid 0 0
/dev/sda7 /usr reiserfs notail,noatime,nodev,ro 0 0
/dev/cdroms/cdrom0 /mnt/cdrom iso9660 noauto,ro 0 0
proc /proc proc defaults 0 0
```

Предупреждение: Установка `/tmp` в режим `noexec` может помешать нормальной работе различных сценариев.

Примечание: За сведениями о дисковых квотах обращайтесь к разделу [КВОТЫ](#).

Примечание: Я не устанавливаю для `/var` параметры `noexec` или `nosuid`, даже если обычно файлы никогда не запускаются из этого каталога, так как `netqmail` устанавливается в `/var/qmail` и требует разрешения на исполнение и доступ к одному файлу с SUID. Я устанавливаю `/usr` в режим только для чтения, так как я никогда ничего не записываю туда, кроме случая обновления Gentoo. Тогда я перемонтирую файловую систему в режиме чтения-записи, обновляю систему и перемонтирую обратно в режиме только для чтения.

Примечание: Даже если вы не используете `netqmail`, Gentoo все же нужен

установленный бит исполнения для `/var/tmp`, так поскольку сборка `ebuild` выполняется именно здесь. Но если у вас есть насущная необходимость монтировать каталог `/var` с параметром `noexec`, можно указать для этого альтернативный путь.

5. Ограничения пользователей/групп

5.a. `/etc/security/limits.conf`

Контроль использования ресурсов может быть очень эффективным при попытке предотвращения локальной атаки на отказ от обслуживания или ограничения максимального количества входов в систему для группы или пользователя. Однако излишне строгие настройки будут затормаживать работу вашей системы и станут причиной сбоев программ, поэтому проверьте каждый из параметров.

Листинг 1: `/etc/security/limits.conf`

```
*      soft core 0
*      hard core 0
*      hard nproc 15
*      hard rss 10000
*      -    maxlogins 2
@dev hard core 100000
@dev soft nproc 20
@dev hard nproc 35
@dev -    maxlogins 10
```

Если вы пытаетесь установить `nproc` или `maxlogins` в 0, то подумайте, может, лучше вместо этого удалить пользователя. В примере выше установлены настройки группы `dev` для процессов, основных файлов и `maxlogins`. Все остальное установлено в значения по умолчанию.

Примечание: `/etc/security/limits.conf` является частью пакета `PAM` и применима только для пакетов, использующих `PAM`.

5.b. `/etc/limits`

`/etc/limits` очень схож с файлом ограничений `/etc/security/limits.conf`. Единственное отличие — формат, который применим лишь для пользователей или заполнителей (но не для групп). Вот пример конфигурации:

Листинг 2: `/etc/limits`

```
*      L2 C0 U15 R10000
kn L10 C100000 U35
```

Здесь устанавливаются настройки по умолчанию и специфичные настройки для пользователя `kn`. `limits` являются частью пакета `sys-apps/shadow`. Если вы включили `pam` в `make.conf`, то нет необходимости устанавливать какие-либо ограничения в этом файле.

5.c. Квоты

Предупреждение: Проверьте, что рабочая файловая система поддерживает квоты. Чтобы задействовать квоты для `ReiserFS`, необходимо наложить заплатку на ядро, которая доступна на сайте [Namesys](http://www.namesys.com). Пользовательские утилиты можно

загрузить с сайта проекта [Linux DiskQuota project](http://www.linux-diskquota.org/). При совместной работе квот с ReiserFS вы можете встретить проблемы. Мы вас предупредили!

Установка квот на файловые разделы ограничивает использование дискового пространства для каждого пользователя или каждой группы. Квоты должны быть включены в ядре и добавлены к точке монтирования в /etc/fstab. Параметр ядра включается в конфигурации ядра в разделе File systems->Quota support. Установите следующие настройки, пересоберите ядро и перезагрузитесь в систему с новым ядром.

Начните с установки квот, запустив emerge quota. Затем измените /etc/fstab и добавьте usrquota и grpquota к разделам, использование которых вы хотите ограничить, так, как показано в примере ниже.

Листинг 3: /etc/fstab

```
/dev/sda1 /boot ext2 noauto,noatime 1 1
/dev/sda2 none swap sw 0 0
/dev/sda3 / reiserfs notail,noatime 0 0
/dev/sda4 /tmp ext3 noatime,nodev,nosuid,noexec,usrquota,grpquota 0 0
/dev/sda5 /var ext3 noatime,nodev,usrquota,grpquota 0 0
/dev/sda6 /home ext3 noatime,nodev,nosuid,usrquota,grpquota 0 0
/dev/sda7 /usr reiserfs notail,noatime,nodev,ro 0 0
/dev/cdroms/cdrom0 /mnt/cdrom iso9660 noauto,ro 0 0
proc /proc proc defaults 0 0
```

Создайте для каждого раздела, для которого включены квоты, файлы квот (aquota.user и aquota.group) и поместите их в корень каждого из разделов.

Листинг 4: Создание файлов квот

```
# touch /tmp/aquota.user
# touch /tmp/aquota.group
# chmod 600 /tmp/aquota.user
# chmod 600 /tmp/aquota.group
```

Это шаг выполняется для любого раздела, для которого включены квоты. После добавления и настройки файлов квот вам понадобится добавить сценарий запуска quota в загрузочный уровень запуска.

Важно: XFS проверяет все квоты самостоятельно, поэтому для нее не требуется добавления сценария quota в начальный уровень загрузки. Есть ряд файловых систем, не указанных в этом документе, которые функционируют также, поэтому обратитесь к странице руководства вашей файловой системы, для того, чтобы узнать, как она обрабатывает проверки квот.

Листинг 5: Добавление квот в загрузочный уровень исполнения

```
# rc-update add quota boot
```

Теперь настроим систему для проверки квот раз в неделю, добавив следующую строку в /etc/crontab:

Листинг 6: Добавление проверки квот в crontab

```
0 3 * * 0 /usr/sbin/quotacheck -avug.
```

Перезагрузив компьютер, время установить квоты для пользователей и групп. С помощью команды edquota -u kn запустится редактор, определенный переменной \$EDITOR (по умолчанию nano) для правки квот для пользователя kn. edquota -g

сделает то же самое для групп.

Листинг 7: Установка квот для пользователя kn

```
Quotas for user kn:
/dev/sda4: blocks in use: 2594, limits (soft = 5000, hard = 6500)
          inodes in use: 356, limits (soft = 1000, hard = 1500)
```

Для большей информации обратитесь к man edquota или [Quota miniHOWTO](#).

5.d. /etc/login.defs

Если правилами определено, что пользователи должны менять свои пароли каждые две недели, измените значения переменной PASS_MAX_DAYS на 14 и PASS_WARN_AGE на 7. Это рекомендуется для того, чтобы предотвратить старение пароля, так как полный перебор всех значений может взломать любой пароль, если есть достаточно времени. Также рекомендуется установить LOG_OK_LOGINS в значение yes.

5.e. /etc/login.access

Файл login.access также является частью пакета sys-apps/shadow, который управляет процессом входа в систему. Он определяет на основании имени пользователя, группы или узла, кто сможет войти в систему, а кто не сможет. Так как по умолчанию этот файл содержит лишь примеры и комментарии, то все пользователи системы могут войти в систему. В зависимости от того, что вы защищаете — сервер или рабочую станцию, рекомендуется настроить этот файл так, чтобы никто, кроме вас (как администратора), не мог получить доступ к консоли.

Примечание: Эти настройки не влияют на суперпользователя.

Листинг 8: /etc/login.access

```
 -:ALL EXCEPT wheel sync:console
 -:wheel:ALL EXCEPT LOCAL .gentoo.org
```

Важно: Настраивая эти параметры, будьте осторожны, так как любая ошибка может привести к тому, что вы не сможете войти в собственную систему, не воспользовавшись правами суперпользователя.

Примечание: Эти настройки не применимы для SSH, так как по умолчанию для SSH не запускается /bin/login. Это может быть включено с помощью параметра UseLogin yes в файле /etc/ssh/sshd_config.

Это устанавливает доступ на вход в систему так, что только члены группы wheel могут входить в систему локально или из домена gentoo.org. Возможно, попахивает паранойей, но лучше быть в безопасности, чем сожалеть о потерянном.

6. Права доступа к файлам

6.a. Файлы, доступные по чтению для всех

Обычные пользователи не должны иметь доступ к конфигурационным файлам или паролям. Атакующий может украсть пароли к базе данных или веб-сайту и

использовать их для дефейса или даже хуже — для удаления файлов. Вот почему важно правильно установить разрешения на файлы. Если вы уверены, что определенный файл может использоваться только суперпользователем, установите для него права 0600 и назначьте ему правильного владельца с помощью `chown`.

6.b. Файлы, доступные по записи для группы или для всех

Листинг 1: Поиск файлов, доступных по записи всем

```
# find / -type f \( -perm -2 -o -perm -20 \) -exec ls -lg {} \; 2>/dev/null
>writable.txt
# find / -type d \( -perm -2 -o -perm -20 \) -exec ls -ldg {} \; 2>/dev/null
>>writable.txt
```

Эти команды создадут огромный файл, содержащий имена файлов, которые имеют права на запись для группы и всех остальных. Проверьте их и устраните ненужные права, запустив для файлов команду `/bin/chmod o-w`.

6.c. Файлы SUID/SGID

Файлы с битами SUID или SGID будут запущены с привелегиями владельца или группы владельца, а не с привелегиями запустившего их пользователя. Обычно это используется для файлов, которые должны запускаться с правами суперпользователя для выполнения необходимых задач. Эти файлы, если содержат уязвимости, могут стать источником повышения привелегий локальным пользователем. Это может быть опасным, поэтому с подобных файлов необходимо любой ценой удалить установленный бит SUID или SGID. Если вы не используете эти файлы, запустите для них команду `chmod 0` или удалите пакет, установивший этот файл (проверить, какому пакету принадлежит определенный файл, можно с помощью команды `equegu`; если у вас ее еще нет, то, чтобы установить ее, просто наберите `emerge gentoolkit`). Иначе просто удалите бит SUID с помощью `chmod -s`.

Листинг 2: Поиск файлов с `setuid`

```
# find / -type f \( -perm -004000 -o -perm -002000 \) -exec ls -lg {} \; 2>/dev/
null >suidfiles.txt
```

Эта команда создаст файл, содержащий список всех SUID/SGID-файлов.

Листинг 3: Список файлов с `setuid`

```
/bin/su
/bin/ping
/bin/mount
/bin/umount
/var/qmail/bin/qmail-queue
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/crontab
/usr/bin/chage
/usr/bin/expiry
/usr/bin/sperl5.6.1
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/gpasswd
```

```

/usr/bin/procmail
/usr/bin/suidperl
/usr/lib/misc/pt_chown
/usr/sbin/unix_chkpwd
/usr/sbin/traceroute
/usr/sbin/pwdb_chkpwd

```

По умолчанию в Gentoo Linux не так много файлов с битом SUID (хотя это зависит от типа вашей установки), но у вас может получиться список, представленный выше. Большинство команд используется только суперпользователем. Удалите SUID-бит с `ping`, `mount`, `umount`, `chfn`, `chsh`, `newgrp`, `suidperl`, `pt_chown` и `traceroute`, запустив для каждого из них `chmod -s`. Не удаляйте бит с `su`, `qmail-queue` или `unix_chkpwd`. Удалив `setuid` с этих файлов, вы не сможете стать суперпользователем и получать почту. Удаляя бит (там, где это безопасно), вы исключаете возможность обычного пользователя (или злоумышленника) получить права суперпользователя с помощью этих файлов.

В моей системе есть только несколько SUID-файлов: `su`, `passwd`, `gpasswd`, `qmail-queue`, `unix_chkpwd` и `pwdb_chkpwd`. Но если у вас запущен X-сервер, то в вашей системе их будет больше, так как X нужны повышенные привелегии, которые могут быть предоставлены посредством SUID.

6.d. SUID/SGID-файлы и жесткие ссылки

Файл может считаться удаленным лишь в том случае, когда нет больше ссылок, указывающих на него. Это может звучать странно, но просто следует понять, что файл (например, `/usr/bin/perl`) на самом деле является ссылкой на `inode`, в котором сохранены данные. На файл может указывать любое число ссылок, и пока каждая из них не будет удалена, файл будет существовать.

Если у ваших пользователей есть доступ к разделам, которые не смонтированы с параметрами `nosuid` или `noexec` (например, если `/tmp`, `/home`, `/var/tmp` не являются отдельными разделами), вы должны удостовериться, что они не смогут создать жесткие ссылки на файлы с SUID- или SGID-битами, благодаря чему они могут иметь доступ к устаревшим версиям файлов.

Предупреждение: Если вы получаете предупреждения об оставшихся жестких ссылках, а ваши пользователи имеют доступ по записи к разделам, позволяющим запускать файлы с SUID/SGID-битами, вы должны прочитать этот раздел очень внимательно. Любой из ваших пользователей может разрушить ваши обновления, сохранив устаревшую версию программы. Если ваши пользователи не могут создавать собственные файлы с битом SUID или могут только запускать программы с помощью динамического загрузчика (разделы, смонтированные с параметром `noexec`), то вам не о чем беспокоиться.

Примечание: Пользователям не нужен доступ по чтению для файла, чтобы создать на него ссылку, им нужен всего лишь доступ по чтению к каталогу, содержащему этот файл.

Чтобы проверить, сколько ссылок имеет файл, вы можете использовать команду `stat`.

Листинг 4: Команда `stat`

```

$ stat /bin/su
  File: `/bin/su'
  Size: 29350          Blocks: 64          IO Block: 131072 regular file

```

```
Device: 900h/2304d      Inode: 2057419      Links: 1
Access: (4711/-rws--x--x)  Uid: (    0/    root)  Gid: (    0/    root)
Access: 2005-02-07 01:59:35.000000000 +0000
Modify: 2004-11-04 01:46:17.000000000 +0000
Change: 2004-11-04 01:46:17.000000000 +0000
```

Чтобы найти файлы с SUID и SGID со множеством ссылок, вы можете задействовать команду `find`.

Листинг 5: Поиск `suid/sgid`-файлов с несколькими ссылками

```
$ find / -type f \( -perm -004000 -o -perm -002000 \) -links +1 -ls
```

7. PAM

7.a. PAM

PAM — это набор библиотек, предоставляющих альтернативный способ аутентификации пользователя в программах. USE-флаг `pam` уже включен по умолчанию. Хотя настройки PAM в Gentoo Linux довольно разумны, всегда есть возможность что-нибудь улучшить. Для начала установите `cracklib`.

Листинг 1: Установка `cracklib`

```
# emerge cracklib
```

Листинг 2: `/etc/pam.d/passwd`

```
auth      required pam_unix.so shadow nullok
account   required pam_unix.so
password  required pam_cracklib.so difok=3 retry=3 minlen=8 dcredit=2
ocredit=2
password  required pam_unix.so md5 use_authtok
session   required pam_unix.so
```

После этого с помощью `cracklib` будет проверяться пароль пользователя, имеет ли он длину по крайней мере 8 символов, содержит ли он по крайней мере 2 цифры, 2 других символа и не менее 3 символов, отличающихся от символов прошлого пароля. Благодаря этому пользователь будет выбирать стойкие пароли. Проверьте [документацию PAM](#) для других параметров.

Листинг 3: `/etc/pam.d/sshd`

```
auth      required pam_unix.so nullok
auth      required pam_shells.so
auth      required pam_nologin.so
auth      required pam_env.so
account   required pam_unix.so
password  required pam_cracklib.so difok=3 retry=3 minlen=8 dcredit=2 ocredit=2
use_authtok
password  required pam_unix.so shadow md5
session   required pam_unix.so
session   required pam_limits.so
```

Любая служба, не описанная в одном из файлов PAM из каталога `/etc/pam.d`, будет использовать правила, взятые из `/etc/pam.d/other`. По умолчанию установлена политика `deny`, как и должно быть. Но нам нужна регистрация,

поэтому добавим `pam_warn.so`. Последняя конфигурация — это `pam_limits`, которая управляется с помощью `/etc/security/limits.conf`. См. раздел [/etc/security/limits.conf](#) для подробного описания этих настроек.

Листинг 4: `/etc/pam.d/other`

```
auth      required pam_deny.so
auth      required pam_warn.so
account   required pam_deny.so
account   required pam_warn.so
password  required pam_deny.so
password  required pam_warn.so
session   required pam_deny.so
session   required pam_warn.so
```

8. Упаковщики TCP

8.а. Упаковщики TCP

Обычно для контроля доступом к службам используется `inetd` (которого в Gentoo нет), но так же можно воспользоваться `xinetd` или другими сервисами.

Примечание: Служба должна быть запущена с помощью `tcpd` в качестве аргумента сервера (для случая `xinetd`). См. раздел о `xinetd` для большей информации.

Листинг 1: `/etc/hosts.deny`

```
ALL:PARANOID
```

Листинг 2: `/etc/hosts.allow`

```
ALL: LOCAL @wheel
time: LOCAL, .gentoo.org
```

Как вы видите, формат записи очень похож на содержимое файла `/etc/login.access`. `Tcpd` поддерживает отдельная служба; она никак не связана с `/etc/login.access`. Эти настройки применимы только для служб, использующих упаковщики TCP.

Также возможно запускать команды при разрешенной службе (это может быть полезным для создания трансляций для коммутируемых пользователей), но это не рекомендуется, так как обычно, решая одну проблему, люди создают еще больше. Допустим, что вы настроили сценарий, отправляющий электронное письмо в том случае, когда кто-то подпал под запрещающее правило, но тогда злоумышленник может запустить DoS-атаку, попадающую под правило запрещения. Из-за этого создастся много ввода-вывода и ненужно электронной почты, поэтому не делайте так! Прочтите `man 5 hosts_access` для дальнейшей информации.

9. Безопасность ядра

9.а. Удаление функциональности

Основное правило при конфигурации ядра — удалять все ненужное. Это не

только создаст маленькое ядро, но и также исключит вероятные уязвимости, которые могут присутствовать в драйверах и других возможностях.

Также рекомендуется отключить поддержку загружаемых модулей. Хотя возможность внедрения руткитов сохраняется, это делает систему более неприступной для обычного злоумышленника, пытающегося установить руткит в виде модуля ядра.

9.b. Файловая система *proc*

Множество параметров ядра может быть изменено с помощью файловой системы */proc* или *sysctl*.

Чтобы иметь возможность на лету изменять параметры и переменные ядра, вам понадобится определить в ядре *CONFIG_SYSCTL*. Она включена по умолчанию в ядрах серии 2.4.

Листинг 1: Отключения IP-форвардинга

```
# /bin/echo "0" > /proc/sys/net/ipv4/ip_forward
```

Проверьте, что проброс IP-пакетов отключен. Эта возможность необходима лишь для узла, имеющего доступ к нескольким сетям. Рекомендуется устанавливать и отключать этот флаг до включения или отключения других флагов.

Листинг 2: Отбрасывание пакетов ping

```
# /bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Это просто заставит ядро игнорировать все сообщения ping (ICMP-пакеты типа 0). IP-пакет, несущий ICMP-сообщение, может содержать также в нагрузку и другую информацию, о которой вы можете не подозревать, поэтому следует отключить прием. Администраторы используют ping как утилиту диагностики и часто выражают недовольство, если она отключена, но нет причины позволять чужакам пинговать узел. Тем не менее, если необходимо разрешить внутренним пользователям использовать ping, то можно отключить сообщения ICMP типа 0 в межсетевом экране (тем самым позволив локальным администраторам использовать эту утилиту).

Листинг 3: Игнорирование широковещательных запросов ping

```
# /bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
```

Это отключает ответ на широковещательные ICMP-запросы, предотвращая реализацию Smurf-атаки. Smurf-атака основана на отправке ICMP-пакета типа 0 (ping) по широковещательному адресу сети. Обычно для этого злоумышленник использует поддельный адрес. Все компьютеры сети ответят на сообщение ping, что может вызвать наводнение трафика на узел, адрес которого был подделан.

Листинг 4: Отключение исходящих маршрутизированных пакетов.

```
# /bin/echo "0" > /proc/sys/net/ipv4/conf/all/accept_source_route
```

Не принимать исходящие маршрутизированные пакеты. Злоумышленники могут использовать исходящую маршрутизацию для генерации трафика, имеющего адрес внутренней сети, но на самом деле возвращаемого назад злоумышленнику, что позволит ему компрометировать сеть. Исходящая маршрутизация очень редко используется по назначению, так что безопаснее ее отключить.

Листинг 5: Отключение приема перенаправлений

```
# /bin/echo "0" > /proc/sys/net/ipv4/conf/all/accept_redirects
# /bin/echo "0" > /proc/sys/net/ipv4/conf/all/secure_redirects
```

Не принимать ICMP-пакеты перенаправления. ICMP-перенаправления могут быть использованы злоумышленником для изменения таблиц маршрутизации.

Листинг 6: Защита против неправильных сообщений об ошибках

```
# /bin/echo "1" > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
```

Включить защиту против приходящих неправильных сообщений об ошибках.

Листинг 7: Включение фильтрации обратного пути

```
# for i in /proc/sys/net/ipv4/conf/*; do
    /bin/echo "1" > $i/rp_filter
done
```

Включает фильтрацию обратного пути. Это поможет быть уверенным, что в пакетах используются правомерные адреса, при этом входящие пакеты будут автоматически отброшены, если запись в таблице маршрутизации об адресе этих пакетов не соответствуют сетевому интерфейсу, с которого они пришли. Это позволит предотвратить подделку IP-адресов. Необходимо включить эту возможность для каждого net/ipv4/conf/*, иначе проверка исходящих пакетов не будет полностью функционировать.

Предупреждение: Тем не менее включение фильтрации обратного пути может стать проблемой при использовании асимметричной маршрутизации (исходящие пакеты идут отличным от входящих пакетов путем) или при использовании немаршрутизируемом узле с несколькими IP-адресами на различных интерфейсах.

Листинг 8: Регистрация всех поддельных, исходящих маршрутизированных и перенаправленных пакетов

```
# /bin/echo "1" > /proc/sys/net/ipv4/conf/all/log_martians
```

Регистрировать поддельные пакеты, пакеты с исходящей маршрутизацией и пакеты перенаправлений.

Все настройки будут сброшены после перезагрузки компьютера. Рекомендуется добавлять их в /etc/sysctl.conf, который будет автоматически загружен сценарием инициализации /etc/init.d/bootmisc.

Синтаксис /etc/sysctl.conf достаточно понятный. Удалите из указанных путей /proc/sys/ и замените / на .:

Листинг 9: Переход к файлу sysctl.conf

```
(Вручную с помощью команды echo:)
/bin/echo "0" > /proc/sys/net/ipv4/ip_forward
```

```
(Автоматически в файле sysctl.conf:)
net.ipv4.ip_forward = 0
```

9.c. Grsecurity

Заплата от [Grsecurity](#) входит в sys-kernel/hardened-sources, но по умолчанию отключена. Сначала сконфигурируйте ядро, а затем настройте параметры Grsecurity. Подробное описание доступных параметров Grsecurity можно найти на странице проекта [укрепленного Gentoo](#).

Последние версии hardened-sources содержат Grsecurity версии 2.*. Для дополнительной информации о наборе патчей Grsecurity обратитесь к документации, доступной на [домашнем сайте Grsecurity](#).

9.d. Kerneli

[Kerneli](#) — это заплата, которая добавляет функции шифрования к существующему ядру. Наложив эту зачатку на ядро, вы получите такие новые параметры, как алгоритмы шифрования, дайджеста и фильтры к зашифрованным образам.

Предупреждение: В настоящий момент заплата kerneli для последнего ядра нестабильна, поэтому будьте осторожны при ее использовании.

9.e. Другие заплаты к ядру

- [Проект OpenWall](#)
- [Linux Intrusion Detection System](#)
- [Rule Set Based Access Control](#)
- [NSA's security enhanced kernel](#)
- [Wolk](#)

И множество других.

10. Безопасность служб

10.a. Apache

Apache поставляется с достаточно полным конфигурационным файлом по умолчанию, но опять же необходимо улучшить некоторые аспекты, например, использовать для Apache только один адрес. Ниже приведены параметры, которые вы должны внести в конфигурационный файл.

Если вы не отключали поддержку ssl в /etc/make.conf перед установкой Apache, то теперь вы должны иметь доступ к серверу с включенным SSL. Чтобы включить ее, просто добавьте следующую строку.

Листинг 1: /etc/conf.d/apache

```
HTTPD_OPTS="-D SSL"
```

Листинг 2: /etc/apache/conf/apache.conf

```
#Добавьте свой IP для прослушивания
Listen 127.0.0.1
BindAddress 127.0.0.1
#Не следует использовать nobody или nogroup для каждого сервиса, запущенного
#без административных привилегий
#(просто добавляем пользователя apache с группой apache)
User apache
```

```
Group apache
#Не сообщать информацию о версии сервера
ServerSignature Off
ServerTokens Prod
```

Apache собран с параметрами `--enable-shared=max` и `--enable-module=all`. По умолчанию это включает все модули, поэтому вы должны закомментировать все неиспользуемые модули в разделе `LoadModule` (`LoadModule` и `AddModule`). Затем перезапустите сервис, выполнив `/etc/init.d/apache restart`.

Документация доступна по адресу <http://www.apache.org>.

10.b. Bind

Документацию можно найти на сайте [Internet Software Consortium](http://www.internetsoftwareconsortium.com). Руководство администратора BIND 9 также доступно каталоге `doc/arm`.

Новые сборочные файлы BIND поддерживают изменение корневого каталога из коробки. После установки `bind` следуйте этим простым инструкциям:

Листинг 3: Изолирование среды BIND

```
# emerge --config bind
(Перед запуском следующей команды вам может потребоваться изменить каталог
chroot в файле /etc/conf.d/named. Иначе будет использован /chroot/dns.)
```

10.c. Djbdns

Djbdns — это реализация DNS, за обнаружение уязвимостей в которой автор готов выплачивать [деньги](#). Принцип работы сильно отличается от Bind 9, однако он работает. Дополнительная информация может быть получена на сайте <http://www.djbdns.org>.

10.d. FTP

В общем случае использование FTP (File Transfer Protocol, протокол передачи файлов) является плохой идеей. Этот протокол отправляет данные незашифрованными (в том числе и пароли), прослушивает 2 порта (обычно это порты 20 и 21), и часто злоумышленники ищут анонимный доступ для обмена вarezом. Так как протокол содержит ряд проблем безопасности, то следует использовать `sftp` или `HTTP`. Если это невозможно, то максимально обезопасьте свои сервисы и будьте готовы.

10.e. Mysql

Если вам необходимо предоставить доступ к базе данных `mysql` локальным приложениям, раскомментируйте следующую строку в `/etc/mysql/my.cnf`.

Листинг 4: Отключение доступа к сети

```
skip-networking
```

Затем мы отключаем использование команды `LOAD DATA LOCAL INFILE`. Это предотвратит несанкционированное чтение локальных файлов. Это также подходит против SQL-инъекций в PHP-сценариях.

Листинг 5: Отключение `LOAD DATA LOCAL INFILE` в разделе `[mysql]`

```
set-variable=local-infile=0
```

Затем необходимо удалить тестовую базу данных (test) и все учетные записи за исключением root.

Листинг 6: Удаление тестовой базы данных и всех ненужных пользователей

```
mysql> drop database test;
mysql> use mysql;
mysql> delete from db;
mysql> delete from user where not (host="localhost" and user="root");
mysql> flush privileges;
```

Предупреждение: Если вас уже есть настроенные учетные записи, то будьте осторожны с этими командами.

Примечание: Если вы изменяли пароли в командной строке MySQL, то всегда очищайте ~/.mysql_history и /var/log/mysql/mysql.log, так как они сохраняют список выполненных SQL-команд с открытыми паролями.

10.f. Proftpd

У proftpd было несколько проблем с безопасностью, но большинство из них было исправлено. Тем не менее, неплохо внести некоторые улучшения:

Листинг 7: /etc/proftpd/proftpd.conf

```
ServerName "My ftp daemon"
# Не показывать ident сервера
ServerIdent on "Go away"

# Упрощаем создание виртуальных пользователей
RequireValidShell off

# Использовать альтернативные файлы паролей и групп (в формате passwd)
AuthUserFile "/etc/proftpd/passwd"
AuthGroupFile "/etc/proftpd/group"

# Разрешения
Umask 077

# Таймауты и ограничения
MaxInstances 30
MaxClients 10 "Only 10 connections allowed"
MaxClientsPerHost 1 "You have already logged on once"
MaxClientsPerUser 1 "You have already logged on once"
TimeoutStalled 10
TimeoutNoTransfer 20
TimeoutLogin 20

# Все входят в изолированную оболочку
DefaultRoot ~

# Не запускать с правами администратора
User nobody
Group nogroup

# Регистрировать любые передачи
```

```
TransferLog /var/log/transferlog

# Проблемы с универсализацией имен файлов
DenyFilter \*.*/*
```

Дополнительная информация может быть найдена на <http://www.proftpd.org>.

10.g. Pure-ftpd

Pure-ftpd является ответвлением оригинального trollftpd с модификациями Фрэнка Денниса по части безопасности и функциональности.

Используйте виртуальных пользователей (и никогда не применяйте системные учетные записи), включив параметр AUTH. Установите его для `-lpuredb:/etc/pureftpd.pdb` и создайте пользователей при помощи команды `/usr/bin/pure-pw`.

Листинг 8: /etc/conf.d/pure-ftpd

```
AUTH="-lpuredb:/etc/pureftpd.pdb"

## Misc. Others ##
MISC_OTHER="-A -E -X -U 177:077 -d -4 -L100:5 -I 15"
```

Добавьте к переменной MISC_OTHER параметры запрета на подключение анонимных пользователей (-E), изменения корневого каталога для всех (-A), запрета записи и чтения файлов, начинающихся с точки (-X), максимального периода бездействия (-I), ограничения рекурсии (-L) и подходящего umask.

Предупреждение: Не используйте параметры `-w` или `-W`! Если же вы хотите, чтобы у вас был сайт с вarezом, то перестаньте читать это руководство!

Дополнительная информация может быть найдена на <http://www.pureftpd.org>.

10.h. Vsftpd

Vsftpd (сокращение от very secure ftp) — это небольшой FTP-демон, который может запускаться с настройками по умолчанию. Он прост и не имеет множества возможностей, присущих pureftpd и proftpd.

Листинг 9: /etc/vsftpd

```
anonymous_enable=NO
local_enable=YES

#read only
write_enable=NO

#enable logging of transfers
xferlog_std_format=YES

idle_session_timeout=20
data_connection_timeout=20
nopriv_user=nobody

chroot_list_enable=YES
chroot_list_file=/etc/vsftpd/chrootlist
```

```
ls_recurse_enable=NO
```

Как вы можете видеть, нет возможности установить для данного сервиса индивидуальные разрешения, но если используется настройки для анонимного доступа, то это он не так уж плох. Иногда неплохо иметь анонимный FTP-сервер (для доступа к открытому ПО), и vsftpd неплохо подходит для этой работы.

10.i. Netqmail

Netqmail часто признается наиболее безопасным почтовым сервером. Он написан с прицелом на безопасность (и паранойю). По умолчанию он не разрешает релей и, начиная с 1996 года, не имеет никаких уязвимостей безопасности. Просто наберите `emerge netqmail` и настройте его!

10.j. Samba

Samba — это протокол, предоставляющий файлы в сетях Microsoft/Novell, и он не должен использоваться в интернете. Его необходимо обезопасить.

Листинг 10: /etc/samba/smb.conf

```
[global]
#Прослушивать определенный интерфейс
interfaces = eth0 10.0.0.1/32

#Использовать зашифрованные пароли
encrypt passwords = yes
directory security mask = 0700

#Разрешить трафик из подсети 10.0.0.*
hosts allow = 10.0.0.

#Включить аутентификацию
#(не используйте режим share)
security = user

#Запретить привилегированные учетные записи
invalid users = root @wheel

#Максимальный отображаемый размер ресурса (не является ограничителем)
max disk size = 102400

#Ужесточить политики паролей
min password length = 8
null passwords = no

#Использовать PAM (если была добавлена поддержка)
obey pam restrictions = yes
pam password change = yes
```

Проверьте, что для каждого ресурса выставлены правильные разрешения и не забудьте прочитать [документацию](#).

Теперь перезапустите сервер и добавьте пользователей, которым необходим доступ к этому сервису. Это можно сделать с помощью команды `/usr/bin/smbpasswd` с параметром `-a`.

10.k. ssh

Для OpenSSH необходимо лишь одно улучшение — использование более надежного механизма аутентификации, основанного на шифровании с использованием открытого ключа. Очень много сайтов (например, <http://www.sourceforge.net>, <http://www.php.net> и <http://www.apache.org>) было подвержено несанкционированному доступу из-за пустых или слабых паролей.

Листинг 11: /etc/ssh/sshd_config

```
#Включаем только версию 2
Protocol 2

#Отключаем вход администратора. Пользователи могут использовать su для входа
PermitRootLogin no

#Включаем аутентификацию по открытому ключу
PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys

#Отключаем .rhost и обычную аутентификацию по паролю
HostbasedAuthentication no
PasswordAuthentication no
PermitEmptyPasswords no

#Разрешаем подключаться только пользователям из групп wheel или admin
AllowGroups wheel admin

#Из этих групп разрешаем подключаться только следующим пользователям
#@<domainname> не обязательна, но заменяет старую директиву
#AllowHosts
AllowUsers kn@gentoo.org bs@gentoo.org

#Вход в систему
SyslogFacility AUTH
LogLevel INFO

(Измените адрес на свой)
ListenAddress 127.0.0.1
```

Также проверьте, что в конфигурационном файле нет параметра UsePAM yes, который перекрывает механизм аутентификации с помощью открытого ключа.

Теперь создайте для каждого пользователя ключ (на компьютере, с которого они будут регистрироваться) с помощью следующей команды:

Листинг 12: Создание пар ключей DSA

```
# /usr/bin/ssh-keygen -t dsa
```

И введите пароль.

Листинг 13: Вывод ssh-keygen

```
Generating public/private dsa key pair.
Enter file in which to save the key (/home/kn/.ssh/id_dsa):[Press enter]
Created directory '/home/kn/.ssh'.
Enter passphrase (empty for no passphrase): [Enter passphrase]
Enter same passphrase again: [Enter passphrase again]
Your identification has been saved in /home/kn/.ssh/id_dsa.
Your public key has been saved in /home/kn/.ssh/id_dsa.pub.
```

```
The key fingerprint is:
07:24:a9:12:7f:83:7e:af:b8:1f:89:a3:48:29:e2:a4 kn@knielsen
```

После этого в каталоге `~/.ssh/` появятся два файла с названием `id_dsa` и `id_dsa.pub`. Файл `id_dsa` является секретным ключом и должен оберегаться от других. Другой файл, `id_dsa.pub`, должен быть размещен на каждом сервере, к которому необходим доступ. Добавьте этот ключ в каталог `~/.ssh/authorized_keys` в домашнем каталоге пользователя. Теперь пользователь должен иметь доступ к подключению:

Листинг 14: Adding the id_dsa.pub file to the authorized_keys file

```
$ scp id_dsa.pub other-host:/var/tmp/currenthostname.pub
$ ssh other-host
password:
$ cat /var/tmp/currenthostname.pub >> ~/.ssh/authorized_keys
```

Теперь ваши пользователи должны тщательно оберегать свои секретные ключи. Поместите их на сменный носитель, который они будут всегда носить с собой, или спрячьте их на рабочих станциях пользователей (поместите их под правила [паролей](#)).

Для дальнейшей информации посетите веб-сайт [OpenSSH](#).

10.1. Использование xinetd

`xinetd` является заменой `inetd` (которого в Gentoo нет), демона сетевых сервисов. Он поддерживает контроль доступа, основанный на адресе удаленного узла и времени доступа. Он также предоставляет расширенные возможности регистрации событий, включая время запуска сервера, адрес удаленного узла, имя удаленного пользователя, время работы сервера и выполненные запросы.

Как в случае с другими сервисами, важно создать хорошую конфигурацию по умолчанию. Но так как `xinetd` запускается с правами администратора и поддерживает протоколы, о работе которых вы можете не знать, рекомендуется не использовать его. Но если вы все равно хотите использовать его, то вы можете укрепить его безопасность:

Листинг 15: Установка xinetd

```
# emerge xinetd tcp-wrappers
```

И отредактируйте конфигурационный файл:

Листинг 16: /etc/xinetd.conf

```
defaults
{
    only_from = localhost
    instances = 10
    log_type = SYSLOG authpriv info
    log_on_success = HOST PID
    log_on_failure = HOST
    cps = 25 30
}

# Это установит работу pserver (cvs) через xinetd со следующими:
# настройками:
# максимум 10 обработчиков (10 подключений одновременно)
```



```
# использовать только TCP
# использовать пользователя cvs для запуска сервиса
# использовать только 1 IP
# разрешать доступ с 10.0.0.*
# разработчики могут пользоваться cvs только с 8 до 17 часов
# использовать упаковщики tcpd (контроль доступа через
# /etc/hosts.allow и /etc/hosts.deny)
# max_load для компьютера 1.0
# Отключающий флаг для настроек по умолчанию, я предпочитаю,
# чтобы он был отключен
service cvspserver
{
    socket_type = stream
    protocol = tcp
    instances = 10
    protocol = tcp
    wait = no
    user = cvs
    bind = 10.0.0.2
    only_from = 10.0.0.0
    access_times = 8:00-17:00
    server = /usr/sbin/tcpd
    server_args = /usr/bin/cvs --allow-root=/mnt/cvsdisk/cvsroot pserver
    max_load = 1.0
    log_on_failure += RECORD
    disable = no
}
```

Для большей информации см. `man 5 xinetd.conf`.

10.m. X

По умолчанию Xorg настроен функционировать в качестве Xserver. Это может быть опасным, так как X использует не зашифрованные TCP-соединения и прослушивает подключения `xclient`.

Важно: Если вам не нужен этот сервис, отключите его!

Но если вам требуется использовать рабочую станцию в качестве X-сервера, используйте команду `/usr/X11R6/bin/xhost` с большой осторожностью. Эта команда позволит клиентам других узлов подключаться к вашему экрану. Это может быть полезно при необходимости запуска приложения X с другого компьютера, и это возможно лишь через сеть, но это может быть использовано и злоумышленником. Синтаксис этой команды — `/usr/X11R6/bin/xhost +hostname`.

Предупреждение: Не используйте возможность `xhost +!` Эта возможность позволит любому клиенту подключиться к X-серверу и взять контроль над ним. Если злоумышленник может получить доступ к X, то сможет регистрировать нажатия на клавиши, что позволит ему взять контроль над вашим компьютером. Если вы все же используете ее, то всегда указывайте узел.

Более безопасным решением является полное отключение этой возможности при старте X-сервера с помощью `startx -- -nolisten tcp` или отключение ее навсегда в файлах конфигурации.

Листинг 17: `/usr/X11R6/bin/startx`

```
defaultserverargs="-nolisten tcp"
```

Вы должны защитить startx, чтобы не допустить его перезаписи при установке новой версии Xorg. Добавьте следующую строку в /etc/make.conf:

Листинг 18: /etc/make.conf

```
CONFIG_PROTECT_MASK="/usr/X11R6/bin/startx"
```

Если вы используете графический диспетчер входа в систему, вам понадобится другой подход.

Для gdm (Gnome Display Manager)

Листинг 19: /etc/X11/gdm/gdm.conf

```
[server-Standard]
command=/usr/X11R6/bin/X -nolisten tcp
```

Для xdm (X Display Manager) и kdm (Kde Display Manager)

Листинг 20: /etc/X11/xdm/Xservers

```
:0 local /usr/bin/X11/X -nolisten tcp
```

11. Изменение корневого каталога и виртуальные серверы

11.а. Изменение корневого каталога

Изменение корневого каталога для службы является способом ограничения среды службы (или пользователя), в которой она имеет доступ лишь к необходимым ресурсам. Запущенная служба под пользователем, отличным от суперпользователя (nobody, apache, named) может предоставить злоумышленнику доступ лишь к тем файлам, к которым имеет доступ пользователь, от имени которого запущена служба. Это означает, что злоумышленник не сможет получить права root, даже если служба подвержена различным уязвимостям.

Некоторые службы, например pure-ftpd и bind, могут быть заключены в chroot. Если служба поддерживает эту возможность, используйте ее, иначе вы можете создать среду собственноручно. Давайте рассмотрим пример создания chroot. Чтобы изучить работу механизма chroot, мы будем экспериментировать с bash (простейший для изучения случай).

Создайте каталог /chroot, выполнив команду mkdir /chroot. Затем определите, какие динамические библиотеки необходимы для работы bash (если он собран с параметром -static, этот шаг можно пропустить):

Следующая команда создаст список библиотек, необходимых для bash.

Листинг 1: Получение списка используемых библиотек

```
# ldd /bin/bash
libncurses.so.5 => /lib/libncurses.so.5 (0x4001b000)
libdl.so.2 => /lib/libdl.so.2 (0x40060000)
libc.so.6 => /lib/libc.so.6 (0x40063000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Теперь создадим изолированную среду для bash.

Листинг 2: Создание изолированной среды для bash

```
# mkdir /chroot/bash
```

```
# mkdir /chroot/bash/bin
# mkdir /chroot/bash/lib
```

Затем скопируем файлы, используемые bash (/lib) в изолированный lib и скопируем файл bash в изолированный каталог bin. Этим мы создадим ту же самую среду, но с ограниченными возможностями. После этого попробуйте выполнить команду `chroot /chroot/bash /bin/bash`. Если вы получите строку приглашения, гласящую `/`, то у вас все получилось! Иначе вам сообщат, какого файла не хватает. Некоторые разделяемые библиотеки могут использовать другие файлы.

Как вы можете заметить, внутри изолированной среды ничего, кроме `echo`, не работает. Это потому что внутри среды `chroot` кроме `bash` нет никаких других команд, а `echo` является встроенной командой.

Подобным образом вы можете создать службу в изолированной среде. Единственное различие в том, что служба может время от времени обращаться к устройствам и файлам настроек в `/etc`. Просто скопируйте их (файлы устройств можно скопировать с помощью команды `cp -a`) в изолированную среду, отредактируйте сценарий инициализации перед тем, как заключить службу в среду. Может быть сложно определить, какие устройства и файлы конфигурации могут понадобиться. Здесь может пригодиться команда `strace`. Запустите службу вместе с `/usr/bin/strace` и отслеживайте все вызовы `open`, `read`, `stat` и, возможно, `connect`. Это подскажет вам, что нужно копировать. Но в большинстве случаев вам понадобится скопировать файл `passwd` (отредактируйте копию, удалив из него всех пользователей, не нужных для запуска службы), `/dev/zero`, `/dev/log` и `/dev/random`.

11.b. Пользовательский режим Linux

Другим способом создания безопасной среды является запуск виртуальной машины. Виртуальная машина — это процесс, выглядящий как отдельная ОС и работающий в реальной операционной системе, которая предоставляет ему необходимые системные ресурсы. Безопасность достигается в том, что если сервер, запущенный внутри виртуальной машины, будет взломан, то будет затронут только виртуальный сервер, а не родительская установка.

Для дальнейшей информации по установке пользовательского режима Linux, обратитесь к [руководству по пользовательскому режиму Linux](#).

12. Межсетевые экраны

12.a. Межсетевой экран

Люди часто думают, что межсетевой экран окончательно защитит их систему, но они ошибаются. В большинстве случаев плохо настроенный межсетевой экран предоставляет еще меньше безопасен, чем его отсутствие. Он тоже является программой, и с ним нужно обращаться так же, как и с остальными программами, так как он может содержать уязвимости.

А теперь подумайте перед реализацией межсетевого экрана! Так ли он вам нужен? Если да, то вам нужно написать правила, определяющие его работу, тип межсетевого экрана и кто должен им управлять. Но сначала прочитайте это

руководство.

Межсетевые экраны используются для двух целей:

- Для защиты извне (черви/злоумышленники)
- Для защиты изнутри (сотрудники/дети)

В основном есть три типа межсетевых экранов:

- Фильтрация пакетов
- Прокси
- Программный шлюз

Межсетевой экран должен работать на выделенном компьютере без лишних запущенных сервисов (или же только с sshd) и обезопасен такими способами, которые описаны в этом руководстве.

12.b. Фильтрация пакетов

Весь сетевой трафик передается в виде пакетов. Непрерывный поток разделяется на маленькие, легко управляемые пакеты, которые собираются в точке назначения. В заголовке каждого пакета содержится информация о способе и месте его назначения. Также эта информация используется работы меж сетевого экрана. Фильтрация основана на:

- разрешении или запрещении пакетов на основании IP-адреса отправителя/получателя
- разрешении или запрещении пакетов на основании порта отправителя/получателя
- разрешении или запрещении пакетов на основании протокола
- разрешении или запрещении пакетов на основании параметров, специфичных для каждого из протоколов

Другими словами, фильтрация основана на содержимом заголовка пакета, а не самого пакета.

Недостатки:

- адрес в пакете может быть ненастоящим адресом IP (или подделанным отправителем)
- данные или запросы в пропущенном пакете могут содержать непредвиденные данные, которые злоумышленник может использовать для эксплуатации известных уязвимостей в сервисах, расположенным на межсетевом экране или позади него
- критично к сбоям

Преимущества:

- простая реализация
- возможность отправки предупреждений о возможных атаках до того, как они произойдут (то есть регистрация сканирования портов)
- хорош против предотвращения SYN-атак

Вот примеры свободных фильтров пакетов для Linux:

- [Iptables](#)
- [Ipchains](#)
- [SmoothWall](#)

Примечание: Рекомендуется использовать iptables. ipchains устарел.

12.c. Прокси

Шлюз сеансового уровня является межсетевым экраном, проверяющим соединения перед разрешением обмена данными. Это значит, что он не только разрешает или запрещает пересылку пакетов на основании их заголовка, но и определяет перед открытием сессии на основании настраиваемых правил, что оба адресата реально существуют. Фильтрация основывается на:

- IP-адресе отправителя/получателя
- порте отправителя/получателя
- времени
- протоколе
- пользователе
- пароле

Весь трафик прослушивается и проверяется, и непрошенный трафик может быть отброшен.

Недостатки:

- Взаимодействует на транспортном уровне и может потребовать изменения программ, предоставляющих транспортные функции.

12.d. Программный шлюз

Шлюзы уровня приложений являются прокси-серверами для приложений, обменивающимися вместо клиентов данными с удаленными системами. Это позволяет находиться вне доступа извне за ДМЗ (демилитаризованной зоной, частью частной сети, видимой сквозь межсетевой экран) или разрешать межсетевому экрану предотвращать соединения извне. Фильтрация основывается на:

- разрешении или запрещении на основании IP-адреса источника/назначения
- на содержимом пакетов
- ограничения доступа к файлу на основе типа файла и расширения

Преимущества:

- Может кэшировать файлы, увеличивая производительность сети
- Детализированная регистрация всех подключений
- Хорошая расширяемость (некоторые прокси могут распределять кэшированные данные между собой)
- Нет прямого доступа извне
- Может изменять содержимое пакета на лету

Недостатки:

- Конфигурация является комплексной

Программные шлюзы принято считать наиболее безопасным решением, так как они не запускаются с правами администратора, и все узлы за ними будут недоступными из интернета.

Пример свободного шлюза:

- [Squid](#)

12.e. Iptables

Для нормального функционирования iptables должен быть включен в ядро. Я

включил поддержку iptables модулей (команда iptables загрузит их по мере необходимости) и пересобрал ядро (но вы можете включить их в само ядро, если намереваетесь отказаться от загружаемых модулей ядра, как сказано выше). Для более детальной информации по настройке iptables в ядре обратитесь к странице [iptables Tutorial Chapter 5: Preparations](#). После пересборки нового ядра (или во время компиляции) вы должны добавить команду iptables. Для этого просто наберите `emerge iptables`.

Теперь проверим свою работу, запустив `iptables -L`. Если команда завершилась ошибкой, значит, что-то не так, поэтому проверьте еще раз настройки.

iptables — новый и весьма улучшенный межсетевой экран в ядрах Linux 2.4.x. Он является наследником ipchains для ядер Linux 2.2.x. Одним из значительных нововведений является то, что iptables способен совершать полноценную фильтрацию пакетов. С ее помощью стало возможным следить за каждым установленным TCP-соединением.

TCP-соединение содержит в себе серию пакетов, содержащих информацию IP-адресе и порте отправителя и получателя, а также последовательное число, поэтому пакеты могут быть воссозданы без потери данных. TCP является протоколом, ориентированном на подключение, в отличие от UDP, который не гарантирует доставку.

Изучая заголовок TCP-пакета, межсетевой экран может определить, является ли полученный пакет частью уже установленного соединения или нет, и принять решение, принять или отбросить этот пакет.

При использовании межсетевого экрана, не отслеживающего соединения, возможно провести его и заставить с помощью манипулирования заголовками TCP-пакета принимать пакеты, которые необходимо отбросить. Это может быть сделано с помощью установки SYN или других флагов заголовка TCP и создания поддельного пакета, который будет считаться частью уже установленного соединения (ведь межсетевой экран не будет отслеживать состояние соединения). При использовании фильтрации пакетов, отслеживающих соединения, можно отбрасывать подобные пакеты, если они не являются частью уже установленного соединения. Это также предотвратит «stealth-сканирование», разновидности сканирования, с помощью которого сканер отправляет пакеты с такой комбинацией флагов, при которых они не будут зарегистрированы межсетевым экраном, полагаящим, что это — обычные SYN-пакеты.

В iptables также есть различные возможности, как например NAT (Network Address Translation, сетевая трансляция адресов) и ограничения по частоте. Ограничение по частоте весьма полезно для предотвращения DoS-атак (Denial of Service, отказ от обслуживания), например SYN-наводнений.

TCP-соединения устанавливаются после так называемого троекратного рукопожатия. При установлении TCP-соединения клиент отправляет серверу пакет с установленным флагом SYN. При получении сервер возвращает клиенту пакет с установленными SYN+ACK. При его получении клиент отвечает третьим пакетом с ACK для подтверждения соединения.

SYN-наводнения основаны на отправке SYN-пакетов с одновременным запретом отправки пакетов SYN+ACK. Клиент может создать пакет с поддельным IP-адресом отправителя, так как ему не нужно на что-либо отвечать. Сервер заполнит очередь подключений полуоткрытыми соединениями, ожидающими окончательного пакета с ACK, до того, как удалит их из очереди. Очередь

ограничена определенным числом, и когда она заполнится, то станет невозможным принимать новые подключения. Если пакет с ACK не будет получен по истечении определенного временного интервала, то он будет автоматически удален из очереди. Настройки таймаута могут быть различными, но обычно они находятся в пределах 30—60 секунд или даже больше. Клиент инициирует атаку, создавая множество SYN-пакетов с различных IP-адресов и отправляя их на адрес цели как можно чаще, заполняя тем самым очередь полуоткрытыми соединениями и не позволяя другим клиентам устанавливать легитимные подключения к серверу.

В данном случае может быть полезным ограничение отправки. Можно ограничить частоту отправки принятых SYN-пакетов с помощью `-m limit --limit 1/s`, тем самым ограничив число SYN-пакетов до одного в секунду и оградив свои ресурсы от SYN-наводнений.

Примечание: Другим решением, предотвращающим SYN-наводнения является использование [SYN cookies](#), которые позволяют вашему компьютеру отвечать на SYN-пакеты без заполнения очереди подключений. SYN cookies могут быть включены при конфигурировании ядра Linux, однако на данный момент эта поддержка считается экспериментальной.

А теперь немного практических занятий!

Будучи загруженным в ядро, iptables предоставляет 5 ловушек, в которые вы можете поместить свои правила. Они называются INPUT, OUTPUT, FORWARD, PREROUTING и POSTROUTING. Каждая из них называется цепочкой и содержит список правил. Каждое правило описывает, что если заголовок пакета выглядит как образец, то делать с этим пакетом нужно то-то и то-то. Если правило не подходит под пакет, то пакет переходит к следующему правилу в цепочке.

Вы можете поместить правила в 5 цепочках напрямую или создать новые цепочки и добавлять в них правила таким же способом, как и в обычную цепочку. Iptables поддерживает следующие параметры.

Параметр:	Описание:
-A	Добавление
-D	Удаление
-I	Вставка
-R	Замена
-L	Просмотр
-F	Удалить все правила в цепочке или все цепочки
-Z	Обнулить счетчики во всех цепочках
-C	Проверить этот пакет в цепочке
-N	Создать новую пользовательскую цепочку
-X	Удалить пользовательскую цепочку
-P	Изменить политику цепочки для цели
-E	Изменить имя цепочки
-p	протоколе

-s	Адрес/маска источника
-d	Адрес/маска назначения
-i	Входящее имя (имя ethernet)
-o	Исходящее имя (имя ethernet)
-j	Перейти (цель для правила)
-m	Расширенные сравнения (могут использовать внешние расширения)
-n	Числовой вывод адресов и портов
-t	Таблица для обработки
-v	Расширенный режим
-x	Расширить числа (вывести все значения)
-f	Проверять только второй и последующие фрагменты
-V	Версия пакета

--line-numbers Указывать номера строк при выводе

Сначала попробуем заблокировать все ICMP-пакеты, идущие на наш компьютер, просто для того, чтобы познакомиться с iptables поближе.

Листинг 1: Блокировка всех пакетов ICMP

```
# iptables -A INPUT -p icmp -j DROP
```

Сначала мы указываем цепочку, в которую хотим поместить правило, затем протокол проверяемого пакета и в конце цель. Цель может быть именем пользовательской цепочки или одной из специальных целей — ACCEPT, DROP, REJECT, LOG, QUEUE или MASQUERADE. В случае, если мы укажем DROP, то пакет будет отброшен без уведомления клиента.

Примечание: LOG является «необрывающей» целью. Если пакет подпадает под правило с целью LOG, то он будет передан следующему правилу в цепочке, а не обработан только этим правилом. Это позволяет регистрировать пакеты и при этом их обрабатывать.

Теперь попробуйте выполнить `ping localhost`. Вы не должны получить ответа, так как iptables будет отбрасывать все входящие ICMP-сообщения. Вы также не сможете проверить и другие компьютеры, так как ответные ICMP-пакеты тоже будут отбрасываться. А теперь, чтобы вновь получать ICMP, очистим цепочку.

Листинг 2: Сбросить все правила

```
# iptables -F
```

Теперь взглянем на полноценную фильтрацию пакетов с помощью iptables. Если нужно полноценное исследование входящих пакетов на интерфейсе eth0, то выполним следующую команду:

Листинг 3: Принимать пакеты от уже установленных соединений

```
# iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Это разрешит принимать любой пакет из уже установленного соединения или связанного в цепочке INPUT. Вы можете отбросить любой пакет, не

соответствующий состоянию таблицы, выполнив команду `iptables -A INPUT -i eth0 -m state --state INVALID -j DROP` сразу же после предыдущей. Эта команда включит фильтрацию пакетов на основании состояния, подключив внешнее расширение «state». Если вам необходимо разрешить подключаться клиентам к компьютеру, то вы можете использовать флаг `--state NEW`. Iptables содержит несколько модулей различного назначения. Вот некоторые из них:

Модуль/сравнение	Описание	Расширенные параметры
mac	Расширение для проверки MAC-адресов входящих пакетов.	<code>--mac-source</code>
state	Включение проверки состояния соединения	<code>--state</code> (состоянием может быть ESTABLISHED,RELATED, INVALID, NEW)
limit	Частотное ограничение	<code>--limit</code> , <code>--limit-burst</code>
owner	Попытка проверить различные характеристики создателя пакета	<code>--uid-owner</code> userid <code>--gid-owner</code> groupid <code>--pid-owner</code> processid <code>--sid-owner</code> sessionid
unclean	Различные случайны проверки пакетов на правильность	

Теперь попробуем создать пользовательскую цепочку и применить ее для одного из существующих:

Листинг 4: Создание пользовательской цепочки

```
(Создаем новую цепочку с одним правилом)
# iptables -X mychain
# iptables -N mychain
# iptables -A mychain -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
(Разрешающая политика по умолчанию для всего исходящего трафика. Входящий будет отброшен.)
# iptables -P OUTPUT ACCEPT
# iptables -P INPUT DROP
(И добавление ее в цепочку INPUT)
# iptables -A INPUT -j mychain
```

Применив правило к входящей цепочке, мы получим следующую политику: все исходящие пакеты будут пропущены, а все входящие — отброшены.

Документация может быть найдена по адресу [Netfilter/iptables documentation](http://netfilter.org/projects/iptables/documentation/).

Теперь взглянем на полный пример. Мои правила для межсетевого экрана/шлюза будут следующими:

- Подключения к межсетевому экрану разрешены только через SSH (порт 22)
- Локальная сеть должна иметь доступ к HTTP, HTTPS и SSH (также должен быть разрешен DNS)
- ICMP может содержать постороннюю информацию и не должен быть разрешен. Конечно же, мы разрешаем некоторые ICMP-сообщения.
- Сканирование портов должно быть определено и зарегистрировано
- SYN-атаки должны быть пресечены
- Весь остальной трафик должен быть отброшен и запротоколирован

Листинг 5: /etc/init.d/firewall

```
#!/sbin/runscript
IPTABLES=/sbin/iptables
IPTABLESSAVE=/sbin/iptables-save
IPTABLESRESTORE=/sbin/iptables-restore
FIREWALL=/etc/firewall.rules
DNS1=212.242.40.3
DNS2=212.242.40.51
#inside
IIP=10.0.0.2
IINTERFACE=eth0
LOCAL_NETWORK=10.0.0.0/24
#outside
OIP=217.157.156.144
OINTERFACE=eth1

opts="${opts} showstatus panic save restore showoptions rules"

depend() {
    need net
}

rules() {
    stop
    ebegin "Setting internal rules"

    einfo "Setting default rule to drop"
    $IPTABLES -P FORWARD DROP
    $IPTABLES -P INPUT DROP
    $IPTABLES -P OUTPUT DROP

    #default rule
    einfo "Creating states chain"
    $IPTABLES -N allowed-connection
    $IPTABLES -F allowed-connection
    $IPTABLES -A allowed-connection -m state --state ESTABLISHED,RELATED -j ACCEPT
    $IPTABLES -A allowed-connection -i $IINTERFACE -m limit -j LOG --log-prefix \
        "Bad packet from ${IINTERFACE}:"
    $IPTABLES -A allowed-connection -j DROP

    #ICMP traffic
    einfo "Creating icmp chain"
    $IPTABLES -N icmp_allowed
    $IPTABLES -F icmp_allowed
    $IPTABLES -A icmp_allowed -m state --state NEW -p icmp --icmp-type \
        time-exceeded -j ACCEPT
    $IPTABLES -A icmp_allowed -m state --state NEW -p icmp --icmp-type \
        destination-unreachable -j ACCEPT
    $IPTABLES -A icmp_allowed -p icmp -j LOG --log-prefix "Bad ICMP traffic:"
    $IPTABLES -A icmp_allowed -p icmp -j DROP

    #Incoming traffic
    einfo "Creating incoming ssh traffic chain"
    $IPTABLES -N allow-ssh-traffic-in
    $IPTABLES -F allow-ssh-traffic-in
    #Flood protection
    $IPTABLES -A allow-ssh-traffic-in -m limit --limit 1/second -p tcp --tcp-flags \
        ALL RST --dport ssh -j ACCEPT
    $IPTABLES -A allow-ssh-traffic-in -m limit --limit 1/second -p tcp --tcp-flags \
        ALL FIN --dport ssh -j ACCEPT
    $IPTABLES -A allow-ssh-traffic-in -m limit --limit 1/second -p tcp --tcp-flags \
        ALL SYN --dport ssh -j ACCEPT
    $IPTABLES -A allow-ssh-traffic-in -m state --state RELATED,ESTABLISHED -p tcp --dport ssh -j
    ACCEPT

    #outgoing traffic
    einfo "Creating outgoing ssh traffic chain"
    $IPTABLES -N allow-ssh-traffic-out
    $IPTABLES -F allow-ssh-traffic-out
    $IPTABLES -A allow-ssh-traffic-out -p tcp --dport ssh -j ACCEPT
```

```

einfo "Creating outgoing dns traffic chain"
$IPTABLES -N allow-dns-traffic-out
$IPTABLES -F allow-dns-traffic-out
$IPTABLES -A allow-dns-traffic-out -p udp -d $DNS1 --dport domain \
    -j ACCEPT
$IPTABLES -A allow-dns-traffic-out -p udp -d $DNS2 --dport domain \
    -j ACCEPT

einfo "Creating outgoing http/https traffic chain"
$IPTABLES -N allow-www-traffic-out
$IPTABLES -F allow-www-traffic-out
$IPTABLES -A allow-www-traffic-out -p tcp --dport www -j ACCEPT
$IPTABLES -A allow-www-traffic-out -p tcp --dport https -j ACCEPT

#Catch portscanners
einfo "Creating portscan detection chain"
$IPTABLES -N check-flags
$IPTABLES -F check-flags
$IPTABLES -A check-flags -p tcp --tcp-flags ALL FIN,URG,PSH -m limit \
    --limit 5/minute -j LOG --log-level alert --log-prefix "NMAP-XMAS:"
$IPTABLES -A check-flags -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
$IPTABLES -A check-flags -p tcp --tcp-flags ALL ALL -m limit --limit \
    5/minute -j LOG --log-level 1 --log-prefix "XMAS:"
$IPTABLES -A check-flags -p tcp --tcp-flags ALL ALL -j DROP
$IPTABLES -A check-flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG \
    -m limit --limit 5/minute -j LOG --log-level 1 --log-prefix "XMAS-PSH:"
$IPTABLES -A check-flags -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
$IPTABLES -A check-flags -p tcp --tcp-flags ALL NONE -m limit \
    --limit 5/minute -j LOG --log-level 1 --log-prefix "NULL_SCAN:"
$IPTABLES -A check-flags -p tcp --tcp-flags ALL NONE -j DROP
$IPTABLES -A check-flags -p tcp --tcp-flags SYN,RST SYN,RST -m limit \
    --limit 5/minute -j LOG --log-level 5 --log-prefix "SYN/RST:"
$IPTABLES -A check-flags -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
$IPTABLES -A check-flags -p tcp --tcp-flags SYN,FIN SYN,FIN -m limit \
    --limit 5/minute -j LOG --log-level 5 --log-prefix "SYN/FIN:"
$IPTABLES -A check-flags -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

# Apply and add invalid states to the chains
einfo "Applying chains to INPUT"
$IPTABLES -A INPUT -m state --state INVALID -j DROP
$IPTABLES -A INPUT -p icmp -j icmp_allowed
$IPTABLES -A INPUT -j check-flags
$IPTABLES -A INPUT -i lo -j ACCEPT
$IPTABLES -A INPUT -j allow-ssh-traffic-in
$IPTABLES -A INPUT -j allowed-connection

einfo "Applying chains to FORWARD"
$IPTABLES -A FORWARD -m state --state INVALID -j DROP
$IPTABLES -A FORWARD -p icmp -j icmp_allowed
$IPTABLES -A FORWARD -j check-flags
$IPTABLES -A FORWARD -o lo -j ACCEPT
$IPTABLES -A FORWARD -j allow-ssh-traffic-in
$IPTABLES -A FORWARD -j allow-www-traffic-out
$IPTABLES -A FORWARD -j allowed-connection

einfo "Applying chains to OUTPUT"
$IPTABLES -A OUTPUT -m state --state INVALID -j DROP
$IPTABLES -A OUTPUT -p icmp -j icmp_allowed
$IPTABLES -A OUTPUT -j check-flags
$IPTABLES -A OUTPUT -o lo -j ACCEPT
$IPTABLES -A OUTPUT -j allow-ssh-traffic-out
$IPTABLES -A OUTPUT -j allow-dns-traffic-out
$IPTABLES -A OUTPUT -j allow-www-traffic-out
$IPTABLES -A OUTPUT -j allowed-connection

#Allow client to route through via NAT (Network Address Translation)
$IPTABLES -t nat -A POSTROUTING -o $OINTERFACE -j MASQUERADE
eend $?
}

start() {
    ebegin "Starting firewall"
    if [ -e "${FIREWALL}" ]; then
        restore
    fi
}

```

```

else
    einfo "${FIREWALL} does not exists. Using default rules."
    rules
fi
eend $?
}

stop() {
    ebegin "Stopping firewall"
    $IPTABLES -F
    $IPTABLES -t nat -F
    $IPTABLES -X
    $IPTABLES -P FORWARD ACCEPT
    $IPTABLES -P INPUT ACCEPT
    $IPTABLES -P OUTPUT ACCEPT
    eend $?
}

showstatus() {
    ebegin "Status"
    $IPTABLES -L -n -v --line-numbers
    einfo "NAT status"
    $IPTABLES -L -n -v --line-numbers -t nat
    eend $?
}

panic() {
    ebegin "Setting panic rules"
    $IPTABLES -F
    $IPTABLES -X
    $IPTABLES -t nat -F
    $IPTABLES -P FORWARD DROP
    $IPTABLES -P INPUT DROP
    $IPTABLES -P OUTPUT DROP
    $IPTABLES -A INPUT -i lo -j ACCEPT
    $IPTABLES -A OUTPUT -o lo -j ACCEPT
    eend $?
}

save() {
    ebegin "Saving Firewall rules"
    $IPTABLESSAVE > $FIREWALL
    eend $?
}

restore() {
    ebegin "Restoring Firewall rules"
    $IPTABLESRESTORE < $FIREWALL
    eend $?
}

restart() {
    svc_stop; svc_start
}

showoptions() {
    echo "Usage: $0 {start|save|restore|panic|stop|restart|showstatus}"
    echo "start)      will restore setting if exists else force rules"
    echo "stop)       delete all rules and set all to accept"
    echo "rules)      force settings of new rules"
    echo "save)       will store settings in ${FIREWALL}"
    echo "restore)    will restore settings from ${FIREWALL}"
    echo "showstatus) Shows the status"
}

```

Вот несколько советов при создании правил для межсетевого экрана:

1. Создайте политику межсетевого экрана до того, как ее реализуете
2. Сделайте ее простой
3. Знайте принцип работы каждого протокола (прочитайте подходящий [RFC](#))
4. Всегда помните, что межсетевой экран — это просто программа,

запускаемая с правами администратора.

5. Проверьте свой межсетевой экран

Если вам кажется, что iptables труден для понимания или нужно слишком много времени для настройки межсетевого экрана, то вы можете попробовать [Shorewall](#). Для генерации правил межсетевого экрана он использует iptables, но акцентируется на правилах и не указывает протокол.

12.f. Squid

Squid является очень хорошим прокси-сервером. Он может фильтровать трафик на основании времени, регулярных выражений в пути/URI, IP-адреса получателя и отправителя, домена, браузера, имени зарегистрированного пользователя, типа MIME и номера порта (протокола). Некоторые возможности не указаны, но трудно перечислить их всех.

В следующем примере я добавил фильтр баннеров вместо фильтра, основанного на фильтрации порносайтов. По этой причине gentoo.org не должен быть перечислен в списке порносайтов. И я не собираюсь тратить свое время на поиски хороших сайтов для вас.

В данном случае, вот мои правила:

- Веб-серфинг (HTTP/HTTPS) разрешен только в рабочее время (с понедельника по пятницу с 8 до 17 часов и в субботу с 8 до 13 часов), но если служащие остаются, то они должны работать, а не сидеть в интернете.
- Скачивание файлов не разрешено (.exe, .com, .arj, .zip, .asf, .avi, .mpg, .mpeg и так далее)
- Нам не нужны баннеры, поэтому они будут отфильтрованы и заменены на прозрачный GIF (здесь пригодится ваша креативность!).
- Все остальные подключения в/из Интернета запрещены.

Все это реализуется в 4 простых шага.

Листинг 6: /etc/squid/squid.conf

```
# Указываем IP и порт
http_port 10.0.2.1:3128

# Стандартная конфигурация
hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY

# Добавляем основные списки контроля доступа
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255

# Добавляем тех, кто может пользоваться прокси
acl localnet src 10.0.0.0/255.255.0.0

# Добавляем порты
acl SSL_ports port 443
acl Safe_ports port 80
acl Safe_ports port 443
acl purge method PURGE

# Добавляем список контроля доступа, основанного на регулярных
```

```
# выражениях, встречаемых в URL
acl archives urlpath_regex "/etc/squid/files.acl"
acl url_ads url_regex "/etc/squid/banner-ads.acl"

# Добавляем список контроля доступа, основанного на времени и дате
acl restricted_weekdays time MTWHF 8:00-17:00
acl restricted_weekends time A 8:00-13:00

acl CONNECT method CONNECT

# Разрешаем менеджеру подключаться с localhost
http_access allow manager localhost
http_access deny manager

# Разрешать очищать запросы только с localhost
http_access allow purge localhost
http_access deny purge

# Запрещать запросы на неизвестные порты
http_access deny !Safe_ports

# Запрещать подключение к портам, не относящимся к SSL
http_access deny CONNECT !SSL_ports

# Мои собственные правила

# Добавляем страницу, отображаемую на месте
# удаленного баннера
deny_info NOTE_ADS_FILTERED url_ads

# Запрещаем баннеры
http_access deny url_ads

# Запрещаем любые архивы
http_access deny archives

# Ограничиваем доступ в рабочим временем
http_access allow localnet restricted_weekdays
http_access allow localnet restricted_weekends

# Запрещаем все остальное
http_access deny all
```

Далее перечислим типы запрещенных к скачиванию файлов. Я добавил zip, viv, exe, mp3, rar, ace, avi, mov, mpg, mpeg, au, ra, arj, tar, gz и z файлы.

Листинг 7: /etc/squid/files.acl

```
\.[Zz][Ii][pP]$
\.[Vv][Ii][Vv].*
\.[Ee][Xx][Ee]$
\.[Mm][Pp]3$
\.[Rr][Aa][Rr]$
\.[Aa][Cc][Ee]$
\.[Aa][Ss][Ff]$
\.[Aa][Vv][Ii]$
\.[Mm][Oo][Vv]$
\.[Mm][Pp][Gg]$
\.[Mm][Pp][Ee][Gg]$
```

```
\.[Aa][Uu]$
\[Rr][Aa]$
\[Aa][Rr][Jj]$
\[Tt][Aa][Rr]$
\[Gg][Zz]$
\[Zz]$
```

Примечание: Обратите внимание на [] с заглавными и строчными буквами. Это сделано для того, чтобы никто не смог обмануть наш фильтр, пытаясь скачать файл с расширением AvI вместо avi.

Далее добавим регулярные выражения для определения баннеров. Вы, возможно, будете более изобретательнее меня:

Листинг 8: /etc/squid/banner-ads.acl

```
/adv/*.gif$
/[Aa]ds/*.gif$
/[Aa]d[Pp]ix/
/[Aa]d[Ss]erver
/[Aa][Dd]/*.gif$
/[Bb]annerads/
/adbanner.*.gif$
/images/ad/
/reklame/
/RealMedia/ads/*
^http://www\.submit-it.*
^http://www\.eads.*
^http://ads\.
^http://ad\.
^http://ads02\.
^http://adaver.*.
^http://adforce\.
adbot\.com
/ads/*.gif.*
_ad\.*cgi
/Banners/
/SmartBanner/
/Ads/Media/Images/
^http://static\.wired\.com/advertising/
^http://*\.dejanews\.com/ads/
^http://adfu\.blockstackers\.com/
^http://ads2\.zdnet\.com/adverts
^http://www2\.burstnet\.com/gifs/
^http://www\.valueclick\.com/cgi-bin/cycle
^http://www\.altavista\.com/av/gifs/ie_horiz\.gif
```

И в заключении мы хотим, чтобы отображался следующий файл вместо удаленного баннера. Он основан половине HTML-файла с прозрачным GIF-изображением размером 4x4.

Листинг 9: /etc/squid/errors/NOTE_ADS_FILTERED

```
<HTML>
<HEAD>
<META HTTP-EQUIV="REFRESH" CONTENT="0; URL=http://localhost/images/4x4.gif">
<TITLE>ERROR: The requested URL could not be retrieved</TITLE>
</HEAD>
<BODY>
```

```
<H1>Add filtered!</H1>
```

Примечание: Не закрывайте теги `<HTML>` и `<BODY>`. Squid сделает это самостоятельно.

Как вы видите, у Squid есть множество возможностей для очень эффективной фильтрации и кэширования. Можно даже использовать альтернативные прокси-сервера Squid для сегментирования очень больших сетей. Приведенная конфигурация подходит для небольшой сети с 1—20 пользователями.

Однако комбинация межсетевого экрана (iptables) и программного шлюза (Squid), возможно, является наилучшей, особенно если Squid находится где-нибудь в безопасном месте, где никто не может иметь к нему доступ извне. Нам все же стоит позаботиться об атаках изнутри.

Теперь вам необходимо настроить клиентские браузеры для использования прокси-сервера. Шлюз предотвратит попытки пользователей общаться с внешним миром без использования прокси.

Примечание: В Mozilla это может быть сделано через Edit->Preferences->Advanced->Proxies.

Также можно настроить прозрачное использование прокси, указав iptables перенаправлять весь исходящий трафик на вход Squid. Это можно сделать, добавив следующее правило на шлюзе:

Листинг 10: Разрешить проброс портов для нашего прокси-сервера

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -j DNAT --to proxyhost:3128
# iptables -t nat -A PREROUTING -p tcp --dport 443 -j DNAT --to
proxyhost:3128
```

Примечание: Если прокси-сервер работает на том же узле, что и межсетевой экран (хотя это не рекомендуется, но может понадобиться из-за отсутствия свободных компьютеров), то используйте цель REDIRECT вместо DNAT (REDIRECT направит пакеты на localhost).

12.g. Изучено

Мы узнали:

1. Межсетевой экран является опасным сам по себе. Плохо настроенный межсетевой экран хуже, чем его отсутствие.
2. Как установить простой шлюз и прозрачный прокси-сервер.
3. Залогом хорошей работы межсетевого экрана заключается в знании протоколов, которые вы собираетесь разрешить.
4. IP-трафик не всегда содержит законные данные, например пакеты ICMP, которые могут содержать вредоносное содержимое.
5. Как предотвратить SYN-атаку.
6. Фильтрация HTTP-трафика предотвратит загрузку нежелательных изображений и вирусов.
7. Сочетание пакетных фильтров и программных шлюзов дает лучший контроль.

Теперь, если вам действительно это нужно, идите и создайте правила межсетевого экрана, отвечающего вашим требованиям.

13. Обнаружение вторжения

13.a. AIDE (Advanced Intrusion Detection Environment)

AIDE — это система обнаружения атак, основанная на узле (Host-Based Intrusion Detection System, HIDS), свободная альтернатива Tripwire (если вы уже работали с Tripwire, то не должны испытывать сложности с конфигурационным файлом AIDE). HIDS используется для обнаружения изменений в важных системных файлах настроек и двоичных файлах с помощью сверки уникальных криптографических контрольных сумм, созданных каждого из файлов и сохраненных в надежном месте. Принцип действия таков: заранее сохраненный и достоверный хеш сравнивается с сгенерированным с текущей копии каждого файла, чтобы определить какой из файлов был изменен. HIDS является великолепным решением для регистрации подозрительных изменений в системе, однако для того, чтобы она заработала, нужно приложить небольшие усилия.

Конфигурационный файл основан на применении регулярных выражений, макросов и правил для файлов и каталогов. Есть следующие макросы:

Макрос	Описание	Синтаксис
ifdef	если определен	@@ifdef "name"
ifndef	если не определен	@@ifndef "name"
define	определить переменную	@@define "name" "value"
undef	удалить переменную	@@undef "name"
ifhost	если "hostname"	@@ifhost "hostname"
ifnhost	если не "hostname"	@@ifnhost "hostname"
endif	endif должен использоваться каждый раз, когда используется любой из вышеприведенных макросов, за исключением define и undef	@@endif

Если у вас есть много компьютеров с Gentoo и вы хотите использовать на них AIDE, то макросы будут весьма полезными. Но не на всех компьютерах запускаются одни и те же сервисы или есть одни и те же пользователи.

Теперь необходимо установить флаги проверок для файлов и каталогов. Возможны сочетания прав доступа, свойств файлов и криптографических хешей (контрольных сумм).

Флаг	Описание
p	права доступа
i	inode
n	количество ссылок
u	пользователь
g	группа
s	размер
b	число блоков

m	время последней модификации
a	время последнего доступа
c	время изменения
S	проверка растущего размера
md5	контрольная сумма MD5
sha1	контрольная сумма SHA1
rm160	контрольная сумма RMD160
tiger	контрольная сумма Tiger
R	p+i+n+u+g+s+m+c+md5
L	p+i+n+u+g
E	Пустая группа
>	растущий лог-файл p+u+g+i+n+S

А если AIDE собран с поддержкой mhash, то доступны следующие возможности:

Флаг Описание

haval контрольная сумма HAVAL

gost контрольная сумма ГОСТ

crc32 контрольная сумма CRC32

Теперь вы можете создать собственные правила, основанные на сочетании вышеперечисленных флагов, например:

Листинг 1: Создание набора правил для AIDE

```
All=R+a+sha1+rm160
Norm=s+n+b+md5+sha1+rm160
```

И последнее, что нам понадобится для создания собственного конфигурационного файла, — это умение добавлять правило для файла или каталога. Чтобы ввести правило, скомбинируйте файл или каталог с правилом. AIDE добавляет все файлы рекурсивно, если вы не указали обратное в другом правиле.

Флаг Описание

! Не добавлять этот файл или каталог.

= Добавлять этот каталог, но не рекурсивно.

А теперь взглянем на полный пример:

Листинг 2: /etc/aide/aide.conf

```
@@ifndef TOPDIR
@@define TOPDIR /
@@endif

@@ifndef AIDEDIR
@@define AIDEDIR /etc/aide
@@endif

@@ifhost smbserve
@@define smbactive
```

```

@@endif

# Расположение базы данных для чтения.
database=file:#{@AIDEDIR}/aide.db

# Расположение базы данных для записи.
database_out=file:aide.db.new

verbose=20
report_url=stdout

# Определение правил
All=R+a+sha1+rmdbl60
Norm=s+n+b+md5+sha1+rmdbl60

@@{TOPDIR} Norm
!@@{TOPDIR}etc/aide
!@@{TOPDIR}dev
!@@{TOPDIR}media
!@@{TOPDIR}mnt
!@@{TOPDIR}proc
!@@{TOPDIR}root
!@@{TOPDIR}sys
!@@{TOPDIR}tmp
!@@{TOPDIR}var/log
!@@{TOPDIR}var/run
!@@{TOPDIR}usr/portage
@@ifdef smbactive
!@@{TOPDIR}etc/smb/private/secrets.tdb
@@endif
=@@{TOPDIR}home Norm

```

В приведенном примере мы указали некоторые макросы, корневой каталог, с которого будет начинать свою работу AIDE, и каталог, в котором находится AIDE. AIDE сверяется с `/etc/aide/aide.db` при проверке на целостность. Но при обновлении или создании нового файла она сохраняет информацию в `/etc/aide/aide.db.new`. Это делается для того, чтобы не переписать старую базу данных. Параметр `report_URL` пока не реализован, но, согласно автору, он должен отправлять электронную почту или даже запуск сценариев.

Теперь по умолчанию пакет AIDE поставляется вместе с рабочим конфигурационным файлом, сценарием-помощником и сценарием планировщика заданий `crontab`. Сценарий-помощник содержит множество задач для вас и предоставляет более дружелюбный интерфейс. Чтобы просмотреть все доступные параметры, попробуйте `aide --help`. Все что вам нужно, чтобы начать, — запустить `aide -i`, и теперь сценарий `crontab` должен найти базу данных и ежедневно отправлять отчеты по электронной почте. Рекомендуется все же пересмотреть файл `/etc/aide/aide.conf` и удостовериться, что конфигурация подходит для данного компьютера.

Примечание: В зависимости от процессора, скорости чтения диска и установленных для файлов флагов, это может занять много времени.

Примечание: Не забудьте установить алиасы, чтобы получать почту, адресуемую суперпользователю. Иначе вы никогда не узнаете, что пишет для вас в отчетах AIDE.

Все же есть небольшой риск, заключающийся в том, что злоумышленник может (если узнает об установленном AIDE) изменить локальную базу данных файлов, обновить ее или изменить `/usr/bin/aide`. Поэтому вы должны создать компакт-диск или иной носитель и сохранить на нем копию файла `.db` и двоичных файлов AIDE. Дополнительную информацию вы можете найти на сайте проекта [AIDE](#)

13.b. Snort

Snort является сетевой системой обнаружения вторжения (Network Intrusion Detection System, NIDS). Чтобы установить и настроить его, воспользуйтесь следующими примерами.

Листинг 3: `/etc/conf.d/snort`

```
PIDFILE=/var/run/snort_eth0.pid
MODE="full"
NETWORK="10.0.0.0/24"
LOGDIR="/var/log/snort"
CONF=/etc/snort/snort.conf
SNORT_OPTS="-D -s -u snort -dev -l $LOGDIR -h $NETWORK -c $CONF"
```

Листинг 4: `/etc/snort/snort.conf`

```
(#Mar 1)
var HOME_NET 10.0.0.0/24
var EXTERNAL_NET any
var SMTP $HOME_NET
var HTTP_SERVERS $HOME_NET
var SQL_SERVERS $HOME_NET
var DNS_SERVERS [10.0.0.2/32,212.242.40.51/32]
var RULE_PATH ./

(#Mar 2)
preprocessor frag2
preprocessor stream4: detect_scans detect_state_problems detect_scans disable_evasion_alerts
preprocessor stream4_reassemble: ports all
preprocessor http_decode:80 8080 unicode iis_alt_unicode double_encode iis_flip_slash
full_whitespace
preprocessor rpc_decode: 111 32771
preprocessor bo: -nobrute
preprocessor telnet_decode

(#Mar 3)
include classification.config

(#Mar 4)
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/x11.rules
```

```
include $RULE_PATH/icmp.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/shellcode.rules
include $RULE_PATH/policy.rules
include $RULE_PATH/porn.rules
include $RULE_PATH/info.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/virus.rules
# include $RULE_PATH/experimental.rules
include $RULE_PATH/local.rules
```

Листинг 5: /etc/snort/classification.config

```
config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic, 2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege Gain,1
config classification: successful-user,Successful User Privilege Gain,1
config classification: attempted-admin,Attempted Administrator Privilege Gain,1
config classification: successful-admin,Successful Administrator Privilege Gain,1

# NEW CLASSIFICATIONS
config classification: rpc-portmap-decode,Decode of an RPC Query,2
config classification: shellcode-detect,Executable code was detected,1
config classification: string-detect,A suspicious string was detected,3
config classification: suspicious-filename-detect,A suspicious filename was detected,2
config classification: suspicious-login,An attempted login using a suspicious username was detected,2
config classification: system-call-detect,A system call was detected,2
config classification: tcp-connection,A TCP connection was detected,4
config classification: trojan-activity,A Network Trojan was detected, 1
config classification: unusual-client-port-connection,A client was using an unusual port,2
config classification: network-scan,Detection of a Network Scan,3
config classification: denial-of-service,Detection of a Denial of Service Attack,2
config classification: non-standard-protocol,Detection of a non-standard protocol or event,2
config classification: protocol-command-decode,Generic Protocol Command Decode,3
config classification: web-application-activity,access to a potentially vulnerable web application,2
config classification: web-application-attack,Web Application Attack,1
config classification: misc-activity,Misc activity,3
config classification: misc-attack,Misc Attack,2
config classification: icmp-event,Generic ICMP event,3
config classification: kickass-porn,SCORE! Get the lotion!,1
```

Дополнительной информация может быть найдена на сайте [Snort](http://snort.org).

13.с. Обнаружение вредоносных программ с помощью chkrootkit

Система HIDS, как AIDE, является лучшим средством для обнаружения изменений в вашей системе, но хуже не будет, если будет установлен еще один рубеж обороны. chkrootkit — это утилита, которая сканирует системные файлы на наличие руткитов — программ, разработанных для сокрытия присутствия взломщика и сохранения доступа к системе, а также на наличие следов кейлоггеров и прочих вредоносных программ. Хотя chkrootkit (и прочие аналоги, например rkhunter) являются полезными утилитами, как контроля за системой, так и для обнаружения следов вторжения, но они не могут гарантировать, что ваша система в безопасности.

Наилучший способ использовать chkrootkit для обнаружения внедрения — запускать его с помощью cron. Сначала установите app-admin/chkrootkit. chkrootkit может запускаться из командной строки или же с помощью записи в cron следующего вида:

Листинг 6: Назначение задания chkrootkit в crontab

```
0 3 * * * /usr/sbin/chkrootkit
```

14. Регулярные обновления

14.a. Регулярные обновления

После успешной установки и настройки ее безопасности систему, тем не менее, нельзя называть завершенной. Процесс обеспечения безопасности является постоянным по времени, ведь большинство известных взломов является результатом использования известных уязвимостей в устаревших системах. Постоянное обновление системы — наиболее значимый шаг, который вы можете совершить для обеспечения наилучшей безопасности.

Если у вас установлена последняя версия portage, то, предварительно обновив дерево с помощью команды emerge --sync, вы можете запустить команду glsa-check --list, которая проверит вашу систему на наличие известных уязвимостей. glsa-check является частью пакета app-portage/gentoolkit.

Листинг 1: Пример вывода glsa-check -l

```
# glsa-check -l
WARNING: This tool is completely new and not very tested, so it should not be
used on production systems. It's mainly a test tool for the new GLSA release
and distribution system, it's functionality will later be merged into emerge
and equery.
Please read http://www.gentoo.org/proj/en/portage/glsa-integration.xml
before using this tool AND before reporting a bug.

[A] means this GLSA was already applied,
[U] means the system is not affected and
[N] indicates that the system might be affected.

200406-03 [N] sitecopy: Multiple vulnerabilities in included libneon ( net-misc/sitecopy )
200406-04 [U] Mailman: Member password disclosure vulnerability ( net-mail/mailman )
.....
```

Предупреждение: Утилита glsa-check все еще является экспериментальной, поэтому, если вы действительно ставите безопасность превыше всего, то лучше также свериться с другими источниками.

Все строки, содержащие [A] и [U], могут быть игнорированы, так как они неприменимы для данной системы.

Важно: Не забывайте, что использование emerge -vpuD world не устанавливает все необходимые обновления пакетов. Используйте glsa-check, если хотите быть уверенным, что все GLSA исправлены в вашей системе.

Листинг 2: Проверка всех GLSA

```
(Подвержена ли ваша система GLSA?)
# glsa-check -t all
WARNING: This tool is completely new and not very tested, so it should not be
used on production systems. It's mainly a test tool for the new GLSA release
and distribution system, it's functionality will later be merged into emerge
```

and equery.
Please read <http://www.gentoo.org/proj/en/portage/glsa-integration.xml>
before using this tool AND before reporting a bug.

This system is affected by the following GLSA:

200504-06
200510-08
200506-14
200501-35
200508-12
200507-16

(Просмотр всех пакетов, подверженных переустановке)

```
# glsa-check -p $(glsa-check -t all)
```

(частичный вывод)

Checking GLSA 200504-06

The following updates will be performed for this GLSA:

app-arch/sharutils-4.2.1-r11 (4.2.1-r10)

Checking GLSA 200510-08

The following updates will be performed for this GLSA:

media-libs/xine-lib-1.1.0-r5 (1.1.0-r4)

(применение необходимых исправлений)

```
# glsa-check -f $(glsa-check -t all)
```

Если вы обновили запущенный сервис, не забудьте перезапустить его.

Также рекомендуется регулярно [обновлять ядро](#).

Вы можете также подписаться на список рассылки gentoo-announce, чтобы получать уведомления GLSA по электронной почте. Инструкции по подписке могут быть найдены на странице [Введение в почтовые рассылки Gentoo Linux](#).

Другим хорошим источником информации по безопасности является [список рассылки Bugtraq](#).

Hardened Gentoo

Ссылка на оригинал: <http://www.habrahabr.ru/blog/linux/22815.html>

Автор: powerman.

С версии: 1.5.

Описание

Для начала расскажу, зачем я публикую эту статью. Дело в том, что большинство пользователей Gentoo Linux до сих пор не использует [Hardened Gentoo](#). И вызвано это обычно тем, что они либо не знают, что это такое, либо считают что это слишком сложно, либо считают что от этого пострадает стабильность, функциональность или производительность системы. Вот эти опасения я и хочу попытаться развеять.

Hardened Gentoo - это несколько изменений в компиляторе и ядре, которые увеличивают общую защищенность системы от взлома. Например, hardened-ядро умеет блокировать массу потенциально опасных операций, а hardened-gcc позволяет защитить компилируемые им программы от взлома типовыми методами а-ля переполнение буфера. Грубо говоря, если у вас стоит "дырявая" версия программы X, и её пытается взломать хакер, то в обычной системе у него это получится, а в hardened - не получится, да ещё и в лог запись пойдёт.

Для установки Hardened на обычный Gentoo нужно будет полностью перекомпилировать всю систему - иначе защиты предоставляемые hardened-gcc не будут использоваться. Hardened - это ещё одна система, которую нужно аккуратно настраивать - переборщишь с защищённостью - ничего работать не будет вообще, недоборщишь - зачем тогда вообще Hardened ставить? Некоторые проги с hardened не уживаются (обычно это бинарные пакеты: дрова nVidia/ATI, Java плюс почему-то такой софт как mplayer/xmms) - это не смертельно, просто приходится для этих прог индивидуально отключать часть защит (для этого есть утилитки)... что огорчает. Ядро используется из пакета hardened-sources, и оно обычно на пару версий отстаёт от gentoo-sources.

Итак, Hardened Gentoo это просто объединение нескольких разных, часто независимых друг от друга, проектов:

1. Hardened toolchain - специальные патчи на gcc/glibc/binutils:

- [SSP](#) - добавляет в бинарник защиту от переполнения буфера, т.е. прога откомпилированная с SSP сама проверяет что у неё не переполнили буфер и килит сама себя если обнаруживает переполнение (как следствие бага в самой программе или попытки её взломать эксплойтом).
- [PIE](#) - не увеличивает защищённость сам по себе, но приводит к генерации более гибкого кода, благодаря чему его можно будет защитить на уровне ядра через PaX.

PIE и SSP не зависят друг от друга, и их можно использовать вместе и по отдельности (после компиляции hardened toolchain можно будет через gcc-

config переключаться между всеми вариантами - PIE+SSP, только PIE, только SSP, ничего (т.е. обычный gcc) - например, если какая-то прога не будет компилироваться.

2. Патчи на ядро. Их бывает много, и разных, :) но в Gentoo есть поддержка только четырёх из них - [PaX](#), [SeLinux](#), [GrSecurity](#) и [RSBAC](#). Функциональность они добавляют трех типов:

- Защита от переполнения буфера (а-ля SSP но со стороны ядра и другими методами, так что они друг друга дополняют): PaX. Например, PaX позволяет запретить выполнение кода в страницах памяти с данными (софтверная реализация NX-бита, которые появился только в 64-битных процах Intel) - PaX просто кильнёт процесс если он попытается нарушить эту защиту; при загрузке программы в память грузит все её функции по случайным адресам, чтобы эксплойту было очень сложно узнать на какой адрес передавать управление (это становится возможным благодаря компиляции с PIE).
- Отключение потенциально опасных "фич" ядра: GrSecurity, RSBAC. Пример: запрет выполнять mount внутри chroot - чтобы хакер взломавший chroot-нутый демон и получивший root-а не смог выйти из chroot.
- Ограничение прав процессов и юзеров, в т.ч. (я бы даже сказал - в основном) юзера root: SeLinux, GrSecurity/RBAC, RSBAC. Здесь идея в том, что админу (вам :) нужно подготовить список с указанием какие проги/юзеры что имеют право делать. Пример: можно ограничить root-овый процесс apache из всех прав root-а только возможностью садиться на 80-й порт и читать файлы в /etc/apache2/. В этом случае даже если его и взломают, и хакер получит "root", то ЭТОТ "root" сможет делать только вышеперечисленные операции... хакер будет крайне разочарован. :)

Эти три "фичи" тоже не зависят одна от другой. Но сами патчи - SeLinux, GrSecurity и RSBAC обычно между собой не совместимы и нужно использовать только один из них. Впрочем, в Gentoo сумели объединить SeLinux и GrSecurity вместе. Часть GrSecurity, которая занимается третьей фичей (ограничением прав) называется RBAC, и её использовать вместе с SeLinux нельзя - или-или.

Итого, варианты есть, например, следующие:

- PaX + GrSecurity
- PaX + GrSecurity (без RBAC) + SeLinux
- PaX + SeLinux
- PaX + RSBAC
- ... etc ...

Я выбрал первый (PaX+GrSecurity) т.к. во-первых настройка SeLinux обещает быть кошмаром, в отличие от GrSecurity/RBAC; во-вторых на мой взгляд поддержка RSBAC в Gentoo ещё сыровата; и в-третьих ну понравился мне GrSecurity, понравился. :))

Установка

Процесс установки Hardened Gentoo (точнее, апгрейда текущей Gentoo системы в Hardened) выглядит примерно так:

1. переключаемся на hardened toolchain и пересобираем им всю систему, чтобы все бинарники использовали PIE и SSP (после этого система становится защищена SSP)
2. устанавливаем hardened-sources (они содержат патчи PaX + GrSecurity + SeLinux + дополнительные от Gentoo) и компилируем их с поддержкой PaX, GrSecurity и GrSecurity/RBAC
3. перегружаемся с новым ядром (после этого система становится защищена ещё и PaX+PIE и GrSecurity)
4. некоторое время настраиваем и отлаживаем ограничения доступа (после чего система становится защищена ещё и GrSecurity/RBAC)

Ожидаемые проблемы:

- не всё может скомпилироваться с PIE+SSP - возможно отдельные пакеты нужно будет патчить или компилировать без одной или обоих из них (мне пока потребовалось через gcc-config переключаться на vanilla gcc только для компиляции X-ов чтобы они работали с ATI-драйверами)
- не всё может нормально работать, т.к. некоторые программы (обычно упоминают X-ы и java) используют выполнение динамически сгенерированного кода для вполне легальных целей, а теперь при попытке это делать они будут клясться либо SSP либо PIE+PaX - для этих программ нужно будет индивидуально отключать часть защит PaX (для этого есть специальные утилиты, например paxctl) и/или компилировать их без SSP
- не всё может работать из-за ограничений "фич" ядра GrSecurity - в этом случае нужно будет часть защит GrSecurity отключать (глобально, в make menuconfig)
- настроить ограничения прав доступа может оказаться не просто, и в любой момент когда какая-нибудь прога сделает что-то, что мы при настройке её прав не учли - она будет пририта ядром... и придётся эти правила в срочном порядке фиксировать

Ну что, поехали... :)

Слишком сильная оптимизация (-O3) вместе с hardened toolchain может приводить к разным глюкам и сбоям компиляции, поэтому нужно в /etc/make.conf заменить -O3 на -O2, и убрать все прочие -f* оптимизаторские флаги.

Переходим на hardened-профайл. (Теоретически вместо этого можно было просто добавить в USE-флаги: "hardened pie ssp".)

```
In -snf ../usr/portage/profiles/hardened/x86/2.6/ /etc/make.profile
```

После переключения профайла на hardened/x86/2.6/ выключилось несколько

нужных мне USE-флагов, которые в обычных профайлах включены автоматически - я их дописал в `make.conf` (у вас флаги могут быть другие, просто не забудьте этот момент проследить):

```
avi encode gtk2 jpeg mpeg oss quicktime spell truetype xv bitmap-fonts truetype-fonts
type1-fonts
```

Компиляция `hardened-toolchain` и пересборка им всех остальных пакетов:

```
emerge binutils gcc glibc emerge -e world
```

```
dispatch-conf
```

Далее, нужно поставить ещё несколько пакетов:

```
emerge paxtest paxctl gradm
```

`paxtest` можно было поставить и раньше, до перехода на `hardened toolchain`. Эта утилита пытается делать разные опасные операции (типа переполнения стека с выполнением кода), которые обычно выполняют эксплойты. Если система защищена, то все её попытки будут пресечены, о чём она и сообщит. В общем, можно её погонять до установки `hardened toolchain`, после а так-же после перезагрузки с ядром где включён PaX, по приколу. :)

`paxtest` как свою зависимость так-же установит прогу `chpax` - это старый способ управлять PaX, вместо которого лучше использовать `paxctl`. Но некоторые проги, распространяемые без исходников, скомпилированы так, что `paxctl` с ними работать не может, и приходится юзать `chpax`.

Ну а `gradm` нужен для настройки RBAC в GrSecurity - тех самых ограничений прав для процессов и юзеров.

Что касается настроек ядра. Я для начала настроил всё так, как описано в доке Gentoo и GrSecurity - при этом не все защиты включены, но большинство софта при таких настройках будет работать. В процессе разбора с настройками возникло несколько теоретических предположений:

- Если загрузить `hardened` ядро ДО перекомпилирования системы, то могут возникнуть проблемы если включена эта опция:
PaX -> Non-executable pages -> Disallow ELF text relocations
- На сервере, где нет X-ов, можно дополнительно включить: (не забыть перед этим проверить, что от этого не перестало работать что-то кроме X-ов, в частности `hwclock`):
Grsecurity -> Address Space Protection -> Disable privileged I/O
- Кроме того, ещё есть предположение, что некоторые ограничения `chroot` могут помешать операциям типа инсталляции Gentoo (которая идёт внутри `chroot`) или попытке починки системы (если, например, для этого загрузиться с CD с таким `hardened` ядром):
Grsecurity -> Filesystem Protections -> Deny mounts
Grsecurity -> Filesystem Protections -> Deny double-chroots
Grsecurity -> Filesystem Protections -> Deny (f)chmod +s
Grsecurity -> Filesystem Protections -> Deny mknod

Настройка

Настройка Hardened состоит из двух частей: настройка ядра, и настройка ролей (RBAC или SELinux). Пример настройки ядра 2.6.20 я сейчас покажу. А RBAC/SELinux я пока не настраивал.

Думаю, будет нагляднее всего привести настройки так, как они выглядят в make menuconfig - это за одно позволит оценить "на глаз" возможности PaX и GrSecurity тем, кто с ними до сих пор не сталкивался.

Между разными версиями ядра они немного меняются, но не значительно.

```
PaX --->
. [*] Enable various PaX features..... PaX Control --->
..... [ ] Support soft mode
..... [*] Use legacy ELF header marking
..... [*] Use ELF program header marking
..... MAC system integration (none) --->
..... Non-executable pages --->
..... [*] Enforce non-executable pages
..... [*] . Segmentation based non-executable pages
..... [ ] Emulate trampolines
..... [*] Restrict mprotect()
..... [ ] . Disallow ELF text relocations
..... [ ] Enforce non-executable kernel pages
..... Address Space Layout Randomization --->
..... [*] Address Space Layout Randomization
..... [*] . Randomize kernel stack base
..... [*] . Randomize user stack base
..... [*] . Randomize mmap() base
..... Miscellaneous hardening features --->
..... [ ] Sanitize all freed memory
..... [*] Prevent invalid userland pointer dereference
Grsecurity --->
. Security Level (Custom) --->
. Address Space Protection --->
... [*] Deny writing to /dev/kmem, /dev/mem, and /dev/port
... [ ] Disable privileged I/O
... [*] Remove addresses from /proc//[smaps|maps|stat]
... [*] Deter exploit bruteforcing
... [*] Runtime module disabling
... [*] Hide kernel symbols
. Role Based Access Control Options --->
... [*] Hide kernel processes
... (3) Maximum tries before password lockout
... (30) Time to wait after max password tries, in seconds
. Filesystem Protections --->
... [*] Proc restrictions
... [*] . Restrict /proc to user only
... [*] Additional restrictions
... [*] Linking restrictions
... [*] FIFO restrictions
... [*] Chroot jail restrictions
... [*] . Deny mounts
... [*] . Deny double-chroots
... [*] . Deny pivot_root in chroot
... [*] . Enforce chdir("/") on all chroots
... [*] . Deny (f)chmod +s
... [*] . Deny fchdir out of chroot
```

```

... [*] . Deny mknod
... [*] . Deny shmat() out of chroot
... [*] . Deny access to abstract AF_UNIX sockets out of chroot
... [*] . Protect outside processes
... [*] . Restrict priority changes
... [*] . Deny sysctl writes
... [*] . Capability restrictions
. Kernel Auditing --->
... [ ] Single group for auditing
... [ ] Exec logging
... [*] Resource logging
... [ ] Log execs within chroot
... [ ] Chdir logging
... [*] (Un)Mount logging
... [ ] IPC logging
... [*] Signal logging
... [*] Fork failure logging
... [ ] Time change logging
... [*] /proc//ipaddr support
... [ ] ELF text relocations logging (READ HELP)
. Executable Protections --->
... [*] Enforce RLIMIT_NPROC on execs
... [*] Destroy unused shared memory
... [*] Dmesg(8) restriction
... [ ] Trusted Path Execution (TPE)
. Network Protections --->
... [*] Larger entropy pools
... [ ] Socket restrictions
. Sysctl support --->
... [*] Sysctl support
... [*] . Turn on features by default
. Logging Options --->
... (10) Seconds in between log messages (minimum)
... (4) Number of messages in a burst (maximum)
[ ] Enable access key retention support
[ ] Enable different security models

```

Абсолютное большинство этих фич работает прозрачно для пользователя. Но есть пара, которые вы можете заметить: во-первых обычные пользователи перестанут видеть процессы других пользователей, а во-вторых они потеряют доступ к некоторым файлам в /proc/, из-за чего вывод команд ifconfig, route, etc. запущенных обычными пользователями станет значительно скромнее.

Надо отметить, что большая часть этих фич может управляться через sysctl. Что, как правило, плохо. А что хорошего в том, что хакер ломает систему наполовину, получает возможность отключить эти защиты через sysctl, после чего доламывает её уже окончательно? К счастью, есть возможность заблокировать изменение настроек GrSecurity через sysctl. Для этого нужно добавить в /etc/sysctl.conf:

```

kernel.grsecurity.disable_modules = 1
kernel.grsecurity.grsec_lock = 1

```

Где первая команда запрещает подгрузку модулей ядра (лучше всего на серверах поддержку модулей даже не включать в ядре, но если это не возможно, то теперь есть выход: подгрузить нужные модули при загрузке системы, а потом с помощью kernel.grsecurity.disable_modules запретить подгрузку модулей - чтобы никто

случайно руткит не подгрузил :)), а вторая запрещает изменение любых настроек GrSecurity.

Недостаток этого в том, что если вам нужно будет таки подгрузить модуль или отключить часть фич GrSecurity (например, защиту chroot для сборки нового Gentoo), то придётся редактировать /etc/sysctl.conf и перезагружаться.

Собственно, настройка на этом окончена.

В качестве резюме, приведу полный набор команд, которые отконвертируют ваш Gentoo в Hardened:

emerge hardened-sources

6

Теперь настройте это ядро (пока не включая фичи
hardened), скомпилируйте и перезагрузитесь в него.

In -snf ../usr/portage/profiles/hardened/x86/2.6/ /etc/make.profile

Уберите все флаги оптимизации из CFLAGS в /etc/make.conf
и установите -O2.
Пример: CFLAGS="-march=pentium-m -O2 -pipe"

Очистите ваш \$PKGDIR (обычно /usr/portage/packages/) для
ускорения времени пересборки системы используя
опции -b и -k команды emerge.

emerge -C linux-headers

emerge linux-headers glibc binutils gcc-config gcc

Прделайте все дополнительные операции, которые могут
требоваться при обновлении gcc (см. GCC Upgrade Guide).

emerge -b glibc binutils gcc portage

emerge -bke system

emerge -ke world

glsa-check -l | grep '[N\']'

Ручками обновите пакеты, которые мог выдать glsa-check.

emerge -a --depclean

emerge -uDNa world

emerge paxtest paxctl gradm

revdep-rebuild

dispatch-conf

Теперь включите в ядре все фичи Hardened,
соберите его и перезагрузитесь.

Впечатления

Я начал с того, что выразил желание развеять распространённые опасения в том, что Hardened - это слишком сложно, либо от этого страдает стабильность, функциональность или производительность системы. Я уже продемонстрировал, что такое Hardened Gentoo в общих чертах, а теперь пройдемся детальнее по этим опасениям.

Сложность

Я пока не занимался настройкой системы ролей (RBAC/SELinux) как раз из-за сложности. Возможно, конечно, только кажущейся, и на самом деле там тоже всё просто, не знаю... :) Я, всё-таки, в основном программист, и на администрирование времени вечно не хватает. В общем, если в Hardened и есть что-то сложное и требующее кучу времени и внимания - это настройка ролей.

А вот всё, что я описал до этого, настраивается очень просто, быстро, **один раз**, и даёт достаточно сильный эффект в виде усиления безопасности системы!

Стабильность

Я уже много лет использую Hardened в описанном виде и дома на workstation, и на всех серверах. Никаких проблем со стабильностью их работы из-за Hardened за это время не возникало, и я не видел жалоб в maillist на эту тему.

Что касается стабильности сборки Gentoo, ведь пакеты постоянно обновляются и компилируются. Пару лет назад была необходимость использовать некоторые workaround-ы - например, чтобы XWindow работали с дровами ATI приходилось их собирать не-hardened gcc (для автоматического переключения gcc в процессе компиляции пакетов был написан простейший скрипт). Ну и ещё по мелочи проблем возникало, но ни одной критичной. Сейчас проблем такого рода нет в принципе. Т.е. вы систему на Hardened перевели, и забыли - можете продолжать всё делать так, как будто ничего не произошло. Только хакать ваш сервер стало сложнее гораздо. :)

Функциональность

Да, для того, чтобы работали некоторые пакеты, до сих пор необходимо использовать утилиты `chpax/paxctl`, для отключения части hardened-защит для конкретных приложений. Но в Gentoo эта операция давно автоматизирована: для этих приложений `chpax/paxctl` выполняется на этапе установки пакета. Так что вам об этом беспокоиться уже не нужно. А за исключением этого нюанса (что для некоторых приложений часть защит выключена), все приложения работают в Hardened Gentoo без проблем (по крайней мере с теми настройками ядра, которые я привёл в предыдущей части).

Производительность

Честно скажу - сам я не мерял. Да и что и как мерять - не совсем понятно. Насколько я понял из инета - можно ожидать до 3-4% потери производительности

в худшем случае, а обычно это будет меньше процента. Но, опять-же, смотря какие "фичи" включать. Если врубить SeLinux или RBAC - тогда можно эти 3-4% потерять, но до их настройки я пока не добрался. :(Визуально - после перехода на Hardened разницы никакой, ни на домашней машине, ни на серверах. Конечно, сколько-то производительности, вероятно, было потеряно из-за перехода на -O2. Ещё я сталкивался с тем, что при выборе "неправильной" опции при настройке ядра из этих двух:

- [] Paging based non-executable pages
- [*] Segmentation based non-executable pages

возникали заметные тормоза. Сейчас я в ядре вообще не вижу первого варианта (который любил потормозить на некоторых типах процессоров).

Выводы

В общем, у вас есть реальная возможность значительно увеличить безопасность своих машин, потратив **один раз** пару часов вашего времени на переход на Hardened и сутки машинного времени (пока будет полностью пересобираться система).

"Думайте сами, решайте сами - иметь, или не иметь." :)

Обновление Gentoo Linux

Руководство по обновлению Gentoo

Ссылка на оригинал: <http://www.gentoo.org/doc/en/gentoo-upgrading.xml>

С версии: 1.2

Перевод: [vnp](#), [Михаил Ярмиш](#), [Алексей Чумаков](#), [Батогов Е.В.](#), [a.k.a JohnBat26](#)

Примечание: Из оригинала включён только самый **актуальный** материал !

1. Gentoo и обновления

Философия

Здесь, на земле Gentoo, обновление понимается совсем не так, как во всем остальном мире Linux. Вы, вероятно, уже знаете, что в наших дистрибутивах мы никогда не следовали «классическому» способу обновления программ: дожидаться нового релиза, скачать его, прожечь, засунуть диск в дисковод и, наконец, следовать установочным инструкциям.

Вы знаете (вы же не зря выбрали Gentoo), как такой процесс раздражает опытного пользователя, стремящегося всегда быть на переднем крае. Учитывая популярность и распространенность инструментов типа **apt** или **apt-rpm**, облегчающих быстрые и частые обновления, мы полагаем, что и опытные пользователи других дистрибутивов испытывают те же чувства. Однако ни один дистрибутив не приспособлен для удовлетворения нужд требовательных пользователей лучше Gentoo, поскольку Gentoo с самого начала строился на идее быстрых последовательных обновлений.

В идеале, вы устанавливаете систему однажды, и более никогда не беспокоитесь о выпусках: просто следуйте инструкциям из введения в Portage в настольной книге Gentoo, где описывается, как поддерживать актуальность системы. Хотя обычно такого подхода достаточно, иногда в ядро системы вносятся изменения, требующие установки вручную.

Выпуски и профили

О процессе выпуска версий Gentoo постоянно задают вопрос: «Зачем нужно часто делать новые выпуски, если они не предназначены для обновления программ у пользователей?». Тому есть ряд причин:

Новый выпуск — это новые исправленные и дополненные установочные диски.

Новый выпуск несет обновленный набор пакетов GRP, так что пользователи, избирающие «быстрый способ» установки (stage3 + заранее скомпилированные пакеты), получают не устаревшую систему.

Наконец, в новых выпусках время от времени реализуются функции,

несовместимые с предыдущими выпусками.

Если в выпуск входят возможности, несовместимые с предыдущими, или содержится набор фундаментальных пакетов и установок, сильно меняющих поведение системы, мы говорим, что он задает новый профиль.

Профиль — это набор конфигурационных файлов, хранимый в подкаталоге `/usr/portage/profiles`, описывающий сценарии `ebuild` системных пакетов, флаги использования (USE) по умолчанию, отображение виртуальных пакетов по умолчанию, а также архитектуру, на которой запускается система.

Используемый профиль определяется символьной ссылкой `/etc/make.profile`, которая указывает на подкаталог в `/usr/portage/profiles`, содержащий файлы профиля. К примеру, профиль `x86 2007.0` по умолчанию находится в `/usr/portage/profiles/default-linux/x86/2007.0`. Файлы в родительских каталогах также входят в состав профиля (и, таким образом, совместно используются различными подпрофилями). Поэтому мы называем такие профили каскадными профилями.

Профили, устаревающие с появлением новых, хранятся в `/usr/portage/profiles` наравне с актуальными, но помечены как нежелательные для использования. Для этого в каталог профиля помещается файл **deprecated**. Файл содержит имя профиля, на который рекомендуется обновиться. Portage использует эту информацию для автоматического предупреждения о необходимости перехода на новый профиль.

Существуют различные причины для создания нового профиля: выпуск новых несовместимых версий базовых пакетов (таких, как **baselayout**, **gcc**, или **glibc**), изменения в USE-флагах по умолчанию или отображении виртуальных пакетов, или, возможно, серьезные системные изменения.

2. Гонка за новыми выпусками

Выпуски без изменений в профиле

Если объявлен новый выпуск Gentoo, в котором не появился новый профиль, то твердо можно притвориться, что вы его не заметили :).

Ваша система будет в точности соответствовать вновь установленной с нового выпуска, если вы обновите установленные пакеты согласно описанию из Настольной книги Gentoo.

Выпуски с изменениями в профиле

Если в выпуске появился новый профиль (как, например, в 2007.0 для x86), у вас есть возможность на него перейти.

Естественно, вы не обязаны это делать, и можете продолжать пользоваться старым профилем, просто обновляя свои пакеты, как описано в Настольной книге Gentoo.

Однако, Gentoo крайне рекомендует обновлять профиль, если он объявляется нежелательным для использования. Когда такое происходит, это означает, что разработчики больше не планируют его поддерживать. В приведенной ниже таблице можно быстро уточнить, какие профили поддерживаются в данный момент.

Переход на новый профиль придется осуществлять вручную. Необходимые действия могут сильно отличаться от выпуска к выпуску, в зависимости от глубины изменений, приносимых в новом профиле.

В простейшем случае, вам просто потребуется изменить символьную ссылку **/etc/make.profile**, а в худшем — придется перекомпилировать всю систему, тем временем изображая танец вуду. Процесс перехода обычно описывается в замечаниях к выпуску. Кроме того, есть инструкции в конце данного руководства.

Поддерживаемые профили

Вы можете увидеть список профилей на Вашей архитектуре, официально поддерживаемой Gentoo когда установите утилиту **eselect (emerge eselect)** и затем выполните следующую команду:

Листинг 2.1 Просмотр поддерживаемых профилей

```
# eselect profile list
```

3. Инструкции по обновлению профилей

Общие инструкции.

Важно: Удостоверьтесь в том, что Ваши порты обновлены перед выполнением любых изменений с профилями !

Первое, выполните **emerge eselect**. Утилита **eselect** позволяет Вам просматривать и выбирать профили без необходимости создания или удаления символических ссылок вручную.

Листинг 3.1: Выбор профиля с помощью утилиты **eselect**

(Просмотр доступных профилей)

```
# eselect profile list
```

(Выберите номер Вашего желаемого профиля из списка)

```
# eselect profile set <номер профиля>
```

Если Вы предпочитаете изменять профили вручную, тогда просто сделайте следующее:

Листинг 3.2 Изменение профилей вручную.

```
# rm /etc/make.profile
```

```
# ln -s ../usr/portage/profiles/<выбранный профиль> /etc/make.profile
```

Примечание: Для большинства архитектур существуют два типа подпрофилей: **desktop** и **server**. Внимательно изучите эти профили, так как они могут

удовлетворять Вашим нуждам лучше, чем чрезвычайно минимальные профили по умолчанию.

Обновление до 2007.0 или 2006.1

Если Вы намереваетесь обновиться до уровня этих профилей, то Вам следовало бы знать, что они ожидают что по умолчанию в системе установлена локаль Unicode. Или более точно, должна быть определена строка UNICODE="yes" в файле **/etc/rc.conf**. Для этоо Вы должны создать Unicode локаль для Вашей системы. Пожалуйста, прочитайте UTF-8 guide чтобы узнать как создать правильную локаль.

Альтернативно, если Вы не желаете устанавливать локаль, тогда Вы можете определить UNICODE="no" в файле **/etc/rc.conf**. И переустановить (re-emerge) пакет **baselayout** (или ждите следующего обновления этого пакета) с USE флагом: -unicode. Вы можете установить -unicode только для baselayout или Вы можете установить его глобально для всех пакетов, добавив Вашу переменную USE в файл

/etc/make.conf.

Листинг 3.3 Опционально: удаление поддержки Unicode.

(Для удаления поддержки Unicode только для baselayout)

```
# echo "sys-apps/baselayout -unicode" >> /etc/portage/package.use
# emerge -a baselayout
```

(Для удаления поддержки Unicode для всей системы)

```
# nano -w /etc/make.conf
USE="-unicode"
# emerge -a baselayout
```

Примечание: Если Вы обновляетесь до 2007.0 на архитектуре SPARC, тогда вы должны следовать [GCC Upgrading Guide](#)

HOWTO Полное обновление системы

Ссылка на оригинал:

http://ru.gentoo-wiki.com/HOWTO_Полное_обновление_системы

С версии: 1.2

Приведение в порядок /var/lib/portage/world

В **world** должен быть список программ, которые нужно доустановить к тем, которые уже входят в "system" (т.е. в текущий профайл).

- в world не должно быть никаких библиотек, и т.д., которые не нужны сами по себе, а нужны только для удовлетворения чьих-то зависимостей (чтобы не продолжать устанавливать/обновлять их, если они уже станут не нужны по какой-то причине) ;
- программ, которые уже входят в "**system**", не должно быть в **world**
- в world нельзя указывать определенную версию софта, это лучше делать в **/etc/portage/package.mask** ;
- скрипт **regenworld** может помочь восстановить world путем анализа **/var/log/emerge.log** и генерации на его базе файла world (он перезапишет текущий world!) ;
- скрипт **dep -p -w** (входящий в состав пакета **udept**) поможет найти избыточные записи в **world**(которые всё-равно нужны другим записям в world или входят в **system**);
- перед серьёзными обновлениями желательно просмотреть **/etc/portage/***, т.к. там могут быть уже не актуальные записи мешающие текущему обновлению.

Обновление profile

- Не каждый Gentoo release включает в себя новый profile (например, 2004.1 был без profile).
- Даже если новый profile есть, то переходить на него не обязательно (если это будет обязательно, то старый профайл будет "протестующий" (deprecated) и emerge об этом должен будет громко кричать).
- Инструкции по обновлению profile будут выкладываться здесь: <http://www.gentoo.org/doc/ru/gentoo-upgrading.xml> и как правило сводиться к изменению симлинка **/etc/make.profile**

USE-флаги

- Запустить **emerge -uDpv --newuse world** и проверить что USE-флаги для всех пакетов выставлены корректно, и при необходимости скорректировать

USE-флаги выставляются в **/etc/make.conf** и **/etc/portage/package.use**

Запуск обновления системы (если не нужно обновлять toolchain)

- Если

```
emerge -puDav --newuse world
```

показывает что будет обновляться пакет входящий в toolchain (linux-headers, glibc, binutils или gcc), то крайне рекомендуется полностью перекомпилировать всю систему - см. следующий пункт - а иначе можно вместо следующего пункта просто запустить:

```
emerge -uDav --newuse world
```

Некоторые причины не использовать emerge -U вместо -u

- Причина 1: Проблемы со **SLOT**

Это, к примеру, происходит потому, что некоторые люди хотели **gimp-2** вместо **gimp-1.2**. Представьте ситуацию, где **gimp-1.2** помечен stable и находится в SLOT 1, **gimp-2** помечен unstable и находится в SLOT 2. Теперь при выполнении **ACCEPT_KEYWORDS=~x86 emerge gimp** получите **gimp-2**.

Позже, когда вы посчитаете, что наступило время обновить свою систему чем-либо похожим на "emerge -U world", эта команда установит gimp-1.2, потому, что gimp находится в world-файле, и флаг "-U" не обрабатывает SLOT должным образом.

- Причина 2: Проблемы, в случае удаления ebuild-ов с Portage-дерева.

Допустим, в Portage находятся 2 версии пакетов foo, foo-1.4 (помеченный как stable) и foo-1.6 (помеченный как unstable). Вы хотите вариант unstable и делаете emerge, как с вышеуказанным gimp. Позже обновляете world как было сказано выше, но в промежутке этого времени вышло критическое обновление для foo-1.6 - foo-1.6.1. Теперь появляется несколько возможностей обработки.

1. foo-1.6 был удален из Portage. Будет установлен foo-1.4, несмотря на "снижение" версии вместо флага "-U"
2. Ситуация будет еще хуже, если foo-1.6 не был удалён из Portage по какой-либо причине: foo-1.6 (тот, что с критической уязвимостью) будет оставаться на вашей системе до тех пор, пока не будет помечено stable что-либо выше чем foo-1.6.

Предупреждение: Не рекомендуется использовать **ACCEPT_KEYWORDS=~x86 emerge foo** о чем можно почитать здесь <http://www.gentoo.org/doc/ru/gentoo-amd64-faq.xml#keyword>

Обновление одного из пакетов входящих в toolchain

- Если обновляется хотя-бы один из **linux-headers**, **glibc**, **binutils** или **gcc**, то рекомендуется пересобрать их дважды, после чего весь **system**, после чего весь **world**.

Примечание: Цель двойной компиляции **toolchain** - получить гарантированно стабильный и корректный **toolchain** не зависящий от предыдущего. Перекомпилировать **system/world** после этого жёсткой необходимости нет, по крайней мере если остальной софт продолжает работать (возможно даже используя библиотеки из старого **toolchain** - см. предыдущие пункты об апгрейде).

Цель перекомпиляции **system/world** - чтобы весь софт получил потенциальное преимущество от установки нового **toolchain**. **system** перекомпилируется перед **world** из тех-же соображений, т.к. при

компиляции программ из **world** используются утилиты из **system**.

- Если увеличивается первая или вторая цифра версии **gcc**, то перед второй сборкой нужно переключиться на новую версию через **gcc-config** - иначе новый **gcc** просто установится параллельно со старым в "новый слот", но по умолчанию использоваться будет старый.
- При сборке **system** после двойной перекомпиляции **toolchain** нет необходимости опять компилировать **toolchain** как часть **system**. Аналогично при сборке **world** после **system** нет необходимости опять компилировать пакеты из **system** как часть **world**. Это можно попробовать обойти либо вручную, либо используя скрипты [\[1\]](#), либо через бинарные пакеты и **emerge -k** (я предпочитаю последний вариант).
- Итак, рекомендованный набор команд:

Листинг 1. Рекомендованный набор команд

```
# для того, чтобы безопасно использовать `emerge -k` нужно очистить
# каталог с текущими бинарными пакетами
# (напр., переместить его в /tmp/portage-packages)
```

```
pkgdir=$(portageq pkgdir)
mv $pkgdir /tmp/portage-packages1
install -d -o portage -g portage $pkgdir
```

```
# первая сборка toolchain
emerge linux-headers glibc binutils gcc-config gcc
```

```
# выбрать новый gcc если он установился в новый слот
gcc-config имя_или_номер_нового_gcc
# см. `gcc-config -l`
source /etc/profile
```

```
# компиляция toolchain с созданием бинарных пакетов
emerge -b glibc binutils gcc portage
```

```
# не компилировать glibc, binutils и gcc
emerge -bke system
```

```
# не компилировать предыдущие пакеты (включая system)
emerge -bke world
```

Примечание: Чисто теоретически существует пакет **binutils-config**, который когда-нибудь может потребоваться использовать аналогично **gcc-config**.

Обновление безопасности.

Примечание: Даже после **emerge -uDav --newuse world** в системе могут оставаться устаревшие пакеты с дырами в безопасности - в слотах!

```
glsa-check -l | grep '[N\]'
emerge ... # если нужно
```

Удаление неиспользуемых пакетов.

После обновления системы в ней могут оказаться пакеты, которые никто не

использует. Эти пакеты желательно удалить, т.к. они не будут в дальнейшем обновляться при ``emerge -uDav --newuse world``.

`emerge -a depclean` # **очень осторожно!!!**

- После обновления библиотек может потребоваться перекомпилировать программы, которые эти библиотеки используют:

Примечание: Для `glsa-check`, `revdep-rebuild` необходимо установить пакет `gentoolkit`

```
rm /root/.revdep-rebuild*.*_*
revdep-rebuild -p
revdep-rebuild
```

Обновление конфигов.

`dispatch-conf`

- Если используется `runit-init` и обновлялся пакет `baselayout`, то нужно восстановить `/sbin/init`:

```
ls -l /sbin/*init*
if (/sbin/init это бинарник, а не симлинк) {
mv /sbin/init /sbin/init-sysv
ln -s runit-init /sbin/init
}
```

- Отслеживание важных сообщений при установке пакетов.

В процессе `emerge world` выдаётся очень много сообщений, причём важные комментарии перемешаны с командами компиляции, и отследить их при сборке нескольких пакетов одновременно невозможно.

Но все эти сообщения можно получить из log-файлов после окончания установки **`emerge world`**. Для этого нужно использовать либо **`enotice`**, либо **`portlog-info`**.

Руководство Gentoo Linux по обновлению GCC

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/gcc-upgrading.xml>

С версии: 1.0

1. Введение

Обновление GCC

Зачем нужно обновлять? Ну, GCC довольно похож на любой другой пакет в вашей системе, но чуточку более важен. GCC следует обновлять всякий раз, когда в новой версии исправляются какие-нибудь раздражающие вас ошибки, добавляются новые нужные функции, или если вы хотите держать свою систему обновленной. Если ни один из этих случаев к вам не относится, обновление можно спокойно откладывать, пока ваша версия GCC поддерживается разработчиками Gentoo.

Если вы устанавливаете новую версию GCC, система не переключается на ее использование автоматически. Вам необходимо явно запросить изменение, потому что в процессе перехода может потребоваться несколько дополнительных шагов. Если вы решите не переключаться, система Portage продолжит использовать более старую версию компилятора, пока вы не передумаете или пока не удалите старый компилятор из своей системы.

В этом руководстве описываются необходимые шаги, нужные для полноценного обновления компилятора, используемого вашей системой Gentoo. Отдельный раздел посвящен переходу с GCC 3.3 на 3.4 или более новые версии и проблемам с libstdc++. Другой частный раздел предназначен пользователям, впервые устанавливающим Gentoo из архива третьей стадии (stage3), после выхода новой версии GCC.

Примечание: Необходимо заметить, что обновление с GCC-3.4 до GCC-4.0 или более нового не требует существенных изменений от пользователя, так как GCC-3.4 и GCC-4.0 используют одинаковый двоичный прикладной интерфейс (ABI). Все что нужно, это использовать gcc-config, чтобы выбрать желаемый компилятор.

2. Общие указания по обновлению

Введение

Важно: Если вы ищете подробные указания по обновлению с GCC-3.3 на GCC-3.4 или более новый, обратитесь к соответствующему разделу.

Важно: Если вы ищете подробные указания по обновлению GCC на вновь установленных системах, обратитесь к соответствующему разделу.

Вообще говоря, переход на версии с исправленными ошибками (bugfix release), как с 3.3.5 на 3.3.6, должен быть довольно безопасен: надо только установить новую версию, переключиться на нее и пересобрать единственный затрагиваемый пакет — libtool. Однако, при некоторых обновлениях GCC нарушается двоичная совместимость, в таких случаях может потребоваться пересборка не только затрагиваемых пакетов, но и даже всего системного набора и пакетов, необходимых для компиляции.

Говоря о необходимости ручного переключения на новую версию компилятора, мы сказали, что оно не происходит автоматически. Тем не менее, есть одно исключение — переход на версию с исправленными ошибками (как с 3.3.5 на 3.3.6), если не используется режим «multislot», позволяющий обеим версиям сосуществовать в одной системе. По умолчанию этот режим выключен, ведь большинству пользователей он ничего не даст.

Листинг 2.1: Обновление GCC

```
# emerge -uav gcc
```

(Вместо "i686-pc-linux-gnu-3.4.5" укажите обновленную версию GCC и настройки CHOST)

```
# gcc-config i686-pc-linux-gnu-3.4.5
```

```
# source /etc/profile
```

(Пересборка libtool)

```
# emerge --oneshot -av libtool
```

Теперь пересоберите набор программ для компиляции, затем world, используя новый компилятор.

Листинг 2.2: Пересборка системы

```
# emerge -eav system
```

```
# emerge -eav world
```

Теперь можно без опасений удалить старую версию GCC. Если вы чувствуете такую необходимость, введите следующую команду (как обычно, вместо =sys-devel/gcc-3.3* укажите версию, которую собираетесь удалить):

Листинг 2.3: Удаление более старой версии GCC

```
# emerge -aC =sys-devel/gcc-3.3*
```

3. Переход с GCC-3.3 на 3.4 или более новый

Введение

Переход с GCC-3.3 на 3.4 или более новую версию не так гладок, ведь между этими версиями изменился двоичный прикладной интерфейс C++ (ABI). Также придется позаботиться о существующей проблеме с библиотекой libstdc++.

Варианты

Важно: Если вы обновляете GCC на машине SPARC, вам придется выбрать путь полной пересборки системы из-за некоторых внутренних изменений двоичного интерфейса (ABI) GCC в области передачи параметров функций.

У вас есть два варианта обновления системы. Первый способ быстрее и требует использования программы revdep-rebuild из пакета gentoolkit, а во втором вся система пересобирается с нуля, чтобы задействовать новые возможности GCC. Ваше дело, какой из способов выбрать. В большинстве случаев первого способа достаточно.

Использование revdep-rebuild

Если вы выбрали этот способ, нужно сначала установить gentoolkit, если вы еще этого не сделали. Затем обновите GCC и переключите систему на новый компилятор. Также пересоберите пакет libtool, чтобы обеспечить пригодность программ для компиляции.

Листинг 3.1: Установка gentoolkit и обновление GCC

```
# emerge -an gentoolkit
# emerge -uav gcc
(вместо "i686-pc-linux-gnu-3.4.5" укажите обновленную
 версию GCC и настройки CHOST)
# gcc-config i686-pc-linux-gnu-3.4.5
# source /etc/profile
```

```
(пересборка libtool)
# emerge --oneshot -av libtool
```

Теперь посмотрите, какие пакеты собирается пересобирать revdep-rebuild. Затем запустите revdep-rebuild на собственно пересборку. Это займет некоторое время, так что потерпите.

Листинг 3.2: Использование revdep-rebuild

```
# revdep-rebuild --library libstdc++.so.5 -- -p -v
# revdep-rebuild --library libstdc++.so.5
```

Примечание: Возможно, у вас появятся проблемы с несуществующими версиями пакетов из-за того, что они устарели или замаскированы. В этом случае можно при запуске revdep-rebuild указать параметр --package-names. Это заставит пакеты пересобирааться, основываясь на именах пакетов, вместо точного имени и версии.

Для обеспечения совместимости с более старыми двоичными приложениями C++ и любыми пакетами, которые revdep-rebuild мог пропустить, надо установить пакет

sys-libs/libstdc++-v3 до того, как удалять GCC 3.3 из своей системы.

Листинг 3.3: Установка libstdc++-v3 и удаление GCC

```
# emerge --oneshot sys-libs/libstdc++-v3
# emerge -aC =sys-devel/gcc-3.3*
```

Использование emerge -e

Этот гораздо более медленный способ пересобирает всю систему, гарантируя, что все пересобирается новым компилятором, и, таким образом, он безопаснее. Сперва потребуется обновить GCC и libtool, и переключить систему на новый компилятор.

Листинг 3.4: Обновление GCC

```
# emerge -uav gcc
(вместо "i686-pc-linux-gnu-3.4.5" укажите обновленную
 версию GCC и настройки CHOST)
# gcc-config i686-pc-linux-gnu-3.4.5
# source /etc/profile
```

(пересборка libtool)

```
# emerge --oneshot -av libtool
```

Для обеспечения совместимости с более старыми двоичными приложениями C++, надо установить в систему пакет sys-libs/libstdc++-v3.

Листинг 3.5: Установка libstdc++-v3

```
# emerge --oneshot sys-libs/libstdc++-v3
```

Теперь займемся пересборкой сначала пакетов system, а затем world. Это займет очень длительное время, зависящее от количества установленных пакетов: ведь будут пересобираются все средства компиляции и поддерживающие системные файлы, а затем каждый пакет, находящийся в вашей системе. Это необходимо, чтобы все пакеты были гарантированно скомпилированы уже с новыми средствами компиляции, включая сами эти средства.

Листинг 3.6: Пересборка system и world

```
# emerge -e system
# emerge -e world
```

Теперь можно удалить старые версии GCC, не вызывая проблем:

Листинг 3.7: Очистка

```
# emerge -aC =sys-devel/gcc-3.3*
```

4. Обновление GCC на новой установке

Введение

Обновление GCC в системе после установки из архива третьей стадии (stage3) — дело простое. Отсутствие избытка установленных программ, которые ссылаются на старую версию GCC — это преимущество пользователей, только что установивших систему. В следующем примере показывается обновление с GCC-3.3 на 3.4 или более новые версии. При обновлении с других версий GCC будут кое-какие отличия. Например, имена библиотек, используемые ниже для `revdep-rebuild`, относятся к версии GCC 3.3, также, как и необходимость установки `libstdc++-v3`.

Пока пользователь не внес каких-либо изменений в систему, для получения системы с новой версией GCC нужно всего несколько шагов. Также, как и при обновлении с GCC-3.3 до 3.4, у пользователя есть две возможности. Но, в отличие от обновления с GCC-3.3 до 3.4, здесь обновление проще, так как различий между способами меньше. Первый способ быстрее, и задействует программу `revdep-rebuild` из пакета `gentoolkit`, подобно вышеописанной процедуре обновления. Использование `revdep-rebuild` предполагает пересборку только тех пакетов, которые действительно ссылаются на библиотеки GCC, тогда как при втором способе система полностью перекомпилируется с новой версией GCC, что занимает намного больше времени. Второй способ никогда не потребуется, и описан только для полноты картины.

Приведенные первые шаги одинаковы для обоих способов, и должны делаться в любом случае.

Листинг 4.1: Обновление GCC

```
# emerge -uav gcc
(вместо "i686-pc-linux-gnu-3.4.5" укажите обновленную
 версию GCC и настройки CHOST)
# gcc-config i686-pc-linux-gnu-3.4.5
# source /etc/profile
```

```
(пересборка libtool)
# emerge --oneshot -av libtool
```

Для обеспечения совместимости с более старыми двоичными приложениями C++, надо установить в систему пакет `sys-libs/libstdc++-v3`.

Листинг 4.2: Установка `libstdc++-v3`

```
# emerge --oneshot sys-libs/libstdc++-v3
```

Использование `revdep-rebuild`

Этот способ требует, чтобы вы сначала установили пакет `gentoolkit`, если это еще не сделано. Затем запустите программу `revdep-rebuild`, чтобы просканировать установленные пакеты, найти и пересобрать нужные.

Листинг 4.3: Установка gentoolkit и запуск revdep-rebuild

```
# emerge -an gentoolkit
# revdep-rebuild --library libstdc++.so.5 -- -p -v
# revdep-rebuild --library libstdc++.so.5
```

Примечание: Возможно, у вас появятся проблемы с несуществующими версиями пакетов из-за того, что они устарели или замаскированы. В этом случае можно при запуске revdep-rebuild указать параметр --package-names. Это заставит пакеты пересобираться, основываясь на именах пакетов, вместо точного имени и версии.

Использование emerge -e

Этот способ, будучи гораздо медленнее, пересобирает всю систему, гарантируя, что все пересобрано новым компилятором. Это необязательно, но допустимо, если вы также изменяете переменную среды CFLAGS или другие переменные make.conf, которые влияют на компиляцию системы.

Так как эти действия выполняются сразу после первоначальной установки, не потребуется перекомпилировать world, как мы поступили бы для обновления компилятора в ранее установленной системе. Однако, для пущей уверенности в том, что обновлены все пакеты, можно запустить обновление цели world вместо system.

Листинг 4.4: Пересборка system

```
# emerge -e system
```

Очистка

Теперь можно удалить старые версии GCC, не вызывая проблем. Выражение ВАША-НОВАЯ-ВЕРСИЯ-GCC замените номером версии, на которую вы перешли:

Листинг 4.5: Очистка

```
# emerge -aC "<sys-devel/gcc-ВАША-НОВАЯ-ВЕРСИЯ-GCC"
```

5. Обычные грабли

Важно на время обновления отключить distcc. Смешение версий компилятора на разных узлах вызовет проблемы при сборке. Это не относится к ccache, так как объекты кэша будут в любом случае сделаны действительными.

Всегда используйте одну и ту же версию GCC для своего ядра и дополнительных модулей ядра. Как только вы пересоберете world с новым GCC, внешние модули (например, app-emulation/qemu-softmmu) не смогут загрузиться. Пожалуйста, чтобы это исправить, пересоберите свое ядро новой версией GCC.

Если вы обновляете GCC на машине SPARC, обязательно еще раз запустите silo -

f после пересборки world, чтобы избежать возможных проблем.

Распространенные сообщения об ошибках

Если ваша система жалуется на что-то вроде `libtool: link: `/usr/lib/gcc-lib/i686-pc-linux-gnu/3.3.6/libstdc++.la' is not a valid libtool archive`, запустите `/sbin/fix_libtool_files.sh 3.3.6` (замените «3.3.6» на номер версии из сообщения об ошибке).

Если вы увидели `error: /usr/bin/gcc-config: line 632: /etc/env.d/gcc/i686-pc-linux-gnu-3.3.5: No such file or directory`, тогда попробуйте удалить `/etc/env.d/gcc/config-i686-pc-linux-gnu` и снова запустить `gcc-config`, следом указав `source /etc/profile`. Делайте это только тогда, когда у вас не настроены никакие кросс-компиляторы.

Если при `emerge -e system` или `emerge -e world` не собирается пакет, выполнение можно продолжить командой `emerge --resume`. Если ошибка появляется снова и снова, пропустите пакет, указав `emerge --resume --skipfirst`. Не запускайте параллельно другие экземпляры `emerge`, чтобы не потерять информацию, нужную для возобновления.

Если во время обновления компиляторов встретится ошибка `spec failure: unrecognized spec option`, попробуйте откатиться на свой компилятор по умолчанию, убрать переменную `GCC_SPECS` и снова обновить компилятор:

Листинг 5.1: Восстановление первичной настройки компилятора

```
# gcc-config 1
# source /etc/profile
# unset GCC_SPECS
# emerge -uav gcc
```

Gentoo и X.Org

Описание настройки МОНОЛИТНОГО X-сервера версий 6.X

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/xorg-config.xml>

Дата: 13.06.07

С версии: 1.0

1. Что такое X Window Server?

Графический интерфейс против командной строки

Обычного пользователя могут испугать мысли о необходимости печатать в командной строке. Почему бы ему не прокладывать свой путь на территории свободы выбора Gentoo (и Linux вообще), просто «водя мышкой»? Ну, *большая улыбка*, конечно, вы сможете это сделать :-). В Linux предлагается широкое разнообразие пользовательских интерфейсов и сред, которые можно установить поверх установленной у вас системы.

Это один из самых больших сюрпризов, с которым сталкиваются новички: графический интерфейс пользователя — это всего лишь прикладная программа, выполняемая в системе. Он не является частью ядра Linux или каких-либо других компонентов системы. Это — мощный инструмент, позволяющий полностью задействовать графические возможности вашего компьютера.

Так как стандартизация очень важна, для организации прорисовки и перемещения окон по экрану, взаимодействия пользователя с мышью и клавиатурой, а также других важных базовых аспектов, был создан стандарт под названием X Window System, сокращённо X11 или просто X. Он используется в Unix, Linux и других Unix-подобных операционных системах во всем мире.

Приложение, которое дает пользователям Linux возможность запускать графические интерфейсы пользователя и использует стандарт X11, называется Xorg-X11, ответвление проекта XFree86. Проектом XFree86 было решено использовать лицензию, возможно, несовместимую с лицензией GPL, поэтому рекомендуется использовать Xorg. Пакет XFree86 больше не поддерживается в официальном дереве портежей.

Проект X.org

Проектом X.org создана и поддерживается бесплатно распространяемая реализация системы X11 с открытым исходным кодом. Это открытая инфраструктура рабочей станции, основанная на X11.

Xorg обеспечивает интерфейс между устройствами вашего компьютера и нужными вам графическими программами. Кроме того, Xorg является полностью сетевым, то есть вы можете запускать программу на одной системе, а отображать ее — на другой.

2. Установка Xorg

Использование emerge

Хватит болтовни, не перейти ли нам к делу? Чтобы установить **Xorg**, вам просто нужно запустить **emerge xorg-x11**. Установка **Xorg** займёт достаточно времени, так что вы можете перекусить, дожидаясь ее завершения.

Перед тем, как установить Xorg, в файле `/etc/make.conf` вам понадобится настроить две очень важные переменные.

Первая из них — **VIDEO_CARDS**, которая используется для выбора драйвера для используемой вами видеокарты, и связана с ее типом и маркой. Одним из распространенных значений является **nvidia** для карт nVidia или **fglrx** для карт ATI Radeon. Это — закрытые драйверы, предоставленные nVidia и ATI. Если вы хотите использовать версии с открытым исходным кодом, используйте **nv** вместо **nvidia**, но помните, эта версия не обеспечивает 3D-ускорение. Аналогично, используйте **radeon** для видеокарт от ATI. В **VIDEO_CARDS** можно указывать несколько драйверов, разделенных пробелами.

Вторая переменная, **INPUT_DEVICES**, используется для указания драйверов устройств ввода. В большинстве случаев связки `keyboard mouse` достаточно.

Теперь нужно решить, какие драйверы использовать, и добавить нужные параметры в файл `/etc/make.conf`:

Листинг 2.1: Пример строк в `make.conf`
 (для поддержки мыши и клавиатуры)
`INPUT_DEVICES="keyboard mouse"`
 (для видеокарт nVidia)
`VIDEO_CARDS="nvidia"`
 (ИЛИ для видеокарт ATI Radeon)
`VIDEO_CARDS="fglrx"`

Дополнительные сведения по настройке карт nVidia и ATI находятся в руководстве по видеокартам nVidia в Gentoo Linux и Gentoo Linux ATI FAQ. Если вы не знаете, какие драйверы выбрать, ознакомьтесь с этими руководствами.

Примечание: Если у вас рекомендуемые настройки не работают, следует запустить `emerge -pv xorg-x11`, проверить все доступные параметры и указать соответствующие вашей системе. Для различных версий Xorg на различных архитектурах эти переменные могут выводиться командой `emerge -pv xorg-server` вместо приведенной выше. Например, для архитектуры x86 и `xorg-x11-7.0`:

Листинг 2.2: Просмотр всех доступных параметров драйверов
emerge -pv xorg-x11

These are the packages that would be merged, in order:

```
Calculating dependencies... done!  
[ebuild R ] x11-base/xorg-x11-7.0-r1 USE="-3dfx" INPUT_DEVICES="keyboard  
mouse -acecad -aiptek -calcomp -citron -digitaledge -dmc -dynapro -elo2300  
-elographics -evdev -fpit -hyperpen -jamstudio -joystick -magellan -magictouch  
-microtouch -mutouch -palmax -penmount -spaceorb -summa -synaptics -tek4957  
-ur98 -vmmouse -void -wacom" VIDEO_CARDS="nvidia -apm -ark -chips -cirrus -cyrix  
-dummy -fbdev -fglrx -glint -i128 -i740 -i810 -imstt -mach64 -mga -neomagic -nsc  
-nv -r128 -radeon -rendition -s3 -s3virge -savage -siliconmotion -sis -sisusb  
-tdfx -tga -trident -tseng -v4l -vesa -vga -via -vmware -voodoo" 0 kB
```

После установки всех необходимых переменных можно перейти к установке пакета Xorg.

Листинг 2.3: Установка Xorg
emerge xorg-x11

После завершения установки, чтобы продолжить, вам наверняка придётся обновить некоторые переменные среды. Для установки значений просто запустите `env-update`, а затем — **`source /etc/profile`**.

Листинг 2.4: Обновление переменных среды
env-update
source /etc/profile

3. Настройка Xorg

Файл `xorg.conf`

Конфигурационный файл Xorg называется `xorg.conf` и находится в папке `/etc/X11`. В пакет Xorg-X11 входит его пример под названием `/etc/X11/xorg.conf.example`, который можно использовать при создании своей собственной конфигурации. Он подробно прокомментирован, но если вы нуждаетесь в подробном описании синтаксиса, не стесняйтесь обратиться к страницам справки:

Листинг 3.1: Чтение страницы справки `xorg.conf`
man 5 xorg.conf

Счастливого чтения всем желающим. Мы, конечно же, не из их числа, поэтому перейдем к выяснению, как можно создать файл автоматически.

По умолчанию: автоматическая генерация `xorg.conf`

Xorg способен самостоятельно подобрать большинство параметров за вас. Скорее всего, вам потребуется всего лишь изменить несколько строк, чтобы установить желаемое разрешение экрана. Если вы заинтересованы в более глубокой настройке, обязательно просмотрите ресурсы, указанные в конце этой главы. Но сначала давайте создадим (надеюсь, работающий) конфигурационный файл Xorg.

Листинг 3.2: Генерация файла `xorg.conf`
`# Xorg -configure`

Обязательно прочтите последние строки, выводимые после завершения опроса оборудования Xorg. Если вы увидите, что где-то Xorg не удалось правильно опознать устройства, то вам придётся править файл `xorg.conf` вручную. Если же всё прошло гладко, Xorg должен сообщить вам, что создан файл и готов для тестирования файл `/root/xorg.conf.new`. Давайте его испытаем :)

Листинг 3.3: Тестирование файла `xorg.conf.new`
`# X -config /root/xorg.conf.new`

Если всё в порядке, вы увидите чёрно-белый узор. Проверьте, работает ли мышь, и подходит ли разрешение экрана. Вы вряд ли точно угадаете разрешение, но всё же заметите слишком низкое. Выйти можно в любой момент, нажав комбинацию клавиш `Ctrl+Alt+Backspace`.

Альтернатива: полуавтоматическая генерация `xorg.conf`

В Xorg есть утилита `xorgconfig`, задающая различные вопросы о вашей системе (о графическом адаптере, клавиатуре и т.п.). Основываясь на ваших ответах, она создаст файл `xorg.conf`.

Листинг 3.4: Полуавтоматическая генерация `xorg.conf`
`# xorgconfig`

Другая утилита, также входящая в пакет Xorg — `xorgcfg`, которая сначала пытается выполнить `Xorg -configure`, а затем запускает X-сервер для более тонкой настройки.

Листинг 3.5: Использование `xorgcfg`
`# xorgcfg`
 (если X даст сбой, или настройка завершится неудачей, попробуйте:)
`# xorgcfg -textmode`

Копирование в `xorg.conf`

Теперь давайте скопируем `xorg.conf.new` в `/etc/X11/xorg.conf`, чтобы не приходилось постоянно запускать `Xorg -config`: набирать просто `X` или `startx` гораздо легче :)

Листинг 3.6: Копирование поверх `xorg.conf`
`# cp /root/xorg.conf.new /etc/X11/xorg.conf`

Использование `startx`

Теперь попробуйте ввести **`startx`**, чтобы запустить свой X-сервер. **`startx`** — это сценарий, запускающий сеанс X, то есть серверы X, а поверх них — некоторые графические приложения. Он решает, какие приложения запустить, исходя из следующей логики:

если в домашнем каталоге есть файл с именем **`.xinitrc`**, то выполняются команды, перечисленные в нём

в противном случае считывается значение переменной `XSESSION` и запускается один из указанных в `/etc/X11/Sessions/` сеансов (указать значение `XSESSION` по умолчанию, для всех пользователей системы, можно в файле `/etc/rc.conf`)

если вышеуказанное завершилось неудачей, производится откат к простейшему диспетчеру окон, обычно `twm`.

Листинг 3.7: Запуск X
`# startx`

Если вы увидели уродливый, отвратительный, омерзительный, безобразный диспетчер окон, то это — `twm`. Чтобы завершить сеанс `twm`, наберите `exit` или нажмите `Ctrl-D` в одном из терминалов `xterm`. «убить» сеанс X также можно, нажав комбинацию клавиш `Ctrl+Alt+Backspace`, чтобы «выгнать его с позором» — наверно, такой вариант годится не всегда. Хотя это не причинит тяжёлого вреда :)

4. Настройка `xorg.conf`

Установка разрешения экрана

Если вы чувствуете, что разрешение экрана неподходящее, вам потребуется проверить два раздела конфигурации. Прежде всего, разделе **`Screen`**, где перечисляются варианты разрешения экрана, с которыми может запускаться X-сервер. По умолчанию в этом разделе может вообще не быть никаких строк о разрешении экрана. В таком случае **`Xorg`** оценивает допустимое разрешение на основе данных из другого раздела: **`Monitor`**.

При этом `Xorg` для вычисления правильных вариантов разрешения использует значения **`HorizSync`** (частота строк) и **`VertRefresh`** (частота кадров) из раздела **`Monitor`**. Пока что оставьте эти параметры как есть. Лишь в том случае, когда изменения в разделе `Screen` (которые мы опишем чуть ниже) не помогают, вам придется заглянуть в технические характеристики своего монитора и указать правильные значения. Также можно воспользоваться программой, определяющей технические характеристики вашего монитора, например, `sys-apps/ddcinfo-`

knoppix.

Предупреждение: Не меняйте «просто так» значения этих двух относящихся к монитору переменных, не сверившись с техническими характеристиками своего монитора. Неправильные значения могут привести в лучшем случае к срыву синхронизации, а в худшем случае — к сгоревшему монитору.

Теперь давайте поменяем значения разрешения. В следующем примере, взятом из `/etc/X11/xorg.conf`, мы добавим строчки `Modes` (режимы) и `DefaultDepth` (цветность), чтобы X-сервер по умолчанию запускался в режиме 24 бит при 1024x768 разрешении экрана. Особо не обращайтесь внимания на значения — это просто пример и, скорее всего, они будут отличаться от настроек вашей системы.

Листинг 4.1: Изменение раздела `Screen` `/etc/X11/xorg.conf`

```
Section "Screen"
    Identifier "Default Screen"
    Device     "S3 Inc. ProSavage KN133 [Twister K]"
    Monitor    "Generic Monitor"
    DefaultDepth 24
    # несколько строк пропущены для наглядности
    SubSection "Display"
        Depth 24
        Modes  "1024x768"
    EndSubSection
EndSection
```

Запустите X (`startx`), чтобы обнаружить, что сервер использует желаемое разрешение :)

Настройка клавиатуры

Чтобы настроить X на использование национальных раскладок, найдите раздел `InputDevice` (устройство ввода), определяющий настройки клавиатуры, и добавьте параметр `XkbLayout` с указанием необходимой раскладки. Для примера, покажем, как добавить бельгийскую раскладку. Просто измените код страны на свой:

Листинг 4.2: Changing the keyboard layout

```
Section "InputDevice"
    Identifier "Generic Keyboard"
    Driver     "keyboard"
    Option     "CoreKeyboard"
    Option     "XkbRules" "xorg"
    Option     "XkbModel" "pc105"
    Option     "XkbLayout" "be"
EndSection
```

Настройка мыши

Если ваша мышь не работает, сначала придётся выяснить, обнаружена ли она вообще ядром. Мыши (в качестве устройств) выглядят как `/dev/input/mouse0` (или `/dev/input/mice`, если вы хотите использовать несколько мышек). В некоторых

случаях используется название `/dev/psaux`. В любом случае, вы можете убедиться, что устройство соответствует вашей мыши, просмотрев вывод соответствующего файла устройства, одновременно передвигая мышь. В большинстве случаев на экране должна появиться бессмыслица. Для остановки вывода нажимайте `Ctrl-C`.

Листинг 4.3: Проверка файла устройства

```
# cat /dev/input/mouse0
```

(не забудьте нажать `Ctrl-C` для завершения)

Если ваша мышь не обнаружена, проверьте, все ли необходимые модули загружены.

Если же ваша мышь найдена, впишите устройства в соответствующий раздел `InputDevice`. В следующем примере видно, как мы устанавливаем еще два параметра: `Protocol` (определяет протокол, используемый мышью; у большинства пользователей — `PS/2` или `IMPS/2`) и `ZAxisMapping` (позволяющий задействовать колесико, если есть).

Листинг 4.4: Изменение настройки мыши в `Xorg`

```
Section "InputDevice"
    Identifier "TouchPad Mouse"
    Driver "mouse"
    Option "CorePointer"
    Option "Device" "/dev/input/mouse0"
    Option "Protocol" "IMPS/2"
    Option "ZAxisMapping" "4 5"
EndSection
```

Запустите **startx** и порадитесь результату :). Поздравляем, теперь (надеемся) у вас есть работающий `Xorg`. Следующий шаг — удалить этот ужасный облегченный диспетчер окон, и использовать «нафаршированный» возможностями диспетчер окон (или даже рабочую среду), такой как `KDE` или `GNOME`, но это не входит в рамки данного руководства :)

5. Ресурсы

Создание и детальная настройка `xorg.conf`

Прежде всего, в **man 5 xorg.conf** дается краткое, но полное описание синтаксиса конфигурационного файла. Обязательно держите эту страницу открытой в соседнем терминале при редактировании своего конфигурационного файла!

Второй источник информации на вашей системе — каталог `/usr/X11R6/lib/X11/doc`, где находятся различные описания `README` для конкретных графических чипсетов.

В сети также есть много сведений о настройке `xorg.conf`. Мы перечислим лишь некоторые из них; обязательно погуглите, чтобы найти другие :). Так как в большинстве параметров `xorg.conf` и `XF86Config` (конфигурационного файла для XFree86) используется одинаковый синтаксис, а информации о XFree86 больше, мы ссылаемся и на такие ресурсы.

The XFree Local Multi-User HOWTO

An Introduction to XFree 4.x Криса Хаузера (Chris Houser)

Другие источники

Если вы хотите обновить свою систему со старым монолитным Xorg на новый модульный Xorg 7, прочтите руководство по переходу на модульный X (англ.)

Дальнейшая информация по настройке различных пакетов, для работы в среде X находится в разделе графическая среда Gentoo нашей документации.

Описание настройки МОДУЛЬНОГО X-сервера версий 7.X (Англ.)

Ссылка на оригинал: http://gentoo-wiki.com/HOWTO_Modular_Xorg

С версии: 1.0

Introduction

This is a guide to Modular Xorg, mainly dealing with special circumstances and troubleshooting.

This article is currently being maintained by AllenJB. If you have suggestions for additions or fixes, please post them to the discussion and bugs page.

Please see also the official Modular Xorg Upgrade Guide.

Preparation

Firstly it'll be useful to know what version of Xorg you're going to be emerging. If you don't know, run **emerge -p xorg-x11** to find out.

Nvidia Drivers

The old nvidia-glx and nvidia-kernel packages have been replaced by a single nvidia-drivers package. If you're still using the old packages, you'll want to upgrade to the new package first, as the old ones will not work with modular Xorg, by running:

```
# emerge -Ca nvidia-glx nvidia-kernel
# emerge nvidia-drivers
#eselect opengl set nvidia
```

Masking Xorg 7.1 (optional)

If your system is going to install xorg-x11-7.1 and you wish to use xorg-x11-7.0 instead you'll need to add the following to /etc/portage/package.mask (create it if it doesn't exist):

```
# Mask xorg 7.1
>=x11-base/xorg-x11-7.1
>=x11-base/xorg-server-1.1.0
>=x11-drivers/xf86-input-evdev-1.1.2-r1
>=x11-drivers/xf86-video-vesa-1.2.0
>=x11-drivers/xf86-input-mouse-1.1.1
>=x11-drivers/xf86-input-keyboard-1.1.0
>=x11-drivers/xf86-video-tdfx-1.2.0
```


Backup

Because Xorg is such a key part of a desktop system, you'll want to backup your current install before continuing. To do this we're going to use the quickpkg command from the gentoolkit package:

```
emerge -a gentoolkit
quickpkg xorg-x11
```

If you want to find out more about the quickpkg tool, run: `man quickpkg`

Additionally, create a list of packages that have files in the Xorg specific directories by running:

```
if [[ ! -L /usr/X11R6 ]]; \
then equery belongs /usr/X11R6 > ~/usr-x11r6-packages \
&& rm -rf /usr/X11R6; fi
```

Restoring the backup

To restore your backed-up package run: `emerge --usepkgonly -a xorg-x11-6.8.2-r8`

If you can't remember the exact version you can find it out by running: `ls /usr/portage/packages/All/xorg-x11*`

You then want to re-emerge all the packages listed in `~/usr-x11r6-packages`.

Uninstall old X.org

Warning: If you are reading this from the same machine you will be upgrading X on, keep in mind two things:

After cleaning out the old X, and before the new X is installed, you will not be able to start any X applications as X is not technically installed. The applications depending on X currently running are just running off the memory of X. Closing them will not enable you to re-open them until the new X is installed.

Programs currently running in X may become unstable during the install. For example, firefox may unexpectedly crash due to not being able to find fonts which are no longer there, or other similar reasons.

It is hence highly recommended that you print out a copy of this guide (or save a tree and open this guide on a console (not in X)) before you begin. Your programs should go back to behaving normally once the install is complete, however you may want to restart X as soon as possible afterwards.

Uninstall your current xorg install by running: **`emerge -Ca xorg-x11`**

/usr/X11R6 symlink

Run **ls -l /usr/X11*** and check that **/usr/X11R6** is a symlink to **/usr**. It should look something like this

```
:  
lrwxrwxrwx 1 root root 6 Mar 27 2005 /usr/X11R6 -> ../usr
```

If it doesn't, delete it and recreate it by running: **cd /usr && ln -s ../usr X11R6**

Remove **/usr/lib/X11/xkb**

The following is a requirement of the **xkeyboard-config** package.

32-bit Users: Run **ls -l /usr/lib/X11/xkb** and if it exists, delete it with **rm -r /usr/lib/X11/xkb**

Emerge Modular X.org

Device Drivers

Modular Xorg introduces a new system for managing the driver packages for your devices. You need to set the appropriate **VIDEO_CARDS** and **INPUT_DEVICES** flags in **/etc/make.conf**. To see what flags are available, run **emerge -upv xorg-x11**. On a typical machine with an Nvidia video card, it might look like this:

File: /etc/make.conf:

```
INPUT_DEVICES="keyboard mouse"  
VIDEO_CARDS="nv nvidia vesa"
```

The **nv** driver is the **open source** unofficial driver that lacks 3D acceleration. The **nvidia** driver is the **official nvidia-drivers package**.

On an ATI System, the **VIDEO_CARDS** line should be like this (unless you don't wish to use proprietary drivers **fglrx**):

File: /etc/make.conf

```
VIDEO_CARDS="radeon fglrx vesa"
```

Add the **dri USE** flag to **/etc/make.conf** to enable **Direct Rendering support**, which most graphics drivers use.

Installation

Now run **emerge -Dvat xorg-x11** to install modular Xorg.

A few useful X11 tools that used to be included in the 6.8 build are now their own packages, you can emerge all of them now by running: **emerge -DuNav xev xdpinfo xvinfo xset xdriinfo xprop**

If you would like some extra fonts, you can emerge them now also:

```
emerge -DuNav font-adobe-100dpi font-adobe-75dpi font-adobe-utopia-100dpi \
font-adobe-utopia-75dpi font-bh-100dpi font-bh-75dpi font-bh-type1 \
font-bh-lucidatypewriter-100dpi font-bh-lucidatypewriter-75dpi \
font-bitstream-100dpi font-bitstream-75dpi ttf-bitstream-vera \
corefonts sharefonts freefonts font-ibm-type1
```

Keyboard Mappings

There may be people who will search for the `ca_enhanced` keyboard mapping in Xorg 7.0. It has not disappeared, it just changed names to `ca(fr)`, as with this keyboard configuration:

File: /etc/X11/xorg.conf:

```
Section "InputDevice"
    Identifier "Keyboard1"
    Driver      "kbd"
    Option "AutoRepeat" "500 30"
    Option "XkbRules" "xorg"
    Option "XkbModel" "pc105"
    Option "XkbLayout" "ca(fr)"
EndSection
```

Notice that Option "XkbRules" "xfree86" won't work for modular xorg-x11. You have to change it to "xorg", or many keys will not produce the desired effect.

In case you defined a custom xkb model (e.g. if you added a diNovo keyboard), check if you modified `/usr/share/X11/xkb` and not `/usr/lib/X11/xkb`.

If your X server is restarting each time you pressed a non dead key, add the following link:

```
# ln -s /usr/share/X11/XKeysymDB /usr/lib/X11/XKeysymDB
```

OpenMotif fails to emerge

If building of openmotif fails run: **emerge -DuNav --oneshot printproto xbitmaps libXp**

Then restart the emerge using: **emerge --resume**

GL Issues

Problems building **xorg-server**, **glx**, and other **GL-related** packages in modular X may be due to a multilib issue.

The first thing to try is to update `eselect-opengl` to at least 1.0.3.

Next check your `opengl` implementation with `eselect opengl list`. If `xorg-x11` isn't the only option, you probably want the alternate option, so change it with: `eselect opengl set <implementation>`

xorg-server tries to build every single input driver

Solution: You need to specify which drivers you want by modifying `/etc/make.conf` to reflect your hardware. If nothing is specified, it will build them all!

Example: File: `/etc/make.conf`
`INPUT_DEVICES="keyboard mouse synaptics evdev"`

Mouse Speed Uncontrollable with evdev Driver

Please see [TIP Setting Mouse Speed Through Command Line](#) for more information.

Mouse Wheel No Longer Works

Please see the Modular XOrg section of [HOWTO Advanced Mouse](#) for more extensive information.

Xorg won't start due to fglrx or nvidia module missing

Re-emerge the appropriate drivers (`ati-drivers` or `nvidia-drivers`) to get X running again after the install. Make sure you recompile the the drivers with same compiler you used for the kernel, otherwise you'll get message about the module format being incompatible when you try to load the `fglrx` kernel module.

Matrox G550, Xinerama and mga driver

The install script of the binary drivers available from Matrox (v4.4 at the time of writing) copies the driver files to
`/usr/lib/modules/drivers`

instead of

/usr/lib/xorg/modules/drivers

Copy the correct files (mga_drv.so/mga_hal_drv.so) manually to get Xinerama working again.

Problems with the keyboard

The AltGr key and/or other national special keys may stop working after an update to modular X. To fix this, unmerge the new xkeyboard-config and re-emerge the xkbdata package which contains correct translations for that key.

If the above fix doesn't work, you can also try running: **xmodmap -e "keycode 113 = Mode_switch"**

Sometimes it can happen that GNOME / KDE keyboard layout switching doesn't work. To correct this, create xkb as a symbolic link in the /usr/lib64/X11 lib directory with:

```
cd /usr/lib64/X11  
ln -s /usr/share/X11/xkb/ xkb
```

Missing fonts

There are X11 fonts that may be in use but not available. See the Xorg and fonts HOWTO and install the listed fonts.

Configuration file locations have changed

All of the configuration files and scripts in Xorg-6.8 were stored in /etc/X11, which was not standard. In modular Xorg, all of the configuration files are in the same place but configuration scripts and default files have moved to /usr/lib/X11/ and /usr/share/X11. When upgrading, since /etc is config-protected, your old configuration files would not have been removed -- this means you'll have some extra files in there that look right but don't do anything.

Also, these new locations are NOT config-protected, so when making changes to say, app-defaults/XTerm-color, be sure to add CONFIG_PROTECT="/usr/share/X11/app-defaults" to /etc/make.conf. Or when updating the init for xdm, it would be a good idea to copy that file from /usr/lib/X11/xdm/Xsetup_0 to /etc/X11/xdm/ and update /etc/X11/xdm/xdm-config accordingly.

Other problems and solutions

Please see Gentoo's bug tracker and the official modular X howto.

If none of the above helps generating a new xorg.conf may solve your problems.
X -configure

XkbOptions

Ссылка на оригинал: <http://ru.gentoo-wiki.com/XkbOptions>

С версии: 1.0

Обновлено: 1.4

Параметры "XkbOptions"

Варианты переключателей между языками в файле `/etc/X11/xorg.conf`

1. **grp:toggle** - переключение правым Alt;
2. **grp:shift_toggle** - двумя клавишами shift;
3. **grp:ctrl_shift_toggle** - ctrl+shift;
4. **grp:alt_shift_toggle** - alt+shift;
5. **grp:ctrl_alt_toggle** - ctrl+alt;
6. **grp:caps_toggle** - переключение по CapsLock;
7. **grp:lwin_toggle** - переключение по левой "Win" клавише на Windows клавиатуре (для X Window старше 4.2);
8. **grp:rwin_toggle** - переключение по правой "Win" клавише на Windows клавиатуре (для X Window старше 4.2);
9. **grp:menu_toggle** - переключение по клавише "Контекстное меню" на Windows клавиатуре (для X Window старше 4.2);
10. **grp:lctrl_toggle** переключение по левой клавише Control
11. **grp:rctrl_toggle** переключение по правой клавише Control
12. **grp:switch** - переключение по правому Alt только на момент нажатия
13. **grp:lwin_switch** - переключение по левой клавише Windows только на момент нажатия
14. **grp:rwin_switch** - переключение по правой клавише Windows только на момент нажатия
15. **grp:win_switch** - переключение по любой клавише Windows только на момент нажатия

Возможные значения grp_led:

1. **grp_led** - использовать индикаторы на клавиатуре для показания изменения группы
2. **grp_led:num** - индикатор Num_Lock показывает изменения группы
3. **grp_led:caps** - индикатор Caps_Lock показывает изменение группы
4. **grp_led:scroll** - индикатор Scroll_Lock показывает изменение группы

Пример

Файл: `/etc/X11/xorg.conf`

```
...
Section "InputDevice"
    ...
    Option "XkbOptions" "grp:switch,grp:caps_toggle,grp_led:scroll"
    ...
EndSection
...
```

Дополнительная информация:

/etc/X11/xkb/rules/xorg.lst

ВАЖНО (от JohnBat26): При использовании библиотек HAL версии $\geq 0.5.10$, evdev версии $\geq 1.2.0$ и kkbswitch, возникают проблемы с переключением раскладок. [Ссылка на bug в bugs.gentoo.org](https://bugs.gentoo.org). Это происходит из-за того, что теперь X.org запрашивает эти параметры у HAL!! Следовательно вся конфигурация переключения раскладок перемещается в каталог ***/etc/hal/fdi/policy***. Таким образом, Вам необходимо создать файл такого содержания:

Листинг 1: Конфигурация раскладок через HAL

```
<?xml version="1.0" encoding="ISO-8859-1"?> <!-- -*- SGML -*- -->
<match key="info.capabilities" contains="input.keyboard">
  <merge key="input.x11_driver" type="string">keyboard</merge>
  <merge key="input.xkb.layout" type="string">us,ru</merge>
  <merge key="input.xkb.model" type="string">pc101</merge>
  <merge key="input.xkb.rules" type="string">xorg</merge>
  <merge key="input.xkb.variant" type="string">,winkeys</merge>
  <merge key="input.xkb.options" type="string">grp:shift_toggle,grp_led:scroll</merge>
</match>
```

назвать его как-нибудь, например **keyboard_layout**, и положить в указанную выше папку. После этого, раскладка клавиатуры должна переключаться нормально. ;)

Также смотрите: [HOWTO hal и устройства ввода](#)

HOWTO Xorg and Fonts (Рус.)

Ссылка на оригинал: http://gentoo-wiki.com/HOWTO_Xorg_and_Fonts

Перевод: [Батогов Е.В. а.к.а. JohnBat26](#)

С версии: 1.3

Дата окончания перевода: 27.10.2007

Назначение этого HOWTO

После изменения названия X сервера с Xfree на X.org, многие люди (включая меня) столкнулись с проблемами, связанными с отрисовкой и сглаживанием шрифтов, особенно это касается тех, кто использует LCD или TFT мониторы. Этот HOWTO делает шрифты значительно более интересными. В дополнение к этому, этот HOWTO объясняет как надо устанавливать шрифты.

Общие сведения

Компьютерные шрифты в целом, и в Linux (X11) в частности, это довольно запутанное дело. Существует много различных форматов шрифтов, в диапазоне от старых растровых шрифтов до шрифтов **TrueType**. Помимо этого, много шрифтов имеют проблемы лицензирования и внутренние проблемы (на которые компания Apple, к счастью, пока не обращает внимания). А если Вы захотите включить сглаживание, то это только добавит путаницы.

Многие программы, особенно **ghostscript**, **mozilla**, **openoffice**, и **tex**, имеют свои собственные представления о шрифтах и их настройке. Это значит, что информация в этом how-to может быть не эффективна для того, чтобы Ваши шрифты заработали.

Смотрите: X11 имеет не одну подсистему шрифтов, а две! Старая "core X font subsystem" использует наименования типа: -misc-fixed-medium-r-normal--10-100-75-75-c-60-iso8859-1. Новая система шрифтов, также известная как "fontconfig", работает с библиотекой **Xft** и, как обещается, полностью заменит старую систему. К тому же, существует ещё и сервер шрифтов, называющийся "xfs".

Замечание по поводу DPI

До настройки любого программного обеспечения, лучшее из того, что Вы можете сделать для получения хорошо выглядящих шрифтов на Linux, это использование LCD/CRT монитора по крайней мере не меньше, чем с 96 dpi (точек на дюйм). Для расчёта этого значения, возьмите горизонтальное

разрешение и разделите его на ширину TFT/LCD экрана в дюймах. Любое значение меньше, чем 96 dpi приведет к значительным проблемам которые нельзя будет устранить. CRT мониторы будут почти всегда отрисовывать шрифты лучше, чем LCD мониторы потому, что они имеют внутреннее сглаживание от сканирования электронного луча и экрана покрытого люминофором.

Замечание по поводу сравнения шрифтов на Windows/Mac и Linux

Шрифты в Windows (clear type) и в Mac OS всегда будут выглядеть лучше, чем шрифты в Linux, когда LCD монитор имеет dpi меньше, чем 96. Когда dpi равняется 96 и более, тогда шрифты в Linux будут выглядеть также или лучше, чем в Windows или Mac OS. Конечно, все это верно, если вы установите пропатченный libXft, доступный в 'xeffects' overlay.

Получение необходимых пакетов

Я предполагаю, что Вы уже правильно установили X.org (X.org версии 6.8.0-r4 и выше должен быть установлен с включенными USE флагами: **truetype** и **type1**). Я также полагаю, что Вы не запускаете X-сервер для множества клиентов и, следовательно, USE флаг: font-server выключен, хотя он может по-прежнему применяться в других случаях.

Необходимые пакеты:

- * x11-base/xorg-x11
- * media-libs/freetype
- * media-fonts/corefonts
- * media-fonts/freefonts

Тем не менее я советую Вам также установить следующие полезные шрифты (**замечание:** media-fonts/unifont блокируют dev-util/eclipse-sdk):

- * media-fonts/ttf-bitstream-vera
- * media-fonts/artwiz-fonts
- * media-fonts/sharefonts
- * media-fonts/terminus-font
- * media-fonts/unifont
- * media-fonts/dejavu

Замечание: f28: Более красивые шрифты: media-fonts/liberation-fonts-ttf, "A GPL-2 Helvetica/Times/Courier заменяют TrueType font set, с уважением к Red Hat"

Активизируйте USE флаги и установите следующие пакеты:

```
# euse --enable truetype type1
```

```
# emerge --newuse --noreplace freetype corefonts freefonts artwiz-fonts sharefonts \
terminus-font ttf-bitstream-vera unifont dejavu xorg-x11
```

Для тех, кто уже установил Xorg и желает обновить все остальные пакеты, которые уже установлены, я рекомендую изменить файл **/etc/make.conf** добавив следующие USE флаги:

```
USE="truetype type1"
```

Затем необходимо заново собрать все пакеты которые могут использовать эти флаги:

```
# emerge -avuDN world
```

Замечание для Xorg 7.0: Кажется Вам необходимо установить ещё и пакет "intlfonts".

```
# emerge intlfonts
```

Вы можете также увидеть это в обсуждении:

<http://forums.gentoo.org/viewtopic.php?p=3643250#3643250>

Информация о хинтинге (Hinting)

Хинтинг является частью процесса отрисовки шрифта, в общем, хинтинг руководит растеризатором так, что он производит шрифт, гораздо лучше выглядящий на экране (особенно при малых размерах шрифта).

Отдавая должное возможным проблемам с патентами, FreeType имеет выбор — собраться с TrueType's Byte Code Interpreter (или BCI) или нет.

Без BCI (по умолчанию), FreeType использует свой собственный автоматический хинтер для улучшения качества отображения шрифтов малых размеров. Этот автохинтер ещё находится в стадии совершенствования и улучшается с каждой версией FreeType. Автохинтер обычно дает великолепные результаты. Однако, многие считают, что родное TrueType BCI производит лучшие результаты, чем автохинтер, т.к. в этом случае используется информация о хинтинге, содержащаяся в шрифте.

Если Вы живете в стране, где патенты на программное обеспечение не являются проблемой, Вы можете включить использование BCI, выключая USE флаг: **bindist**. Дополнительную информацию о freetype и патентах смотрите на: <http://freetype.sourceforge.net/patents.html>.

```
#echo "media-libs/freetype -bindist" >> /etc/portage/package.use
#emerge -av freetype
```

Если Вы считаете, что шрифты слишком размытые на Вашем LCD мониторе,

попытайтесь включить флаг **bindist** и посмотреть разницу, может быть это будет лучше в Вашем случае.

Изменение конфигурационных файлов

После установки шрифтов, Вам необходимо изменить конфигурационные файлы в порядке распознавания их системой. Вот эти файлы:

1. `/etc/fonts/local.conf`;
2. `/etc/X11/xorg.conf`;
3. `~/.fonts.conf` .

Для проверки того, какие шрифты установлены, Вы можете использовать **fc-list** и **xlsfonts**. Для просмотра шрифтов, Вы можете использовать **gfontview**.

Что делать, если Ваши шрифты не показываются утилитой **fc-list**, что делать если Ваши шрифты не отображаются в **xlsfonts**, что делать если Ваши шрифты не появляются в **OpenOffice**. Что, наконец, делать, если шрифты не показываются в **ghostscript** и/или в **tex**:

Смотрите файлы: `/etc/fonts/local.conf` и `~/.fonts.conf`

Редактируйте конфигурационные файлы для системы шрифтов: **Xft**. Изменения в файле `/etc/fonts/local.conf` касаются всей системы; выборочные изменения в пользовательском конфигурационном файле `~/.fonts.conf` имеют приоритет над системными установками в большинстве случаев. Единственная опция, которая не может быть перекрыта в пользовательском файле — это настройка сглаживания (anti-aliasing setting).

В файле `/etc/fonts/local.conf` нет необходимости устанавливать все директории со шрифтами, кроме тех случаев, когда шрифты устанавливаются в нестандартный каталог. Файл `/etc/fonts/fonts.conf` включает `/usr/share/fonts` и некоторые другие директории. Другими словами, Вы не должны изменять этот файл, т.к. Xorg будет просматривать Ваш файл `~/.fonts.conf`, если он имеется.

Заметьте то, что файлы `/etc/fonts/local.conf` и `~/.fonts.conf`, являются XML-файлами. И их легко изменять в редакторах, которые поддерживают синтаксис XML .

Следующий пример содержит множество опций, которые Вы можете захотеть использовать:

Файл: `/etc/fonts/local.conf` или `~/.fonts.conf`:

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<!-- /etc/fonts/local.conf file to configure system font access -->

<fontconfig>
```

```

<!-- Enable sub-pixel rendering -->

<!--
    <match target="font">
        <test qual="all" name="rgba">
            <const>unknown</const>
        </test>
        <edit name="rgba" mode="assign"><const>rgb</const></edit>
    </match>
-->

<!-- Use the Autohinter -->

    <match target="font">
        <edit name="autohint" mode="assign"><bool>true</bool></edit>
    </match>

<!-- Disable Autohinting for bold fonts -->

    <match target="font">
        <test name="weight" compare="more">
            <const>medium</const>
        </test>
        <edit name="autohint" mode="assign"><bool>false</bool></edit>
    </match>

<!-- Exclude/Include a range of fonts for Anti Aliasing -->

<!--
<match target="font">
    <test qual="any" name="size" compare="more">
        <double>9</double>
    </test>
    <test qual="any" name="size" compare="less">
        <double>14</double>
    </test>
    <edit name="antialias" mode="assign">
        <bool>true</bool>
    </edit>
</match>
-->

<!-- And/Or disable Anti Aliasing for a range on pixel-based size.
    Disabling this using both methods seems to fix Firefox. -->

<!--
<match target="font">
    <test compare="less" name="pixelsize" qual="any">
        <double>20</double>
    </test>
    <edit mode="assign" name="antialias">
        <bool>false</bool>
    </edit>
</match>

```

```
-->

<dir>/home/johnbat26/extrafonts</dir>

</fontconfig>
```

В представленном выше файле опция: "sub-pixel rendering" закомментирована потому, что она используется только на LCD мониторах. Если её включить на CRT мониторе, то это заставит Xorg производить много дополнительной работы по отрисовке. Возможные значения: "**none**", "**rgb**", "**bgr**", "**vrgb**" и "**vbgr**". Для большинства LCD мониторов наиболее подходящим значением будет: "**rgb**", которое и использует приведенный выше пример.

Вы конечно захотите использовать какую-нибудь форму хинтинга. Gentoo поддерживает 2 формы хинтинга:

- **AutoHinter**

По умолчанию используется auto hinter из пакета FreeType, и это также является установкой по умолчанию в приведенном выше примере конфигурационного файла. Тем не менее, когда используется автохинтинг, тогда Вы можете захотеть выключить его использование на жирных шрифтах, иначе они могут стать слишком широкими. Пример выше как раз и делает это.

- **TrueType Byte Code Interpreter (BCI)**

Если Вы собрали FreeType с активацией интерпретатора байт-кода (TrueType Byte Code Interpreter) (см. выше), и желаете использовать BCI, тогда выключите auto hinter полностью. (Убедитесь, что `<edit name="autohint" mode="assign"><bool>true</bool></edit>` использует "false" вместо "true". Существуют патентные проблемы на использование TrueType BCI; если Вы желаете избежать их полностью, тогда не используйте BCI.

И KDE и GNOME имеют прекрасный графический интерфейс, в котором пользователь может выбрать уровень хинтинга: "легкий", "средний", "сильный" ('light,' 'medium,' and 'strong'). Это позволяет обратиться к FreeType 'auto hinter', а не к BCI. Другими словами 'легкий,' 'средний,' и 'сильный' хинтинг, позволяют всем шрифтам выглядеть одинаково, когда BCI используется вместо автохинтера (auto hinter).

Вторая опция с окончания файла, также закомментирована, так как некоторые люди предпочитают исключать или включать набор размеров шрифтов для применения сглаживания к ним. Это общий способ, чтобы сделать Ваши шрифты выглядящими очень похоже на шрифты по умолчанию в Windows(R). Когда Вы отключаете сглаживание, убедитесь также, что выключена опция "sub-pixel rendering", - для достижения наилучших шрифтов без сглаживания.

Наконец, последняя опция, связанный с пикселом, размер шрифта (pixel-based

font sizes)... Пожалуйста, читайте комментарии к это опции!

Последняя строка в примере пользовательских шрифтов, это изменение директории по умолчанию (/usr/share/fonts/) для местоположения шрифтов. Скорее всего Вам не нужна данная строка, если так, то можете просто удалить её.

Помните, что все изменения в этом файле распространяются на всю систему и будут перекрыты ~/.fonts.conf, исключая только установку anti-aliasing setting, которая не может быть перекрыта ,если установлена в этом файле. Также заметьте, что синтаксис файла: /etc/fonts/local.conf одинаков с локальным пользовательским конфигурационным файлом: ~/.fonts.conf, и, следовательно, Вы можете копировать глобальный файл, и просто переименовать его.

/etc/X11/xorg.conf

Когда Вы настраиваете X ("иксы"), важно убедиться, что они правильно понимают размер Вашего экрана, и так же важно, что Вы запускаете LCD монитор на его родном разрешении, которое используется для вычисления горизонтального и вертикального DPI экрана — шрифты в GTK+ приложениях кажутся особенно безобразными на основе неправильной информации о экране.

Это может также привести к неправильному определению размера шрифтов в различных приложениях. Некоторые шрифты будут корректного размера в зависимости от используемого метода, а другие будут иметь неправильные размеры. Я заметил большие различия, когда просматривал страницы в Firefox. Если на странице использовался CSS, то она отображалась хорошо, иначе если использовался чистый HTML, то шрифты получаются либо огромные либо небольшие в зависимости от направления неправильной установки DPI. Я также обнаружил, что DPI 96 является хорошим размером.

Установка ширины и высоты экрана (в миллиметрах) используя следующую инструкцию:

DisplaySize <width> <height>

в секции **Monitor** файла: **/etc/X11/xorg.conf**, решает эту проблему.

File: /etc/X11/xorg.conf

```
Section "Monitor"
    Identifier      "Monitor0"
    :
    DisplaySize     340 270
    :
EndSection
```

Простейший путь для измерения ширины и высоты— это воспользоваться линейкой

Например, монитор в 15 дюймов имеет размеры приблизительно 340мм x 270 мм,

а 19 дюймов — 380мм x 305 мм.

Значения `DisplaySize` могут быть вычислены по формуле **$25.4 \times \text{width} / \text{dpi}$** или, соответственно, **$25.4 \times \text{height} / \text{dpi}$** , 25,4 мм на дюйм. Например, если использовать

DPI равным 75 на разрешении 1600x1200, то расчет будет следующий: **$25.4 \times 1600 / 75 = 542$** и **$25.4 \times 1200 / 75 = 406$** , и в результате устанавливаем **`DisplaySize 542 406`**.

Другой способ для подсчёта этих значений, это попросить X сервер вычислить их для Вас. Запустите его используя команду **`startx -- -dpi 96`**, где **96** это требуемое значение для DPI, и проверьте размер:

```
#xdpyinfo | grep dimension
dimensions: 1400x1050 pixels (XXXxYYY millimeters)
```

Используйте эти значения: XXX и YYY для параметра **`DisplaySize`**.

Ещё один способ обнаружить размерность экрана, если Ваше аппаратное обеспечение предоставляет эту информацию (Вам может потребоваться установить **`emerge -tav xrandr`** сначала) это использовать:

```
# xrandr --query
SZ: Pixels      Physical      Refresh
*0 1680 x 1050  ( 431mm x 272mm ) *60
Current rotation - normal
Current reflection - none
Rotations possible - normal
Reflections possible - none
```

Я достиг лучших результатов, используя измерения, полученные с помощью этого калькулятора: <http://www.raydreams.com/prog/dpi.aspx>

Если вы используете драйверы nVidia, то наилучшим способом следовало бы добавить эти опции в секцию `device`. Это также намного лучше работает с двумя мониторами:

Файл: `/etc/X11/xorg.conf`

```
....
Option "UseEdidDpi" "FALSE"
Option "DPI" "96 x 96"
....
```

Далее, у нас есть список наших каталогов со шрифтами в файле **`/etc/X11/xorg.conf`**.

Пути к шрифтам ищутся в том порядке, в котором они расположены в списке, поэтому мы должны сделать небольшие изменения.

Достаточно много современных приложений и окружений рабочего стола, таких как Gnome и KDE используют утилиту **`fontconfig`**, так что только шрифты, которые действительно требуются X сервером (или сервером шрифтов) это `fixed` и `cursor`.

/usr/share/fonts/misc:unscaled

Первым путем, который мы хотим видеть в нашем списке, является: **/usr/share/fonts/misc**. Там находятся все важные **fixed** и **cursor** шрифты. Они являются критичными для правильной работы **Xorg**. Там же находится курсор по умолчанию, и помещение его первым в списке сохраняет X от использования других курсоров, которые устанавливают некоторые пакеты шрифтов (artwiz)

Опция: **:unscaled** говорит X использовать **unscaled** версии этих шрифтов всякий раз, когда возможно предотвратить pixelation. Эта опция может применяться для любого bitmap шрифта

/usr/share/fonts/75dpi:unscaled
/usr/share/fonts/100dpi:unscaled

Это основные bitmap шрифты. Мы обычно хотим загружать шрифты для 75dpi перед 100dpi, так как большинство тем ожидают этого. Тем не менее, большинство современных мониторов имеют более высокое значение dpi, так что если Вы обнаружите у себя косоглазие читая меню, Вы скорее всего захотите поэкспериментировать с переключением порядка в списке шрифтов

/usr/share/fonts/Type1
/usr/share/fonts/TTF

Type1 и TrueType шрифты. Порядок не имеет большого значения, но заметьте что оба пакета содержат Luxi Sans/Serif/Mono шрифты.

/usr/share/fonts/corefonts
/usr/share/fonts/freefonts
/usr/share/fonts/sharefonts

это Microsoft's TrueType core fonts, и набор TrueType шрифтов и коллекция свободных Type1 Fonts, соответственно.

/usr/share/fonts/artwiz
/usr/share/fonts/terminus
/usr/share/fonts/ttf-bitstream-vera
/usr/share/fonts/unifont

Если Вы установили пакеты упомянутые выше, то Вам следовало бы добавить эти пути в список шрифтов.

/usr/share/fonts/local
/usr/share/fonts/cyrillic

эти шрифты обычно не нужны и вероятно пустые или не существуют. Для гарантии, проверьте файл font.dir внутри каждой директории. Если этот файл пуст, тогда этот каталог нет необходимости добавлять в список путей к шрифтам.

/usr/share/fonts/*everythingelse*

все директории новых шрифтов, которые Вы установили. Большинство из них

уникальны. Если Вы обеспокоены о каком-нибудь отдельном шрифте, поместите его наверх списка. Вы можете проверить список из доступных шрифтов, посмотрев файл **fonts.dir**, находящийся в директории шрифта.

Теперь сделаем несколько финальных изменений в плане порядка следования путей в списке. Возьмём основные bitmap шрифты и поместим их последними. Это делается потому, что у нас есть намного лучшие TTF или Type1 версии некоторых из этих шрифтов которые мы чаще используем. Также пакет **media-fonts/corefonts** содержит большое количество core fonts и его следовало бы поместить повыше в списке.

В результате список должен выглядеть подобно:

Файл: /etc/X11/xorg.conf (Список путей к шрифтам)

```
...
Section "Files"
    FontPath  "/usr/share/fonts/misc:unscaled"
    FontPath  "/usr/share/fonts/Type1/"
    FontPath  "/usr/share/fonts/TTF/"
    FontPath  "/usr/share/fonts/corefonts"
    FontPath  "/usr/share/fonts/freefonts"
    FontPath  "/usr/share/fonts/sharefonts"
    FontPath  "/usr/share/fonts/terminus"
    FontPath  "/usr/share/fonts/ttf-bitstream-vera"
    FontPath  "/usr/share/fonts/unifont"
    FontPath  "/usr/share/fonts/75dpi:unscaled"
    FontPath  "/usr/share/fonts/100dpi:unscaled"
    FontPath  "/usr/share/fonts/artwiz"
    FontPath  "/usr/share/fonts/cyrillic"
EndSection
...
```

Также проверьте в том же файле, что загружается модуль freetype

Файл: /etc/X11/xorg.conf

```
Load "freetype"
```

Перезагрузка служб

После изменения конфигурационных файлов, Вам необходимо перезагрузить Xorg. Если Вы используете X display manager, такой как GDM, KDM, XDM и т.д. тогда просто выйдите из Вашей сессии и перезагрузите X, нажав **"Ctrl+Alt+Backspace"**, находясь на экране входа в систему. Если это не работает, то рестарт Xorg возможен с помощью перезагрузки службы **xdm**.

Кто-то могут заметить подвисание системы, запуская следующие команды в Xorg, так что я советую запускать их в другом терминале, нажав "CTRL + ALT + F1" и войти в систему. (Нажатие ALT + F7 вернет Вас обратно в X из консоли.)

```
# /etc/init.d/xdm restart
```

После выполнения этой команды, Ваш X display manager будет перезагружен. Вновь войдя в систему, Вы должны получить шрифты со сглаживанием (anti-

aliased).

Mozilla Firefox and Thunderbird

Mozilla Firefox 1.0+

Для активизации шрифтов со сглаживанием в Firefox 1.0. убедитесь, что Вы собрали Firefox с выключенным USE флагом: **moznoxft**. Запустите утилиту **ufed** для этого (не забудьте установить её: **emerge ufed**).

Итак:

#emerge mozilla-firefox

Необходимо хотя бы один раз запустить Firefox, чтобы он создал профиль в :
~/.mozilla/firefox/default.xxx/.

Направляйтесь в **about:config** в Вашем браузере, или альтернативно, измените файл **user.js** в ~/.mozilla/firefox/default.xxx/, где **xxx** являются буквами или цифрами.

Firefox 1.0.4 сохраняет файл: **prefs.js** в ~/.mozilla/firefox/xxx.default/, где xxx это строка из цифр и букв.

Либо через диалог **about:config**, либо в файле user.js Вы должны вставить следующие строки, как показано ниже:

Внимание: Установка "font.FreeType2.printing" в true, может вызвать внезапное падение Firefox при попытке печати через.

Файл: ~/.mozilla/firefox/____.default/prefs.js

```
// TrueType
user_pref ("font.FreeType2.enable", true);
user_pref ("font.freetype2.shared-library", "libfreetype.so.6");
// if libfreetype was built without hinting compiled in
// it is best to leave hinting off
user_pref ("font.FreeType2.autohinted", true);
user_pref ("font.FreeType2.unhinted", true);
// below a certain pixel size anti-aliased fonts produce poor results
user_pref ("font.antialias.min", 0);
user_pref ("font.embedded_bitmaps.max", 1000000);
// sample user_pref s for TrueType font dirs
user_pref ("font.directory.truetype.1", "/usr/share/fonts/ttf-bitstream-vera");
user_pref ("font.directory.truetype.2", "/usr/share/fonts/TTF");
user_pref ("font.directory.truetype.3", "/usr/share/fonts/corefonts");
```

```
user_pref ("font.directory.truetype.4", "/usr/share/fonts/freefont");
user_pref ("font.FreeType2.printing", true);
```

Если Вы следуете этому HOWTO, тогда Вы не будете иметь проблем с перечисленными каталогами, иначе Вы можете изменить их.

В Firefox 1.0, следуйте в Edit » Preferences » General » Fonts & Colors.

В Firefox 1.5, следуйте в Edit » Preferences » Content » Fonts & Colors » Advanced.

Затем, установите эти параметры:

- * **Proportional: Serif (Size: 16)**
- * **Serif: Bitstream vera serif**
- * **Sans-serif: Bitstream vera sans**
- * **Monospace: Bitstream vera sans mono (Size: 12)**
- * **Display Resolution: System setting**

Также проверьте, что опция: "Всегда использовать мои: Шрифты" не выбрана.

Затем перезагрузите Firefox и направляйтесь на www.gentoo.org со сглаженными шрифтами!

Если Вы, используя KDE и Firefox, не получили сглаженных шрифтов после этого, тогда идите в Центр управления KDE / Внешний вид и темы / Шрифты, выключите сглаживание, нажмите применить, затем включите сглаживание и вновь нажмите применить.

Mozilla Thunderbird 0.7, 1.0.2 и выше

Существует очень мало опций в Thunderbird, которые связаны со шрифтами, таким образом все опции должны быть применены в user.js. Создайте этот файл в ~/.thunderbird/default/xxxxxxxx.slt/ (или, в новых версиях, ~/.thunderbird/xxxxxxxx.default/). В файле user.js Вы можете вставить те же строки, которые вставляли для Mozilla Firefox: заметьте только, что Вы вставляли в файл prefs.js ?

Файл: ~/.thunderbird/default/xxxxxxxx.slt/user.js

```
// TrueType
user_pref ("font.FreeType2.enable", true);
user_pref ("font.freetype2.shared-library", "libfreetype.so.6");
// if libfreetype was built without hinting compiled in
// it is best to leave hinting off
user_pref ("font.FreeType2.autohinted", true);
user_pref ("font.FreeType2.unhinted", true);
// below a certain pixel size anti-aliased fonts produce poor results
user_pref ("font.antialias.min", 0);
user_pref ("font.embedded_bitmaps.max", 1000000);
user_pref ("font.scale.tt_bitmap.dark_text.min", 0);
user_pref ("font.scale.tt_bitmap.dark_text.gain", "0.0");
// sample user_pref s for TrueType font dirs
user_pref ("font.directory.truetype.1", "/usr/share/fonts/ttf-bitstream-vera");
```

```
user_pref ("font.directory.truetype.2", "/usr/share/fonts/TTF");
user_pref ("font.directory.truetype.3", "/usr/share/fonts/corefonts");
user_pref ("font.directory.truetype.4", "/usr/share/fonts/freetype");
user_pref ("font.FreeType2.printing", true);
```

Затем, когда откроете Thunderbird 1.0.2, следуйте в : Edit » Preferences » Fonts Or (on 0.7) Tools » Options » Fonts и установите эти параметры:

Proportional: Serif (Size: 16)

Serif: Bits-bitstream vera serif

Sans-serif: Bits-bitstream vera sans

Monospace: Bits-bitstream vera sans mono (Size: 12)

Minimum font size: 9

Также проверьте, что выбрана опция: "Позволять сообщениям использовать другие шрифты".

Затем перезагрузите Thunderbird и читайте Вашу почту с новыми шрифтами!

Проблемы с размерами шрифтов

Если в Ваших приложениях из KDE шрифты имеют неправильный размер, и кроме этого, KDE не было установлено (т.е. Центр управления KDE недоступен :(), тогда Вам следует изменить Ваш файл `~/.kde/share/config/kdeglobals` следующим образом:

Файл: `~/.kde/share/config/kdeglobals`

```
[General]
StandardFont=Bitstream Vera Sans,7,-1,5,50,0,0,0,0,0
activeFont=Bitstream Vera Sans,7,-1,5,75,0,0,0,0,0
fixed=Bitstream Vera Sans Mono,7,-1,5,50,0,0,0,0,0
font=Bitstream Vera Sans,7,-1,5,50,0,0,0,0,0
menuFont=Bitstream Vera Sans,7,-1,5,50,0,0,0,0,0
taskbarFont=Bitstream Vera Sans,7,-1,5,50,1,0,0,0,0
toolBarFont=Bitstream Vera Sans,7,-1,5,50,0,0,0,0,0
```

Число после имени шрифта это его размер (Я не знаю, что означают другие числа)

Так получилось, что шрифты в меню Firefox получились огромными. Чтобы решить эту проблему просто добавьте следующее в файл:

Файл: `~/.mozilla/firefox/<yourprofile>/chrome/userChrome.css`

```
* {
  font-family: Sans-Serif !important;
  font-size: 8pt !important;
}
```

и сделайте тоже самое для Thunderbird
(~/.thunderbird/<profile>/chrome/userChrome.css)

В моем случае, после применения всех этих изменений, шрифты для полей ввода (кнопок, текстовых полей или выпадающих меню) на некоторых страницах подобным Google или Amazon были слишком большими и использовался один из ужасных шрифтов. Чтобы изменить это, следуйте в userContent.css (он отличается от файла userChrome.css, который мы только что изменяли), и добавьте следующее:

Файл: ~/.mozilla/firefox/<yourprofile>/chrome/userContent.css

```
input, textarea, select, button,
input[type="button"], input[type="reset"], input[type="submit"] {
    font-size: 8pt !important;
    font-family: Verdana !important;
}
```

Что делать, если Linux шрифт в Konsole не работает.

Пользователи KDE вероятно знакомы с Konsole — эмулятором терминала KDE. Но некоторые пользователи получают следующую ошибку когда они пытаются использовать Linux шрифты: **Font ``-misc-console-medium-r-normal--16-160-72-72-c-80-iso10646-1'` not found**

Вы можете решить эту проблему следующими действиями:

1. Запустите **kdesu kcmshell kcmfontinst**, когда Вы находитесь в KDE.
2. Введите пароль root.
3. Нажмите на иконку "add fonts icon" (an @-sign with a +-sign on it)
4. Направляйтесь в каталог **/usr/kde/3.2/share/fonts/** и добавьте доступные шрифты (files ending in .pcf.gz).

Что, если Ваши шрифты стали размытыми и не читаемыми

У меня шрифты временами становились не читаемыми в Gnome 2.12

Полное выключение сглаживания (AntiAliasing) (Убирание комментариев со следующих секций в файлах: **/etc/fonts/local.conf** и **~/.fonts.conf**, как они объявлены в примерах выше) решит эту проблему.

Если это не помогло и Вы используете драйвер **media-video/nvidia-kernel-1.0.8178-r3**,

попытайтесь убрать комментарий на опции **render acceleration** в секции Device файла **xorg.conf**:

```
Option "RenderAccel"
```

Ошибка при запуске startx

Если Вы получаете ошибку: "Fontconfig error" когда запускаете **startx**:

```
"Fontconfig warning: no <cachedir> elements found. Check configuration."  
"Fontconfig warning: adding <cachedir>/var/cache/fontconfig</cachedir>"  
"Fontconfig warning: adding <cachedir>~/.fontconfig</cachedir>"
```

Просто исправить её можно запуском:

```
emerge --noconfmem --oneshot fontconfig && dispatch-conf
```

XEmacs

Если после всей проделанной работы, Вы обнаружили, что не имеете хорошего шрифта для XEmacs (который не поддерживает сглаживание), тогда дополнительно установите следующее:

```
* x11fonts-jmk
```

```
emerge x11fonts-jmk
```

Затем добавьте ещё одну строку FontPath в Ваш файл: xorg.conf:

```
FontPath "/usr/share/fonts/jmk"
```

Перезагрузите подсистему шрифтов (просто выйдя из X). Затем попытайтесь установить Ваш Xemacs шрифт в **"Neep Alt"**. Вы вероятно хотите альтернативный вариант, вместо обычного, для отображения символа амперсанда.

Полезные ресурсы

Это ресурсы из которых я брал информацию, когда писал этот HOWTO:

- * [How to enable AA in Fluxbox, GKrellM, Abiword, Phoenix etc...](#)
- * [X.Org & Fonts](#)
- * [\[xfree\] caratteri sfocati](#)
- * [Moz Firefox 0.9 \(all versions\) general discussion thread.](#)

Смотрите также

- * [Control Font DPI in X](#)
- * [Forum Post](#)
- * [XFree86 Font De-uglification HOWTO](#)
- * [Optimal Use of Fonts on Linux](#)
- * [Mini HOWTO: Getting Perfect Fonts on Mandrake 10.1](#)

Примечание (от JohnBat26), мой файл: ~/.fonts.conf:

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<fontconfig>
<!-- Info at http://gentoo-wiki.com/HOWTO_Xorg_and_Fonts --><!-- Replace Courier with a better-looking
font -->
  <match target="pattern" name="family" >
    <test name="family" qual="any" >
      <string>Verdana</string>
    </test>
    <edit mode="assign" name="family" >
<!-- Other choices - Courier New, Luxi Mono    <string>Bitstream Vera Sans Mono</string> --><!-- Other
choices - Courier New, Luxi Mono -->
      <string>Verdana</string>
    </edit>
  </match>
  <match target="font" >
    <edit mode="assign" name="rgba" >
      <const>rgb</const>
    </edit>
    <edit mode="assign" name="autohint" >
      <bool>true</bool>
    </edit>
    <edit mode="assign" name="antialias" >
      <bool>true</bool>
    </edit>
    <edit mode="assign" name="hinting" >
      <bool>true</bool>
    </edit>
    <edit mode="assign" name="hintstyle" >
      <const>hintmedium</const>
    </edit>
  </match>
  <!-- Disable autohint for bold fonts, otherwise they look *too* bold --> <match target="font" >
    <test compare="more" name="weight" >
      <const>medium</const>
    </test>
    <edit mode="assign" name="autohint" >
      <bool>false</bool>
    </edit>
  </match>
  <!-- Reject bitmap fonts in favour of Truetype, Postscript, etc. --> <selectfont>
    <rejectfont>
      <pattern>
        <patelt name="scalable" >
          <bool>false</bool>
        </patelt>
      </pattern>
    </rejectfont>
  </selectfont>
  <match target="font" >
    <edit mode="assign" name="rgba" >
      <const>rgb</const>
    </edit>
  </match>
  <match target="font" >
    <edit mode="assign" name="hinting" >
      <bool>true</bool>
```

```
</edit>
</match>
<match target="font" >
  <edit mode="assign" name="hintstyle" >
    <const>hintmedium</const>
  </edit>
</match>
<match target="font" >
  <edit mode="assign" name="antialias" >
    <bool>true</bool>
  </edit>
</match>
</fontconfig>
```


Руководство по видеокартам nVidia в Gentoo Linux

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/nvidia-guide.xml>

С версии: 1.0

1. Введение

Компания **nVidia** выпускает свои собственные драйверы для Linux, отличающиеся хорошей производительностью и поддержкой трехмерного ускорения. В дереве Portage присутствуют два драйвера — **nvidia-drivers** для новых видеокарт nVidia и **nvidia-legacy-drivers** для старых.

Примечание: Раньше в Gentoo было два отдельных пакета — для модуля ядра nVidia(nvidia-kernel) и для библиотек GLX X11 (nvidia-glx). Эти пакеты были удалены из дерева Portage для замены на nvidia-drivers и nvidia-legacy-drivers. Если вы до сих пор используете nvidia-kernel и nvidia-glx, то вы должны перейти на использование новых пакетов.

2. Настройка карты

Конфигурирование ядра

Как уже было сказано, драйвер **nVidia** настраивается и запускается для текущего ядра. Он собирается в качестве модуля, поэтому ядро должно поддерживать загрузку модулей. Если вы использовали **genkernel** для конфигурации ядра, тогда все уже сделано. Если же нет, тогда дважды проверьте конфигурацию вашего ядра на наличие поддержки загружаемых модулей:

Листинг 2.1: Включение загрузки модулей ядра

```
Loadable module support --->
[*] Enable loadable module support
```

Также вам потребуется включить в ядре **Memory Type Range Register**:

Листинг 2.2: Включение MTRR

```
Processor and Features --->
[*] MTRR (Memory Type Range Register) support
```

Также, если у вас AGP-видеокарта, вы можете включить поддержку **agpgart** в ядре или в виде модуля. Если вы не используете встроенный **agpgart**, то

драйверы будут использовать собственную реализацию **agpgart** под названием **NvAGP**. На одних системах эта реализация работает лучше, чем встроенная, на других — хуже. Вам понадобится проверить это на своей собственной системе, чтобы сделать выбор в пользу лучшей производительности. Если вы не хотите экспериментировать, просто используйте встроенный в ядро agpgart:

Листинг 2.3: Включение agpgart

```
Device Drivers --->
Character devices --->
<*> /dev/agpgart (AGP Support)
```

Замечания к архитектурам

Важно: Что касается процессоров **x86** и **AMD64**, встроенный в ядро драйвер конфликтует с двоичным драйвером, выпускаемым **nVidia**. Если вы собираете ядро для этих процессоров, следует полностью убрать поддержку встроенного драйвера, как показано ниже:

Листинг 2.4: Удаление встроенного драйвера

```
Device Drivers --->
Graphics Support --->
< > nVidia Framebuffer Support
< > nVidia Riva support
```

Хорошей альтернативой кадрового буфера будет VESA:

Листинг 2.5: Включение поддержки VESA

```
Device Drivers --->
Graphics Support --->
<*> VESA VGA graphics support
```

Затем в разделе «VESA driver type» выберите либо **vesafb**, либо **vesafb-tng**. Если у вас процессор AMD64, вы должны выбрать **vesafb**, а не **vesafb-tng**:

Листинг 2.6: Выбор типа кадрового буфера

```
(X) vesafb
( ) vesafb-tng
```

Если вы используете **vesafb**, то дополнительную информацию можно посмотреть в файле **/usr/src/linux/Documentation/fb/vesafb.txt**, или же найдите нужную документацию по кадровому буферу в каталоге **/usr/src/linux/Documentation/fb/**.

Последующая настройка ядра

The **nvidia-drivers** and **nvidia-legacy-drivers ebuilds** автоматически определяют версию ядра на основании символической ссылки **/usr/src/linux**. Пожалуйста проверьте правильность конфигурации ядра и что эта ссылка правильно

указывает на исходные коды. Обратитесь к разделу «Конфигурирование ядра» из Настольной книги за подробностями конфигурирования ядра.

Если вы используете gentoo-sources-2.6.11-r6, каталог /usr/src может выглядеть примерно так:

Листинг 2.7: Проверка символической ссылки /usr/src/linux

```
# cd /usr/src
```

```
# ls -l
```

(Проверьте, что ссылка linux указывает на нужный каталог)

```
lrwxrwxrwx 1 root root 22 Apr 23 18:33 linux -> linux-2.6.11-gentoo-r6
```

```
drwxr-xr-x 4 root root 120 Apr 8 18:56 linux-2.4.26-gentoo-r4
```

```
drwxr-xr-x 18 root root 664 Dec 31 16:09 linux-2.6.10
```

```
drwxr-xr-x 18 root root 632 Mar 3 12:27 linux-2.6.11
```

```
drwxr-xr-x 19 root root 4096 Mar 16 22:00 linux-2.6.11-gentoo-r6
```

В приведенном выводе можно увидеть, что символическая ссылка linux указывает на ядро linux-2.6.11-gentoo-r6.

Если ссылка не указывает на правильные исходники, вы должны обновить ее:

Листинг 2.8: Создание или обновление символической ссылки /usr/src/linux

```
# cd /usr/src
```

```
# ln -snf linux-2.6.11-gentoo-r6 linux
```

Необязательный шаг: уточнение поддержки устаревших видеокарт

Примечание: К несчастью, некоторые устаревшие видеокарты не поддерживаются более новыми версиями пакетов nvidia-drivers. У nVidia есть список поддерживаемых видеокарт. Посмотрите этот список, прежде чем будете устанавливать драйверы.

В следующем списке указаны неподдерживаемые ныне видеокарты:

Листинг 2.9: Неподдерживаемые видеокарты

TNT2

TNT2 Pro

TNT2 Ultra

TNT2 Model 64 (M64)

TNT2 Model 64 (M64) Pro

Vanta

Vanta LT

GeForce 256

GeForce DDR

GeForce2 GTS

GeForce2 Pro

GeForce2 Ti

GeForce2 Ultra

GeForce2 MX Integrated graphics

Quadro

Quadro2 Pro

Если ваша карта присутствует в списке устаревших карт, то для поддержки 3D вам потребуется установить **nvidia-legacy-drivers**.

Установка подходящих драйверов

Теперь пора установить драйверы.

Листинг 2.10: Установка драйверов nVidia
(если вашей карты нет в списке устаревших карт)
emerge nvidia-drivers
(если ваша карта присутствует в списке устаревших карт)
emerge nvidia-legacy-drivers

Важно: Каждый раз, когда вы собираете новое ядро или пересобираете текущее, вам понадобится запускать `emerge nvidia-drivers` или `emerge nvidia-legacy-drivers` для переустановки модулей ядра nVidia.

После завершения установки запустите `modprobe nvidia` для загрузки модуля в память.

Листинг 2.11: Загрузка модуля
lsmod | grep nvidia && rmmod nvidia
modprobe nvidia

Чтобы не приходилось вручную загружать модуль после каждой загрузки, вероятно, вам потребуется, чтобы это делалось автоматически при загрузке, поэтому откройте `/etc/modules.autoload.d/kernel-2.6` (или `kernel-2.4` в зависимости от используемой версии ядра) и добавьте строку `nvidia` в файл. После этого не забудьте запустить **modules-update**.

Важно: Если вы собрали `agpgart` в качестве модуля, то вам понадобится указать его в `/etc/modules.autoload.d/kernel-2.6` (или `kernel-2.4`, в зависимости от версии ядра).

Листинг 2.12: Запуск `modules-update`
modules-update

Настройка X-сервера

Как только необходимые драйверы установлены, вам потребуется настроить X-сервер, чтобы задействовать драйвер `nvidia` вместо используемого по умолчанию `nV`.

Откройте файл `/etc/X11/xorg.conf` своим любимым текстовым редактором (как `nano` или `vim`) и найдите раздел `Device`. В этом разделе измените строку `Driver`:

Листинг 2.13: Замена `nv` на `nvidia` в файле настроек X-сервера

```
Section "Device"
    Identifier "nVidia Inc. GeForce2"
    Driver     "nvidia"
    VideoRam   65536
EndSection
```

Теперь перейдите в раздел `Module` и проверьте, что будет загружаться модуль `glx`, а `dri` — не будет:

Листинг 2.14: Изменение раздела `Module`

```
Section "Module"
    (...)
    # Load "dri"
    Load "glx"
    (...)
EndSection
```

Далее, в разделе `Screen` проверьте, что каждая из инструкций `DefaultDepth` установлена в 16 или 24, или что у вас есть только подразделы `Display` с установленной `Depth` в 16 или 24. Без этого расширение `nVidia GLX` не будет запускаться.

Листинг 2.15: Обновление раздела `Screen`

```
Section "Screen"
    (...)
    DefaultDepth 16
    Subsection "Display"
    (...)
EndSection
```

Запустите **`eselect`**, чтобы X-сервер использовал GLX-библиотеки `nVidia`:

Листинг 2.16: Запуск `eselect`

```
# eselect opengl set nvidia
```

Добавление пользователей в группу `video`

Добавьте ваших пользователей в группу `video`, чтобы у них был доступ к файлам устройств `nvidia`:

Листинг 2.17: Добавление пользователя в группу `video`

```
# gpasswd -a youruser video
```

Если вы не используете файловую систему **udev**, то, в целом, нет необходимости делать шаги, описываемые в следующем разделе, но они ничего не испортят и сделают систему стойкой к переменам.

Тестирование карты

Для тестирования карты nVidia запустите X и наберите `glxinfo | grep direct`. Вы должны увидеть, что прямая прорисовка (direct rendering) включена:

Листинг 2.18: Проверка статуса прямой прорисовки

```
$ glxinfo | grep direct
direct rendering: Yes
```

Чтобы узнать скорость прорисовки FPS (кадров в секунду), используйте программу **glxgears**.

Включение поддержки nvidia

Некоторые пакеты, например, `mplayer` и `xine-lib`, используют локальный USE-флаг «`nvidia`», который разрешает поддержку аппаратного ускорения потока Mpeg (XvMCNVIDIA), полезного при просмотре широкоформатных фильмов. Добавьте «`nvidia`» к переменной USE в файле `/etc/make.conf` или добавьте его в качестве USE-флага для `media-video/mplayer` и/или `media-libs/xine-lib` в `/etc/portage/package.use`.

Затем запустите `emerge -uD --newuse world`, чтобы пересобрать приложения, которых коснулось изменение USE-флагов.

Использование инструмента nVidia Settings

Начиная с версии 1.0.6106, компания nVidia предоставляет программу настройки. Эта программа может изменять настройки графики без перезапуска X-сервера и находится в дереве Portage под названием `media-video/nvidia-settings`.

3. Проблемы

Работа 2D на компьютерах с ОЗУ 4 ГБ и более

Если вы сталкиваетесь с проблемами при 2D-ускорении nVidia, скорее всего, вы не можете установить диапазон совмещения записи (`write-combining range`) для MTRR. Чтобы проверить это, просмотрите содержимое `/proc/mtrr`:

Листинг 3.1: Включена ли поддержка `write-combining`

```
# cat /proc/mtrr
```

Каждая строка должна содержать «write-back» или «write-combining». Если вы видите строку, содержащую «uncachable», то вам понадобится изменить настройки BIOS, чтобы исправить это.

Перезагрузитесь и войдите в BIOS, затем найдите настройки MTRR (возможно в «CPU Settings»). Измените установки с «continuous» на «discrete» и загрузитесь назад в Linux. Теперь вы не нигде не должны увидеть значения «uncachable», и 2D-ускорение должно работать без какого-либо замедления.

Пытаясь загрузить модуль ядра, я в ответ получаю «no such device»

Обычно это происходит, когда у вас нет подходящей видеокарты. Убедитесь, что ваша видеокарта основана на чипе **nVidia** (это можно проверить командой `lspci`).

Если вы уверены, что у вас карта **nVidia**, проверьте BIOS и проверьте, включен ли параметр Assign IRQ to VGA.

4. Настройка для опытных

Документация

Пакет драйверов nVidia поставляется с исчерпывающей документацией. Она устанавливается в `/usr/share/doc`, ее можно посмотреть следующей командой:

```
Листинг 4.1: Просмотр документации NVIDIA
(для nvidia-drivers)
$ less /usr/share/doc/nvidia-drivers-*/README.gz
(для nvidia-legacy-drivers)
$ less /usr/share/doc/nvidia-legacy-drivers-*/README.gz
```

Параметры модуля ядра

Модуль ядра `nvidia` способен принимать некоторые параметры, с помощью которых вы можете изменять поведение драйвера. Большинство из них описано в документации. Чтобы добавить или изменить эти параметры, отредактируйте файл `/etc/modules.d/nvidia`. Не забудьте запустить `modules-update` после изменения этого файла и держите в голове, что вам понадобится перезагрузить модуль `nvidia`, чтобы новые настройки подействовали.

```
Листинг 4.2: Изменение параметров nvidia
(Отредактируйте /etc/modules.d/nvidia вашим любимым редактором)
# nano -w /etc/modules.d/nvidia
(Обновление информации о модулях)
```

```
# modules-update
(Выгрузка модуля nvidia...)
# modprobe -r nvidia
(...и его загрузка снова)
# modprobe nvidia
```

Расширенная настройка X-сервера

GLX-интерфейс также имеет множество настраиваемых параметров. Они управляют настройками TV-вывода, двойных мониторов, определением частоты монитора и т.д. И снова, все доступные параметры подробно описаны в документации.

Если вы хотите задействовать любой из этих параметров, то вам нужно перечислить их в соответствующем разделе Device конфигурационного файла X-сервера (обычно /etc/X11/xorg.conf). Например, я хочу отключить всплывающий логотип nVidia при загрузке X-сервера:

Листинг 4.3: Расширенная конфигурация nvidia в настройках X-сервера

```
Section "Device"
    Identifier "nVidia Inc. GeForce2"
    Driver     "nvidia"
    Option     "NoLogo" "true"
    VideoRam   65536
EndSection
```


Распространенные вопросы об ATI в Gentoo Linux

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/ati-faq.xml>

С версии: 1.0

1. Поддержка устройств

Поддерживается ли моя видеоплата ATI?

В xorg-x11 поддерживаются многие (но не все) видеоплаты ATI, по крайней мере, поддерживаются их функции ускорения двумерной графики. Поддержка трехмерного ускорения обеспечивается либо проектом DRI, входящим в состав xorg-x11, либо драйверами с закрытым исходным кодом, поставляемыми фирмой ATI.

Графический процессор	Распространенное название	Поддержка
Rage128	Rage128	xorg DRI
R100 Radeon 7xxx,	Radeon 64	xorg DRI
R200, R250, R280	Radeon 8500, Radeon 9000, Radeon 9200	xorg DRI, ATI DRI
R300, R400, R500	Radeon 9500—x800	xorg 2D, ATI DRI

У меня есть видеоплата All-In-Wonder/Vivo. Поддерживаются ли ее мультимедийные возможности?

Мультимедийные возможности этой платы поддерживаются проектом GATOS. Эти драйверы будут в скором времени включены в дерево разработки xorg.

Я использую не архитектуру x86. Какие у меня варианты?

Поддержка X11 на платформах PPC и Alpha сходна с поддержкой X11 на x86. Однако, драйверы производства ATI с закрытым исходным кодом не поддерживаются на PPC или Alpha, поэтому вы не сможете использовать трехмерные возможности графического процессора (ГП) R300. Если у вас именно такая видеоплата, и вы хотите, чтобы она поддерживалась в X11, свяжитесь с ATI и потребуйте, чтобы они выпустили описание вашего ГП. Для AMD64 уже выпущены закрытые драйверы, поэтому теперь пользователи AMD64 могут наслаждаться теми же возможностями видеоплат, что и пользователи x86.

Важно: Для включения поддержки `agpgart` на некоторых материнских платах AMD64 вам потребуется отключить поддержку K8 IOMMU.

У меня есть ноутбук. Поддерживается ли моя модель ATI Mobility?

Должна, но у вас может возникнуть проблема с настройкой из-за идентификатора OEM PCI ID, присвоенного изготовителем. В большинстве случаев, вам может потребоваться самостоятельно составить файл конфигурации или использовать

утилиту `xorgconfig`.

2. Установка

Пакеты

В пакете **xorg-x11** представлена реализация X11

В ядрах 2.6.x модули DRI могут собираться вместе с ядром или устанавливаться пакетом **x11-drm**

В ядрах 2.4.x необходимо использовать пакет `x11-drm`

В пакете **ati-drivers** представлены закрытые драйверы ATI для X и модули ядра для ядер как серии 2.4, так и 2.6

Если вы хотите использовать внутреннюю поддержку `agpgart` производства ATI вместо предоставляемой в ядре Linux, то драйвер `agpgart` и драйвер материнской платы (в конфигурации вашего ядра) должны быть собраны как модули или вообще не собираться.

Листинг 2.1: Установка драйверов

(если нужно установить с X11 только драйверы и модули Rade128)

```
# VIDEO_CARDS="r128" emerge x11-drm
```

(чтобы получить только поддержку Radeon)

(R100, R200, R250, R280, пока без R300)

```
# VIDEO_CARDS="radeon" emerge x11-drm
```

(чтобы установить драйверы фирмы ATI с закрытым исходным кодом)

(R200, R250, R280 и R300)

```
# emerge ati-drivers
```

(чтобы установить X11 без каких-либо модулей ядра)

```
# emerge xorg-x11
```

Настройка

Для создания конфигурационного файла `xorg.conf` рекомендуется использовать `xorgcfg` или `xorgconfig`. В качестве альтернативы вы можете использовать параметр автоконфигурации Xorg:

Листинг 2.2: Автоматическая настройка X

```
# X -configure
```

За дополнительной информацией о том, как создать базовый файл конфигурации `xorg.conf`, пожалуйста обратитесь к описанию настройки X-сервера в Gentoo.

Примечание: Если вы установили пакет `ati-drivers`, вы также сможете использовать **`fglrxconfig`**.

Важно: У пользователей PPC есть возможность использовать отдельный конфигуратор **`Xorgautoconfig`**, установив пакет `Xorgautoconfig`, но это

необязательно.

Переключение на OpenGL

Как только X установлен, настроен и заработал, он способен использовать библиотеки ATI OpenGL:

Листинг 2.3: Запуск `eselect`
`# eselect opengl set ati`

3. Дополнительные источники

Прочтите [Hardware Acceleration Guide](#) для получения дополнительных сведений по настройке своей графической карты ATI.

Дополнительные сведения о Gentoo Linux и двоичных драйверах ATI можно прочитать в [Wedge Unofficial Gentoo ATI Radeon FAQ](#).

HOWTO fbsplash

Ссылка на оригинал: http://ru.gentoo-wiki.com/HOWTO_fbsplash

С версии: 1.2

Введение

Во первых я хочу выразить благодарность gentoo-разработчику [Spock](#) за разработку gensplash (fbsplash and splashutils) и vesafb-tng. Как альтернативу этому how-to, вы можете попробовать Spock's [Gensplash-in-5-easy-steps How-to](#).

Что такое gensplash?

Gensplash - это проект основанный [Spock`om](#), цель которого — создание набора инструментов, способных "обогатить" процесс загрузки Gentoo (отображение сообщений, анимации, полос состояния, и т. д.).

Что такое fbsplash (framebuffer splash)?

Fbsplash - это часть gensplash проекта. Согласно документации по fbsplash, "framebuffer splash - это особенность ядра которая разрешает отображать фоновые изображения на выбранных консолях и переключать первую консоль в так называемый безмолвный режим (silent mode), в то время когда система загружается/перезагружается/выключается. Доступ до устройства fbsplash осуществляется путем обращения к `/dev/fbsplash`. Для работы с fbsplash, требуется устройство поддерживающее framebuffer.

Что такое устройство frame buffer?

Устройство frame buffer обеспечивает уровень абстракции устройствам вывода графики, что обеспечивает буфер кадров некоторых видеоустройств и интерфейс доступа к ним для программного обеспечения, т. е. программе не обязательно знать о существовании низкоуровневых системных вызовов, регистров и т. д. Доступ к устройству осуществляется через определенные файлы, обычно расположенные в ветке `/dev` дерева каталогов, в частности `/dev/fb*`.

Существует множество различных frame buffer устройств.

1. Некоторые из них, стандартные поддерживают большинство современных видеокарт:
 - vesafb (vesa frame buffer);
 - vesafb-tng (vesa frame buffer нового поколения).
2. Другие специфичные, будут работать только с определенными моделями:
 - rivaafb (может работать с nvidia видеокартами);
 - radeonfb (может работать с ati видеокартами).

Предполагается, что Fbsplash будет работать на любом из frame buffer устройств (и вообще на любой видеокарте способной на 8/15/16/24/32 битную цветовую гамму и directcolor/pseudocolor режимы).

Установка fb splash и splashutils

В дальнейшем как пример используется vesafb-tng frame buffer устройство. Также подразумевается что вы используете Gentoo дистрибутив.

Code: **Emerging splashutils & themes**

```
# emerge -va splashutils splash-themes-gentoo splash-themes-livedcd
```

Установка заплат ядра

Во первых убедитесь что fb splash и vesafb-tng заплаты установлены в вашем ядре. Начиная с версии 2.6.10 (а может и раньше) всё это входит в gentoo-sources. Если у вас нет этих заплат вы можете взять их с [Spock's homepage](#)

Конфигурация ядра

Во первых нужно включить поддержку vesafb-tng frame buffer в вашем ядре (вы можете пропустить этот шаг если у вас уже настроен framebuffer драйвер отличный от vesa-tng, например i810fb):

```
Device Drivers ---> Graphics support--->:
<*> Support for frame buffer devices
<*>   VESA VGA graphics support
      VESA driver type (vesafb-tng)
(1280x1024@60) VESA default mode
```

Еще вам нужна поддержка fb splash в вашем ядре.

```
Device Drivers > Graphics support > Console display driver support:
<*> Framebuffer Console support
<*> Support for the framebuffer splash
```

Если вы хотите загружать initramfs изображение динамически во время загрузки (смотрите "Загрузка initramfs изображений динамически во время загрузки" раздел этого howto), вы можете включить еще одну специальную опцию. Если вы ещё не знаете что хотите делать, включите её все равно (это не повредит :)

```
Device Drivers > Block devices:
<*> RAM disk support
(4096) Default RAM disk size (kbytes) (NEW)
[*]   Initial RAM disk (initrd) support
```

Initial RAM disk (initrd) support теперь (в ядре 2.6.22) находится в General setup.

Компиляция ядра (Вручную)

```
# cd /usr/src/linux
```

Вы должны убедиться что не осталось файлов после предыдущей компиляции в каталоге вашего ядра:

```
# make mrproper
```

Теперь настраиваем ядро.

```
# make menuconfig
```

После того как вы настроили ваше ядро, откомпилируйте (Важно: этот шаг нужно сделать до установки splashutils). "make" скомпилирует одновременно и образ ядра('make bzImage') и модули ('make modules').

```
# make
```

Конфигурация / компиляция ядра (Для пользователей genkernel)

Необходимо запустить genkernel со следующими ключами:

```
# genkernel --gensplash=livedcd-2006.0 --gensplash-res=1280x1024 --menuconfig all
```

где ключи означают следующее:

--gensplash=livedcd-2006.0 - использовать тему livedcd-2006.0 (из пакета splash-themes-livedcd),

--gensplash-res=1280x1024 - при загрузке необходимо использовать разрешение 1280x1024,

--menuconfig - необходимо запустить ncurses конфигуратор ядра,

all - компилировать ядро, модули и initrd образ.

Установка splashutils

Теперь настало время для установки splashutils. Splashutils это ПО используемое для загрузки изображений (а также перекодировки jpg, png).

```
# emerge splashutils
```

Конечно, /usr/src/linux должна ссылаться на исходники с установленными заплатками..

Конфигурация / компиляция ядра (ручная работа)

Теперь ваше ядро и система подготовлены для того чтобы показать splash screen. Для завершения компиляции и установки вашего нового ядра осталось сделать следующее.

Если /boot расположен на отдельном разделе подмонтируйте его.

1. mount /boot

Затем инсталлируйте модули и образ ядра.

1. make modules_install
2. make install

Создание образа initramfs

Приведенных выше инструкций (а также второй части следующего howto, [Configuring your bootloader](#)) достаточно, чтобы получить splash screen после того как ядро будет загружено. Однако если вы хотите чтобы сексапильная девка с пингвином на маечке появлялась раньше (сразу после инициализации - vesafb-tng), вам понадобится **initramfs**.

Что такое initramfs?

(Цитата [Spock'a](#)): *"Initramfs (initial ramfs) это кусок кода который распаковывает сжатый cpio образ во время загрузки ядра.*

Чтобы использовать initramfs его надо сначала создать. Образ initramfs должен содержать:

1. картинки, которые будут использоваться как фон;
2. необходимы файлы конфигурации;
3. утилиту помощника (splash_helper).

Изображения и конфигурационные файлы вместе образуют **тему gensplash**.

Существует **два** пути для использования initramfs image:

1. компиляция initramfs image непосредственно в ядро;
2. загрузка initramfs image динамически во время загрузки системы (с использованием 'initial ram disk' - initrd).

Вы можете сами выбрать вариант, который вам по душе. .

Компиляция образа initramfs непосредственно в ядро

Сначала удалите старый образ initramfs:

```
# rm /usr/src/linux/usr/initramfs_data.cpio.gz
```

Теперь создайте новый образ initramfs, содержащий изображения, конфигурационные файлы и userspace helper. Проще всего будет сделать это при помощи **splash_geninitramfs** (идущим в составе splashutils). Его использование похоже на утилиту 'splash' из пакета bootsplash.

```
# splash_geninitramfs -v -g /usr/src/linux/usr/initramfs_data.cpio.gz -r
1024x768 emergence
```

При компиляции образа initramfs непосредственно в ядро, вам придется пересобрать ядро (этого недостатка лишен способ с использованием initrd):

```
# cd /usr/src/linux
# touch usr/initramfs_data.cpio.gz
# make bzImage
# mount /boot
# make install
```

Загрузка образа initramfs динамически

Этот вариант практически идентичен сборке образа initramfs вместе с ядром. Различие заключается в том, что вместо создания образа **внутри** ядра (и последующей сборке его командой 'make' и копированием в точку монтирования /boot при помощи 'make install'), вы создадите образ initramfs как **отдельный (initrd) файл** и вам не придется пересобирать ядро. Однако, придется добавить строчку в конфигурационный файл загрузчика (по которой он найдет initrd файл). Этим мы и займемся в этой части howto

Сначала при помощи splash_geninitramfs создадим initrd файл (который содержит изображения, конфиги и userspace helper):

```
# mount /boot
# cd /etc/splash
```

```
# splash_geninitramfs -v -g /boot/fbsplash-emergence-1024x768 -r 1024x768
emergence
```

Как видно файл будет создан в точке монтирования(или каталоге) /boot.

К сожалению, по [словам Spock'a](#), *"вы не сможете дописать новую initramfs информацию к initrd образу"*. Это означает, что данной возможностью не смогут воспользоваться пользователи genkernel ядер, т. к. genkernel создает собственный образ initrd для процесса загрузки.

solshark [solshark_at_rambler_dot_ru]: genkernel 3.3.6 позволяет сделать так:

1. genkernel --gensplash=DangerGirl initrd

Так что теперь и для пользователей genkernel есть возможность воспользоваться данной фишкой.

Если вы не меняете ядро, то просто поменяйте строку описания initrd в [grub.conf](#) с:

```
root=/dev/ram0 real_root=/dev/$ROOT
initrd (hd0,0)/boot/initrd
```

на:

```
root=/dev/$ROOT
initrd (hd0,0)/boot/fbsplash-emergence-1024x768
```

Создание собственных тем для gensplash

Стандартная тема 'emergence' и её конфигурационные файлы находятся в [/etc/splash/emergence](#).

Если хотите создать новую (например с названием 'newtheme'), вам следует:

1. создать для неё каталог в [/etc/splash](#):

```
# mkdir /etc/splash/newtheme
```

2. создать каталог 'images' в [/etc/splash/newtheme](#):

```
# mkdir /etc/splash/newtheme/images
```

и скопировать туда картинки (в форматах png/jpg) которые будут использованы в теме.

3. создать необходимые конфигурационные файлы. Они должны иметь расширение *.cfg и находиться в каталоге темы ([/etc/splash/newtheme](#)). Образцы конфигурационных файлов можно посмотреть в теме ('emergence'):

```
# ls -l /etc/splash/emergence/
images/
1024x768.cfg
1280x1024.cfg
800x600.cfg
```

Очень важно убедиться в том в опциях 'jpeg=' и 'silentjpeg=' (в *.cfg файлах) прописаны корректные ссылки на файлы с изображениями (в каталоге [/etc/splash/newtheme/images](#)). Если вы не используете jpeg для создания темы, замените опции на 'pic=' и 'silentpic='.

Для более близкого знакомства с форматом тем fbsplash, изучите документацию

пакета splashutils:

```
# less /usr/doc/splashutils-version/theme_format.gz
```

Перевод тем bootsplash в темы gensplash

Так же Spock сделал удобную утилиту **bootsplash2fbsplash** (этот скрипт идет в комплекте со splashutils) ,который сделает легким преобразование тем bootsplash формат gensplash.

Например если у вас есть тема 'gentoo-highquality' ,расположенная в [/etc/bootsplash](#) ,вам нужно только скопировать(от рута):

```
# bootsplash2fbsplash gentoo-highquality
```

и утилита сделает gensplash-совместимую тему в каталоге [/etc/splash](#) ([/etc/splash/gentoo-highquality](#)) со всеми необходимыми файлами.

Использование собственных тем gensplash

Для создания нового образа из темы которую вы только что сделали(пусть ее имя 'newtheme',а разрешение 1280x1024), вам необходимо использовать splash_geninitramfs.

Если вы решили вкомпилировать образ в ядро(первый вариант):

```
# splash_geninitramfs -v -g /usr/src/linux/usr/initramfs_data.cpio.gz -r 1280x1024 newtheme
# cd /usr/src/linux
# touch usr/initramfs_data.cpio.gz
# make bzImage
# mount /boot
# make install
```

Если вы используете initrd (второй вариант):

```
# mount /boot
# splash_geninitramfs -v -g /boot/fbsplash-newtheme-1280x1024 -r 1280x1024 newtheme
```

Вот что означают опции splash_geninitramfs:

```
-v splash_geninitramfs выводит дополнительную информацию
-g указывает splash_geninitramfs создать образ initramfs
  (/usr/src/linux/usr/initramfs_data.cpio.gz
  или /boot/fbsplash-newtheme-1280x1024)
-r говорит splash_geninitramfs использовать разрешение 1280x1024
-'newtheme'указывает splash_geninitramfs искать .cfg файлы
  в каталоге /etc/splash/newtheme
```

Для просмотра описания всех опций splash_geninitramfs' введите:

```
# splash_geninitramfs --help
```

Последний пункт в этой части how-to,полезный совет из документации splashutils':

Если вы хотите сохранить немного памяти и не планируете использовать все доступные разрешения вы можете скопировать .cfg файлы и фоновые картинки только для того разрешения,которое вы используете.

А теперь пришло время настроить ваш загрузчик.

Настройка загрузчика

Параметры ядра

Вот опции, которые необходимо добавить в конфигурационные файлы grub(lilo):

1. Необходимо указание устройства frame buffer. Вот как это может выглядеть например для драйвера vesafb-tng:

```
video=vesafb:ywrap,mtrr,1024x768-16@85
```

Вышеприведенный пример указывает на то, что мы хотим использовать разрешение 1024x768, 16 бит цвет и частоту обновления 85Hz. Если вы используете vesafb-tng, вам также **необходимо** удалить опцию 'vga=' из файла конфигурации вашего загрузчика (эта опция используется только 'обычными' vesafb frame buffer устройствами):

```
# НЕ СТАВЬТЕ опцию 'vga=' для vesafb-tng (если она стоит, то удалите ее):
vga=791
vga=0x517
```

2. fbsplash'y необходимы свои параметры:

```
splash=verbose,theme:emergence
```

Сначала fbsplash выбирает **режим** (в примере используется режим 'verbose'). Вот цитата из документации по fbsplash *"fbsplash может работать в двух режимах: **verbose** и **silent**. Первый режим подразумевает простую консоль с фоновым рисунком. Режим **silent** был впервые реализован в bootsplash. Когда этот режим включен, консоль переключается в графический режим. При этом, текст не печатается. Все что будет отображаться в этом режиме зависит от программ, запущенных в пользовательском пространстве. Таким образом, может быть реализован, например, индикатор прогресса загрузки.*

Вторая часть параметра fbsplash позволяет указать какую **тему** нужно использовать (в примере используется стандартная тема 'emergence').

Для работы fbsplash необходимо указать оба параметра: режим и тему.

3. Если вы решили использовать **initrd** (посмотрите предыдущую главу - [Загрузка образа initramfs](#)), вам необходимы дополнительные опции в конфигурации загрузчика:

```
initrd /boot/fbsplash-emergence-1024x768      # Grub
initrd = /boot/fbsplash-emergence-1024x768    # Lilo
```

В приведенном примере подразумевается, что имя рама диска fbsplash-emergence-1024x768 (и он размещен в каталоге или разделе /boot).

Следующие примеры для обоих загрузчиков grub и lilo. Используется тема 'newtheme', разрешение 1280x1024, 32 бит цвета частота обновления 70Hz. Указан режим fbsplash 'silent' и используется initrd файл.

Пример конфигурации Grub

```
title Gentoo
```

```

root (hd0,0)
kernel (hd0,0)/kernel-2.6.8-gentoo-r1 root=/dev/hda3
video=vesafb:ywrap,mtrr,1280x1024-32@70 splash=silent,theme:newtheme
initrd (hd0,0)/fb splash-newtheme-1280x1024

```

Замените (hd0,0) своим загрузочным разделом (или,если /boot не на отдельном разделе,используйте корневой раздел добавив '/boot' к путям образа ядра и initrd) Пользователи genkernel должны использовать что-то типа этого(все в одной строке):

```

kernel (hd0,0)/kernel-2.6.8-gentoo-r1 init=/linuxrc root=/dev/hda3
video=vesafb:ywrap,mtrr,1280x1024-32@70 splash=silent,theme:newtheme

```

Пример конфигурации Lilo

```

image=/boot/kernel-2.6.8-gentoo-r1
label = "Gentoo"
root = /dev/hda3
append="video=vesa:ywrap,mtrr,1280x1024-32@70 splash=silent,theme:newtheme"
read-only
initrd = /boot/fb splash-newtheme-1280x1024

```

Не забудьте запустить lilo после изменения lilo.conf:

```

# mount /boot
# lilo -v

```

После ребута и вы получите графическую загрузку.

Добавление фона на все консоли

Если вы хотите чтобы на всех консолях был сплеш (картинка из режима 'verbose'),а не только на первой,то отредактируйте [/etc/conf.d/splash](#) установив нужные опции и и добавьте splash в ваш стандартный уровень запуска:

```
# rc-update add splash default
```

Если вы используете bootsplash, удалите его из уровня запуска:

```
# rc-update del bootsplash
```

Gensplash для других дистрибутивов

(Jakub) Я думаю этот раздел должен состоять из трех частей:

Вступление: Gensplash работает на других дистрибутивах.

1. Что нужно сделать по-другому (в сравнении с этим руководством) - к примеру из-за отсутствия emerge. Это касается всех дистрибутивов не подобных Gentoo.
2. Какие дополнительные действия нужно предпринять - например модификация загрузочных скриптов. Тоже для не Gentoo дистрибутивов.
3. Другая специфика не охваченная предыдущими двумя разделами.

Различия

1. Наложение патчей на ядро - необходимо. Мы не используем gentoo-dev-

- sources (хотя все возможно)
2. Конфигурация/сборка ядра - из-за невозможности использования genkernel.
 3. Установка splashutils - нет package.keywords, нет emerge. Ручками:). Добавить ссылки. Предложение перейти на Gentoo было бы своевременно.
 4. Добавление splash screen на все терминалы(и удаление bootpslash) - скорее всего зависит от дистрибутива. Способ для Gentoo скорее всего работать не будет (очевидно).

Дополнительные шаги

Модификация загрузочных скриптов потребуется для работы индикатора прогресса. Два варианта:

- с использованием bootsplash (простой)
- без использования bootsplash (более сложный)

Тут должно быть название пункта 3

Вариант для Debian упоминался Spock'ом на forums.gentoo.org.

Решение проблем

Если /usr на отдельном разделе

Если у вас splashutils версии 0.9-pre08 и новее и каталог /usr монтируется с другого раздела, gensplash должен работать "из коробки" (без напильника). Однако, если у вас более древняя версия splashutils и, по каким-либо причинам, нет возможности ее обновить следуйте нижеописанным инструкциям.

Программы и библиотеки, которые нужны splash_util'y чтобы запустить fbset и отобразить вашу сплешку расположены в /usr/bin и /usr/lib. И если эти каталоги находятся на других разделах (например при использовании RAID или LVM2 архитектуры), они могут быть недоступны (не примонтированы) в момент, когда они нужны splash_utils и вы не сможете увидеть свою замечательную сплешку.

Как решить? Скопируйте их в место где они **доступны** как описано в (спасибо **aslocum** и форумам Gentoo) [Gensplash-How-to with progressbar!!!!](#) треде:

```
# cp /usr/bin/fbset /sbin
# cp /usr/lib/libjpeg.so.62 /lib
# cp /usr/lib/libpng.so.3 /lib
```

поправьте /sbin/splash (например при помощи nano, набрав `nano -w /sbin/splash`) и приведите строку 31 к виду:

```
res=`/sbin/fbset | grep "^mode" | sed -r 's/mode "([0-9]+x[0-9]+).*/\1/'`
```

Где надо только прописать /sbin/ перед fbset. Сохраните результат и выйдите из редактора, и ваша сплешка должна появиться.

Установка фона во время загрузки

Вы наверное заметили, что фоновое изображение загружается с небольшой задержкой, даже если вы используете образ initramfs. До этого момента используется разрешение 640x480 и черно-белая цветовая гамма.

Такое положение дел нормально. Единственным выходом (чтобы получил фон как можно раньше) будет использование драйвера vesafb вместо vesafb-tng. Вот [ответ Spock'a](#) по этому поводу: "Образ *initramfs* используется, чтобы подключить корневую ФС, и до того как инициализируется кадровый буфер, так что не волнуйтесь по этому поводу. Как вы заметили, компьютер загружается в текстовом режиме, а не в графическом. Графический режим устанавливается после инициализации подсистемы кадрового буфера. Но до этого момента должна произойти инициализация других вещей, а это требует времени. Вот почему в течение секунды-двух вы видите сообщения ядра. Вы наверно скажете, что с vesafb у вас такого не было. И будете правы. Но vesafb исключение, а не правило (его инициализация происходит во время загрузки ядра). Все остальные кадровые драйверы (vesafb-tng, rivafb, radeonfb) инициализируются с подсистемой fbdev".

Ссылки

- [Домашняя страница Spock'a](#)
- [Spock's Gensplash-in-5-easy-steps](#)

Из форумов Gentoo:

- [Gensplash-How-to с прогрессбаром!!!!](#)

Оффлайновая документация:

- Документация по Fbsplash (/usr/src/linux/Documentation/fb/splash.txt).
- Документация по кадровым буферам (/usr/src/linux/Documentation/fb/framebuffer.txt).
- Документация по splashutils (/usr/share/doc/splashutils-0.9_pre07/).

Hardware 3D Acceleration Guide (Англ.)

Ссылка на оригинал: <http://www.gentoo.org/doc/en/dri-howto.xml>

С версии: 1.0

1. Introduction

What is hardware 3D acceleration and why do I want it?

With hardware 3D acceleration, three-dimensional rendering uses the graphics processor on your video card instead of taking up valuable CPU resources drawing 3D images. It's also referred to as "hardware acceleration" instead of "software acceleration" because without this 3D acceleration your CPU is forced to draw everything itself using the Mesa software rendering libraries, which takes up quite a bit of processing power. While Xorg typically supports 2D hardware acceleration, it often lacks hardware 3D acceleration. Three-dimensional hardware acceleration is valuable in situations requiring rendering of 3D objects such as games, 3D CAD and modeling.

How do I get hardware 3D acceleration?

In many cases, both binary and open-source drivers exist. Open-source drivers are preferable since we're using Linux and open source is one of its underlying principles. Sometimes, binary drivers are the only option, like with nVidia's cards. Binary drivers include x11-drivers/nvidia-drivers and x11-drivers/nvidia-legacy-drivers for nVidia cards and media-video/ati-drivers for ATI cards. Other open-source drivers include media-video/kyro-kernel for KyroII cards and media-video/ati-gatos for ATI cards, which aim to support ATI's video capabilities more fully.

What is DRI?

The Direct Rendering Infrastructure, also known as the DRI, is a framework for allowing direct access to graphics hardware in a safe and efficient manner. It includes changes to the X server, to several client libraries and to the kernel. The first major use for the DRI is to create fast OpenGL implementations.

What is X11-DRM and how does it relate to regular Xorg?

X11-DRM is an enhancement to Xorg that adds 3D acceleration for cards by adding the kernel module necessary for direct rendering.

Purpose

This guide is for people who can't get direct rendering working with just Xorg. X11-DRM works for 3dfx, gamma, i8x0, matrox, rage128, radeon, mach64 and sis300 series drivers. VIA card owners should not use x11-drm, but should instead use the DRI provided in recent kernels (>2.6.13). See the DRI homepage for more info and documentation.

Feedback

With suggestions, questions, etc., e-mail Donnie Berkholz.

2. Install Xorg and configure your kernel

Install Xorg

Please read our Xorg Configuration Guide to get Xorg up and running.

Configure your kernel

Probe for your chipset and enable just that one.

Code Listing 2.1: Checking your AGP chipset

```
# emerge pciutils; lspci | grep AGP
```

```
# 00:01.0 PCI bridge: Intel Corp. 440BX/ZX/DX - 82443BX/ZX/DX AGP bridge (rev 03)
```

(Your output may not match the above due to different hardware.)

If your chipset is not supported by the kernel you might have some success by passing `agp=try_unsupported` as a kernel parameter. This will use Intel's generic routines for AGP support. To add this parameter, edit your bootloader configuration file!

Most, if not all, kernels should have these options. This was configured using gentoo-sources-2.4.20-r5.

Code Listing 2.2: Configuring the kernel

```
# ls -l /usr/src/linux
```

```
lrwxrwxrwx 1 root root 22 May 29 18:20 /usr/src/linux -> linux-2.4.20-gentoo-r5
```

(Make sure /usr/src/linux links to your current kernel.)

```
# cd /usr/src/linux
```

```
# make menuconfig
```

Code Listing 2.3: make menuconfig options

Processor type and features --->

<*> MTRR (Memory Type Range Register) support

Character devices --->

<M> /dev/agpgart (AGP Support)

[M] Intel 440LX/BX/GX and I815/I820/I830M/I830MP/I840/I845/I850/I860 support

(Enable your chipset instead of the above.)

< > Direct Rendering Manager (XFree86 4.1.0 and higher DRI support)

Make sure the Direct Rendering Manager (DRM) is off. The X11-DRM package will provide its own.

Compile and install your kernel

Code Listing 2.4: Compiling and installing kernel

(This example is for a 2.4 kernel)

```
# make dep && make clean bzImage modules modules_install
# mount /boot
# cp arch/i386/boot/bzImage /boot
```

If you want your kernel to be named something other than bzImage, be sure to copy to /boot/yourname instead. Don't forget to set up grub.conf or lilo.conf and run /sbin/lilo if you use LILO.

3. Install X11-DRM and configure direct rendering

Install X11-DRM

Code Listing 3.1: Installing X11-DRM

```
# emerge x11-drm
```

Configure Xorg.conf

Some chipsets require you to rebuild xorg-x11 with USE="insecure-drivers". This applies to mach64, unichrome and savage chipsets on xorg-x11-6.8.2, and to mach64 and unichrome on xorg-x11-6.8.99.x. Savage users should not try xorg-x11-6.8.99.x as support for the savage there is broken.

Code Listing 3.2: Rebuilding xorg-x11

(Add the insecure-drivers USE flag if you use one of the aforementioned chipsets)

```
# echo "x11-base/xorg-x11 insecure-drivers" >> /etc/portage/package.use
```

```
# emerge xorg-x11
```

Open /etc/X11/xorg.conf with your favorite text editor and edit it to enable DRI and GLX.

Code Listing 3.3: xorg.conf

```
...
Section "Module"
    Load "dri"
    Load "glx"
...
EndSection
...
Section "Device"
    Driver "radeon"
...
EndSection
...
Section "dri"
    Mode 0666
EndSection
```


If you are using a different driver, replace "radeon" with yours.

Changes to `modules.autoload.d`

You will need to add the module name that your card uses to `/etc/modules.autoload.d/kernel-2.6` to ensure that the module is loaded automatically when the system starts up.

Code Listing 3.4: Editing `/etc/modules.autoload.d/kernel-2.6`
(Change module name as required.)
`intel-agp`

Note: If you compiled `agpgart` as a module, you will also need to add it to `/etc/modules.autoload.d/kernel-2.6`.

4. Test 3D acceleration

Reboot to the new kernel

Reboot your computer to your new kernel and login as a normal user. It's time to see if you have direct rendering and how good it is.

Code Listing 4.1: Testing rendering

\$ startx

(No need to load modules for your driver or `agpgart`, even if you compiled them as a module.)
(They will be loaded automatically.)

\$ glxinfo | grep rendering

direct rendering: Yes

(If it says "No", you don't have 3D acceleration.)

\$ glxgears

(Test your frames per second (FPS) at the default size. The number should be)
(significantly higher than before installing `x11-drm`. Do this while the CPU is as idle as possible.)

5. Using the CVS sources

Warning: Don't do this if the package worked.

Do you need the CVS?

First you have to check whether the `x11-drm` package works. If it doesn't and you have checked your logs to verify it's not a configuration error, you might want to consider the CVS sources. There are also daily driver snapshots available if you do not wish to build the full CVS.

Do the CVS sources support your card?

Check the DRI supported cards list to see if the CVS supports your card. Even if it doesn't, but it supports a similar card, try it.

Follow the CVS Instructions

The DRI project has a document about CVS compiling themselves. Please read the document and follow the instructions up to the Installing for X.org part.

Install the CVS

Verify that the DRI kernel module(s) for your system were built:

Code Listing 5.1: Verification

```
# cd ~/DRI-CVS/build/xc/programs/Xserver/hw/xfree86/os-support/linux/drm/kernel; ls
```

For the 3dfx Voodoo, you should see tdfx.o. For the Matrox G200/G400, you should see mga.o. For the ATI Rage 128, you should see r128.o. For the ATI Radeon, you should see radeon.o. For the Intel i810, you should see i810.o. If the DRI kernel module(s) failed to build, you should verify that you're using the right version of the Linux kernel. The most recent kernels are not always supported.

Install over your X.org installation. You may wish to back up xorg-x11.

Code Listing 5.2: Backing up Xorg

```
# quickpkg xorg-x11
```

(This backs up your Xorg-X11 package.)

```
# make install
```

Follow the "Configure Xorg" section above.

To load the appropriate DRM module in your running kernel, copy the kernel module to /lib/modules/`uname -r`/kernel/drivers/char/drm/ then run modules-update and restart your X server. If you're not running the kernel you'll be using it in, instead of `uname -r`, use that kernel's name.

Warning: Make sure you first unload any older DRI kernel modules that might be already loaded. Note that some DRM modules require that the agpgart module be loaded first.

6. *Tweak your performance*

Get the most out of direct rendering

A few options may increase performance by up to 30 percent (or more) over the default. Set them in /etc/X11/xorg.conf. However, you will first need to check that your

motherboard and video card support these options.

First, let's see if your video card can support fast writes. We'll do this by closely inspecting the output from `lspci`. Specifically, we are looking at the "VGA compatible controller" information.

Code Listing 6.1: Video card check

```
# lspci -vv
```

```
01:00.0 VGA compatible controller: ATI Technologies Inc Radeon Mobility M6 LY (prog-if 00 [VGA])
```

```
...
```

```
Capabilities: [58] AGP version 2.0
```

```
    Status: RQ=48 Iso- ArqSz=0 Cal=0 SBA+ ITACoh- GART64- HTrans- 64bit- FW+ AGP3-
Rate=x1,x2,x4
```

While quite a lot of information is produced, we are looking for FW in the "Status" subsection of the AGP "Capabilities" section. If FW+ is present in the "Status" line, it means the card supports fast writes. We can now check if the motherboard supports fast writes.

Important: If you do not see FW+, but instead see FW-, you cannot enable fast writes in `xorg.conf`. Your card does not support fast writes.

Now let's make sure the motherboard supports fast writes as well. This time, look at the "Host bridge" section of your `lspci` output.

Code Listing 6.2: Motherboard check

```
# lspci -vv
```

```
00:00.0 Host bridge: Intel Corporation 82830 830 Chipset Host Bridge (rev 02)
```

```
...
```

```
Capabilities: [a0] AGP version 2.0
```

```
    Status: RQ=32 Iso- ArqSz=0 Cal=0 SBA+ ITACoh- GART64- HTrans- 64bit- FW+ AGP3-
Rate=x1,x2,x4
```

Again, examine the "Status" subsection of your AGP "Capabilities" section. Look for FW. If you see FW+, your motherboard supports fast writes.

Important: Remember, both your video card information and your motherboard information must show the same FW capability. If either device shows FW- in "Status", you cannot enable fast writes in `xorg.conf`.

Assuming that all has gone well and both your motherboard and video card support fast writes, let's enable this option in `/etc/X11/xorg.conf` and get the best performance out of your hardware.

Code Listing 6.3: `xorg.conf`

```
Section "Device"
```

```
    Option "AGPMode" "4"
```

```
    (This increased FPS from 609 to 618.)
```

```
    Option "AGPFastWrite" "True"
```

```
    (This had no measurable effect, but it may increase instability of your computer.)
```

```
    (You may also need to set it in your BIOS.)
```

Option "EnablePageFlip" "True"

(This improved FPS from 618 to 702. It also is "risky" but few people have reported problems.)

...

EndSection

Warning: Enabling AGPFastWrite on a VIA chipset will very likely cause your machine to lock up. VIA chipsets do not play nicely with fast writes, so use this setting at your own risk.

Note: Remember, if you want fast writes to work properly, you will have to first enable the appropriate option in your BIOS.

If you want to set even more features, check out the feature matrix on the DRI web site or the features listing on Sourceforge.

7. Troubleshooting

It doesn't work. I just recompiled my kernel or switched to a new one.

Whenever you rebuild your kernel or switch to another kernel, you'll have to rebuild the kernel module. Note that you don't need to remerge xorg-x11, but you will need to remerge x11-drm.

It doesn't work. I don't have rendering, and I can't tell why.

Try insmod radeon before you start the X server. Also, try building agpgart into the kernel instead of as a module.

When I startx, I get this error: "[drm] failed to load kernel module agpgart"

That's because you compiled agpgart into the kernel instead of as a module. Ignore it unless you're having problems.

Direct rendering doesn't work, and in /var/log/Xorg.0.log I have an error about driver version too low.

You aren't using the x11-drm driver. Check if you compiled DRM and the driver into the kernel; you shouldn't have.

I have a Radeon, and I want TV-Out.

Check out ati-gatos drivers. emerge -s gatos.

It doesn't work. My card is so incredibly new and cool that it isn't supported at all.

Try out the binary drivers. For ati-drivers, a listing is at http://www.schneider-digital.de/html/download_ati.php. If those don't support it, use fbdev. It's slow, but it works.

I have a PCI card and it doesn't work. Help!

In section "Device" enable ForcePCIMode.

Code Listing 7.1: Enabling ForcePCIMode
Option "ForcePCIMode" "True"

8. Acknowledgments

Christopher Webber for suggesting a troubleshooting question about changing or recompiling kernels

Steve, for suggesting consistency between the cases of dri and DRI in XF86Config

9. References

- <http://forums.gentoo.org/viewtopic.php?t=46681>
- <http://forums.gentoo.org/viewtopic.php?t=29264>
- <http://dri.freedesktop.org/>
- http://www.retinalburn.net/linux/dri_status.html

Gentoo и KDE

HOWTO по настройке KDE версии 3.x.x

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/kde-config.xml>

С версии: 1.0

1. Что же такое K Desktop Environment?

Проект

Проект **KDE** - проект открытого программного обеспечения посвящённый разработке **KDE**, графической рабочей среде для рабочих станций Linux и Unix. Разработкой занимаются несколько сотен разработчиков со всего мира, преданных разработке бесплатного программного обеспечения. Смотрите также Что из себя представляет проект **KDE**.

Программное обеспечение

K Desktop Environment - это простая рабочая среда построенная на хорошо продуманной прикладной структуре (application framework), которая предоставляет функциональную совместимость, drag 'n drop и многое другое. Кроме необходимых компонентов, KDE предоставляет готовые к использованию приложения для всех мыслимых задач: файловый менеджер, интернет-браузер, офис приложения, email-клиент и т.д. И всё это относится к KDE проекту.

KDE доступен более чем на 70 языках и охватывает огромное количество пользователей. Для тех, кто интересуется, есть много скриншотов. Для большей информации о KDE, читайте статью Что такое KDE? на KDE.org.

Сообщество

Существует много сайтов KDE сообществ. На KDEnews.org вы найдёте последние новости про KDE вообще. KDE developers.org специализируется на разработке KDE, а KDE-форум больше подходит для широкой массы. Больше ссылок можно найти на Странице семейства KDE.

2. Установка KDE

Что вам нужно?

Прежде всего, если вы заинтересованы в установке KDE (или поддержке kde), вам придётся убедиться, что переменная USE содержит kde и qt флаги. Для тех, кто

не знает, Qt - графическая библиотека элементов управления (graphical widget library), которую использует KDE.

Во-вторых, вам нужно подумать над тем, что вы хотите установить из дистрибутива KDE. Как ранее упоминалось, KDE очень богат возможностями и включает в себя много программ. Для этого вам не обязательно верить нашим словам -- сами проверьте :)

Листинг 2.1: Перечисляем все пакеты KDE
`# emerge --pretend kde | less`

Если вы не заинтересованы в установке всех этих пакетов, вы можете поставить лишь некоторые из них. Очевидно, что вам понадобится kdatabase, так как он содержит основные и необходимые программы. Следующая таблица показывает несколько пакетов, которые вы можете установить.

Таблица: Пакеты KDE.

Пакет	Описание
kdeaccessibility	Программы для людей с ограниченными возможностями разрабатываются Проектом KDE Accessibility
kdeadmin	Административные утилиты, такие как KCron (Планирование задач), KUser (Управление пользователями) и KDat (Управление резервными копиями)
kdeartwork	Некоторые программы, связанные с искусством, включая хранители экрана и темы. Также смотрите artist.kde.org .
kdeedu	Образовательные KDE приложения для школьников от 3 до 18 лет. Также смотрите Проект KDE Edu.
kdegames	KDE игры. Больше информации можно найти по ссылке KDE центр игр.
kdegraphics	Графические приложения для KDE, плюс KSnapshot (Программа для снятия скриншотов), KPaint (Простой графический редактор), Kpdf (Для просмотра PDF), KIconEdit (Icon Editor) и KPovModeler (Создание 3D моделей).
kde-i18n	Этот пакет включает поддержку для многих языков (также валют, форматов чисел и дат, ...) и документации. Больше информации по ссылке Проект KDE i18n.
kdemultimedia	Мультимедийные приложения, включая поддержку CD, MP3, DVD, форматов, звуковых и видео приложений. Больше информации по ссылке Проект KDE Multimedia website.
kdenetwork	Сетевые приложения, такие как kppp (Dial-In) и lisa (networking). Заметьте, что konqueror (файловый менеджер и веб-браузер) часть пакета kdatabase!
kdepim	Личные Информационные инструментальные средства

	Управления, такие как KOrganizer (Журнал), KAddressbook (Адресная книга), Kontact (Программное обеспечение для коллективной работы) и KMail (E-mail). Больше информации по ссылке Проект KDE PIM.
kdesdk	Основные инструменты разработки, включая KBabel (Утилита для перевода), KBugBuster (Front end для отслеживания ошибок KDE) и Kompare (GUI для просмотра разницы между файлами).
kdetoys	Различные безделушки, которыми можно позабавится, в ожидании, пока доставят вашу пиццу. В этом пакете вы найдёте такие апплеты, как eyesapplet и fifteenapplet, а также стильные утилиты как amog, которые ничего не делают, кроме как пожирают ресурсы :)
kdeutils	Графические системные утилиты как kcalc (Калькулятор), kdessh (SSH терминал), kfloppy (производит действия связанные с floppy), и другие.

К примеру, чтобы поставить KDE только с сетевыми и административными приложениями:

Листинг 2.2: Пример установки индивидуальных компонентов KDE

```
# emerge kdbase kdenetwork kdeadmin
```

Если вам интересно знать: компиляция KDE занимает очень много времени :)

Первые ощущения

Теперь давайте взглянем на результаты. Твоя мама наверное говорила тебе никогда не работать под root-ом. Ну что ж, последуем маминому совету и запустим KDE под пользователем. Зайдите под вашим пользователем и настройте вашу сессию так, чтобы загружался KDE после выполнения команды startx. Вы можете это сделать добавив строчку `exec startkde` в `~/.xinitrc`:

Листинг 2.3: Настраиваем локальную сессию

```
$ echo "exec startkde" > ~/.xinitrc
```

Теперь выполните команду startx для загрузки вашей графической среды.

Листинг 2.4: Запускаем KDE

```
$ startx
```

Вас встретит приложение под названием **KPersonalizer**. Поздравляем, теперь давайте посмотрим, как можно настроить KDE...

3. Настройка KDE

KPersonalizer

KPersonalizer - приложение, которое настроит KDE за вас. Это очень полезный мастер, который помогает быстро изменить KDE под ваши нужды. KPersonalizer автоматически появится при первой загрузке KDE.

Первое, о чём спросит вас KPersonalizer будет страна и язык на ваш выбор. Так как мы не устанавливали никаких языковых пакетов, выбор языков будет небольшим -- очевидно в списке будет только английский. Не волнуйтесь, мы сменим язык позже (если вы конечно захотите этого).

Потом вам нужно будет выбрать System Behaviour. Эта настройка включает в себя активацию окна, выбор мыши, и другое. При выборе определённого поведения, можно увидеть его описание. Если вы не уверены что выбрать, не паникуйте -- всегда можно будет поменять настройки.

Далее KPersonalizer спросит о количестве разнообразных эффектов. Чем больше эффектов вы активируете, тем моднее будет KDE, но и процессор будет более загружен. Однако, не стоит так переживать из-за этого -- на компьютере с процессором в 600 Mhz с 128 Mb памяти разрешение всех эффектов не особо не повлияет на быстродействие системы.

Наконец, остаётся выбрать стиль. Стиль определяет декорацию окон, тему, вид кнопок, и другое. Попробуйте несколько стилей, чтобы выбрать тот, который вам понравится. Я ещё не говорил вам, что KDE полностью настраиваемый?

Откиньтесь на спинку кресла и наслаждайтесь -- запустится KDE и вас встретит милая, чистая и функциональная интерактивная рабочая среда.

Установка языковых пакетов

Если ваш родной язык не английский или вы хотите работать в KDE на другом языке, тогда читайте дальше. Мы установим языковые пакеты для языков, которые вам нужны.

Все языковые пакеты содержатся в одном **kde-i18n** пакете. Чтобы поставить только нужные пакеты, а не все, нужно присвоить значение переменной LINGUAS. Рекомендуется прописать переменную в /etc/make.conf.

Листинг 3.1: Прописываем переменную LINGUAS в /etc/make.conf

```
# nano -w /etc/make.conf
(К примеру, мы поставим языковые пакеты для русского (ru)
и французского (fr))
LINGUAS="ru fr"
```

Теперь выполните команду `emerge kde-i18n`. Когда установка закончится, запускайте KDE и KDE центр управления (Control Center) (K-menu > Settings >

Control Center). Это приложение, где можно изменить намного больше настроек KDE, чем в KPersonalizer.

Чтобы изменить язык, идите в Regional & Accessibility, Country/Region & Languages. Потом добавьте язык на ваш выбор и перезагрузите KDE.

Графический вход в систему

Если вы хотите использовать kdm как менеджер графического входа в систему (что значит, что вам не придётся каждый раз набивать на клавиатуре команду startx) сначала загляните в /etc/X11/Sessions, чтобы узнать, как называется KDE сессия:

Листинг 3.2: Узнаем название KDE сессии

```
# ls /etc/X11/Sessions
Xsession fluxbox kde-3.2.1
```

В примере выше, KDE сессия называется kde-3.2.1. Давайте добавим её в /etc/rc.conf в переменную XSESSION. Также измените значение переменной DISPLAYMANAGER на kdm.

Листинг 3.3: Изменяем значение переменной XSESSION в /etc/rc.conf

```
# nano -w /etc/rc.conf
(Измените две следующие переменные)
XSESSION="kde-3.2.1"
DISPLAYMANAGER="kdm"
```

Завершаем добавив **xdm** в **default runlevel**:

Листинг 3.4: Добавляем xdm в default runlevel

```
# rc-update add xdm default
```

После перезагрузки системы, как графический вход в систему будет использоваться KDM.

Способы переключения раскладок клавиатуры в KDE (3в1)

Ссылка на оригинал: <http://gentoo.blog.ru/3638115.html>

Авторы: [gentoo](#)

С версии: 1.4

Дата: 18.12.2007

Читая [Планету блогов о *NIX](#) (нагло пиарю) наткнулся на несколько постов о настройке переключения раскладок клавиатуры. Кто-то советует править `xorg.conf`, кто-то - поставить `kkbswitch`, самые пытливые заставляют завестись встроенное средство переключения раскладок в KDE. Сам использовал в основном `kkbswitch` (сейчас включил встроенный `kxkb`), только в SuSE переключался средствами `xorg.conf` - не нашёл `kkbswitch` в репах. Недостаток использования переключения из `xorg.conf` - раскладка для всех окон всегда одна. Встроенная переключалка KDE работает неплохо, если настроить, но... в русской раскладке не срабатывают хоткеи (решено, спасибо комраду Fallen). У `kkbswitch` изъянов мной замечено не было, с тремя языками работает (кого волнует этот вопрос). Кстати, он читает `xorg.conf` и соответствующим образом настраивает себя, так что для начала потребуется настроить `xorg.conf`.

1. /etc/X11/xorg.conf

```
Section "InputDevice"
    Identifier "Keyboard0"
    Driver "kbd"
    Option "XkbRules" "xorg"
    Option "XkbModel" "pc104"
    Option "XkbLayout" "us,ru"
    Option "XkbOptions" "grp:ctrl_shift_toggle"
    Option "XkbVariant" ",winkeys"
    Option "AutoRepeat" "250 30"
EndSection
```

Добавление: Option "XkbOptions" "grp_led:caps"
будет включать/выключать индикатор CapsLock'a при переключении языков.

2. Поставить [kkbswitch](#). Берёт инфо о языках и клавишах переключения из `xorg.conf`. Пример для 3х языков:

```
Section "InputDevice"
    Identifier "Keyboard0"
    Driver "kbd"
    Option "XkbRules" "xorg"
    Option "XkbModel" "pc104"
    Option "XkbLayout" "us,de,ru"
    Option "XkbOptions" "grp:ctrl_shift_toggle"
    Option "XkbVariant" ",,winkeys"
    Option "AutoRepeat" "250 30"
EndSection
```

Перед `winkeys` в данном случае две запятые (т.к. русский язык теперь третий по

счёту). Языки переключаются по-кругу по Ctrl+Shift.

Один нюанс - в этом случае при трёх языках у меня почему-то не срабатывает переход в текстовую консоль по Ctrl+Alt+F1-6, хм.

3. Включение встроенного средства переключения раскладок клавиатуры KDE.

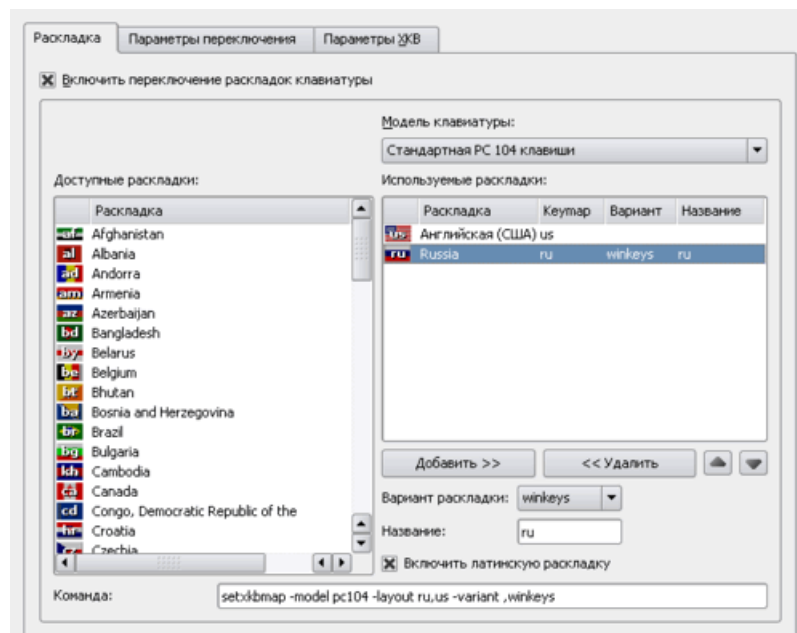
Достичь окончательного (?) просветления в этом вопросе помог пост "[Переключение раскладок KDE в комиксах](#)". В настройке переключения есть как минимум две неочевидные вещи. Рассмотрим два варианта: настройка для двух языков и для трёх, а в качестве кнопок переключения - Ctrl+Shift (как наиболее популярный вариант у иммигрантов с Windows).

а) Настройка для двух языков (Англ/Рус).

Центр Управления - Региональные и спец. возможности - Раскладка клавиатуры

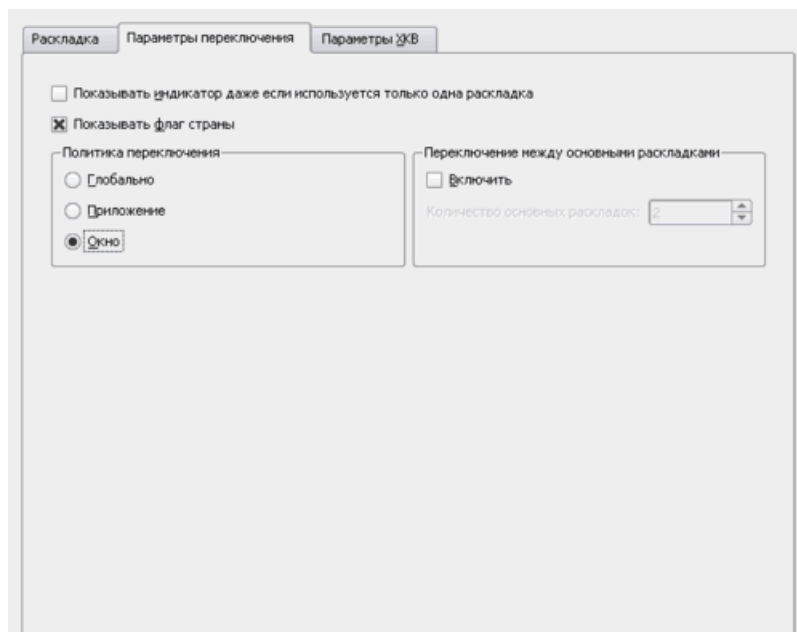
1я закладка (Раскладка):

Добавляем две раскладки - Англ и Русскую, причём для русской ставим галку "Включить латинскую раскладку" (иначе в русской раскладке не будут работать хоткеи):



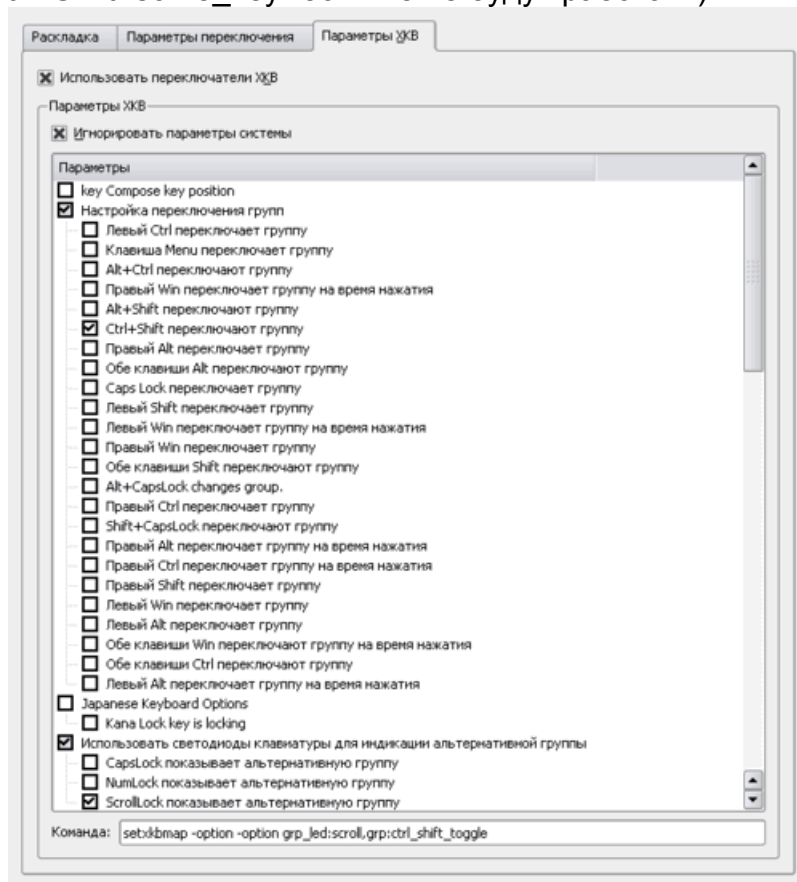
2я закладка (Параметры переключения):

по вкусу



Зя закладка (Параметры ХКВ):

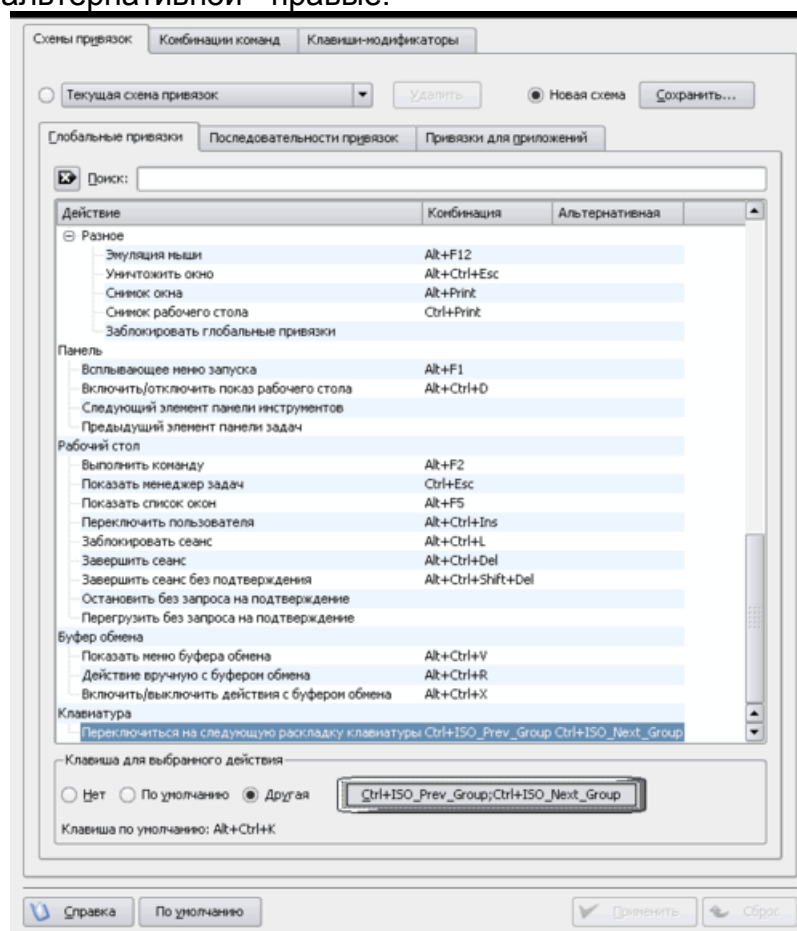
Выбираем "Ctrl+Shift переключает группу" и, ниже, можно выбрать "ScrollLock показывает альтернативную группу" (учтите, что хоткеи, в которых используется комбинация "Ctrl+Shift+some_key" больше не будут работать).



Первая неочевидная вещь заключается в том, что "Настройка переключения групп" на данной закладке - это отнюдь не переключатель раскладок (!), это переключатель групп языков в конкретной раскладке. Поэтому идём к пункту четыре:

4й шаг: Центр Управления - Региональные и спец. возможности - Комбинации клавиш:

В самом низу нужно назначить клавиши для "Переключиться на следующую раскладку клавиатуры" - здесь и задаётся желаемая комбинация, в нашем случае - попробуем задать Ctrl+Shift. Для основной комбинации нажимаем левые Ctrl+Shift, для альтернативной - правые.



Скорей всего вы увидите "Ctrl+ISO_Prev_Group; Ctrl+ISO_Next_Group", проверяем переключаются ли раскладки. Не переключаются? И у меня нет. В справке проблема описана так: "Если kxkb не переключает с помощью комбинации клавиш, когда вы переключаетесь на какую либо раскладку, но переключает при использовании мыши, возможно ваша раскладка не содержит клавишу, назначенную в комбинации".

Можно исправить вручную так:

\$ nano ~/.kde/share/config/kdeglobals

Найти строку "Switch to Next Keyboard Layout=..." и изменить её на:
Switch to Next Keyboard Layout=Ctrl+Shift_L;Ctrl+Shift_R

(кстати, для варианта переключения по левой клавише "Win" такие ухищрения не будут нужны, и хоткеи "Ctrl+Shift+some_key" будут работать)

Идём снова в "Раскладка клавиатуры", отключаем и вновь включаем:

[x] Включить переключение раскладок клавиатуры

Всё, теперь должно работать.

б) Настройка для трёх языков (Англ/Рус/Укр). Настраивается практически аналогично:

1я закладка (Раскладка):

Добавляем три раскладки - Англ, Русскую, Украинскую. Для русской и украинской ставится галка "Включить латинскую раскладку" (здесь это важно не столько для отработки хоткеев, сколько для удобства работы с тремя языками).

2я закладка (Параметры переключения):

пока не включаем опцию "Переключение между основными раскладками".

3я закладка (Параметры ХКВ):

То же самое что и для варианта двух раскладок, без каких-либо изменений (если настраиваете на три раскладки после того, как настраивали для двух, то, возможно придётся отключить и вновь включить здесь "Использовать переключатели ХКВ").

4й шаг:

\$ nano ~/.kde/share/config/kdeglobals

Найти строку "Switch to Next Keyboard Layout=..." и изменить её на:

Switch to Next Keyboard Layout=Ctrl+Shift_L

(заметьте, альтернативная комбинация не задана)

Идём снова в "Раскладка клавиатуры", отключаем и вновь включаем:

[x] Включить переключение раскладок клавиатуры

Смотрим, что получилось. А получиться должно вот что:

При нажатии левых Ctrl+Shift изменяется текущая раскладка (и соотв. флажок в трее изменяется по кругу - Англ-Рус-Укр).

При нажатии правых Ctrl+Shift изменяется текущая группа раскладки (флажок остаётся прежним). Вот так выглядит структура раскладок и групп:

Английская раскладка

- англ. группа

Русская раскладка

- русская группа

- англ. группа

Украинская раскладка

- украинская группа

- англ. группа

То есть, переключившись левыми Ctrl+Shift на русскую раскладку, для набора английских букв не надо снова дважды жать левые Ctrl+Shift, можно правыми переключить текущую группу на английскую, набрать что требовалось, и правыми

же переключиться опять на русский язык.

в) Настройка для трёх языков (Англ/Рус/Укр) с использованием опции "Основных раскладок".

Всё то же самое, только:

2я закладка (Параметры переключения):

Включаем опцию [x] "Переключение между основными раскладками".

Количество основных раскладок: 2

При таком варианте по левому Ctrl+Shift будет происходить переключение только между Англ и Рус раскладками. На третий язык (Укр) можно переключиться только выбрав его мышкой. Причём после выбора будет происходить переключение между Англ и Украинскими раскладками (для всей системы или только для текущего окна - в зависимости от настройки).

Вот, наверное, и всё.

Какой вариант удобнее - каждый решает для себя сам.

PS: в Fedora8 по-умолчанию могут отсутствовать флажки языков, можно взять из другого дистрибутива по адресу /usr/share/locale/l10n/ru/flag.png (.../us/flag.png), либо, цитирую: "Небольшое уточнение: в Fedora 8 есть флажки стран, только входят они в пакет wordtrans и находятся в /usr/share/wordtrans/flags/".

PPS: включение этого средства отключает настройки раскладок xorg.conf, при выключении для возврата к использованию настроек xorg.conf необходимо перезапустить X'сы.

Для переключения языков в текстовой консоли в Gentoo редактируется файл /etc/conf.d/keymaps.

Gentoo KDE 4.0 Guide (Рус.)

Ссылка на оригинал: <http://www.gentoo.org/proj/en/desktop/kde/kde4.xml>

Авторы: [Ingmar Vanhassel](#) , [Mark Kowarsky](#)

Перевод: [Батогов Е.В. а.к.а. JohnBat26](#)

С версии: 1.4

Обновлено в: 1.5

Дата: 28.02.2008

Дата окончания перевода: 27.03.2008

1. Установка

Удаление всех, предшествующих официальной версии, ebuilds.

Если Вы использовали [the genkdesvn overlay](#), тогда рекомендуется удалить все версии KDE, предшествующие официальной версии, перед обновлением. Большинство пользователей достаточно будет выполнить: **layman -d kde**. Кроме того, если Вы изменяли какие-либо файлы, связанные с KDE, в каталоге `/etc/env.d`, тогда Вам необходимо вручную удалить их.

Включение необходимых USE-флагов

KDE 4.0 требует не просто Qt версии 4 (`x11-libs/qt:4`), но и то чтобы она была собрана с минимальным набором USE-флагов. Добавьте следующую строку в Ваш файл `/etc/portage/package.use`.

Листинг 1.1: Установка необходимых USE-флагов для `x11-libs/qt:4`

```
x11-libs/qt:4 accessibility dbus gif jpeg png qt3support ssl zlib
```

Удаление маскирования и прописывание ключевых слов

В настоящее время (24.01.2008) KDE версии 4.0.0. жестко замаскирован (**hard masked**), поэтому Вы должны удалить эти маски перед его установкой. Пользователям с `ACCEPT_KEYWORDS="~x86"` или `ACCEPT_KEYWORDS="~amd64"`, нет необходимости выполнять какие-либо другие шаги. Всем остальным пользователям необходимо поместить необходимые пакеты в файл `package.keywords`.

Так как `/etc/portage/package.unmask/` и `/etc/portage/package.keywords/` могут быть директориями, Вы можете просто сохранить следующие примеры файлов в эти директории для снятия маскирования и прописывания ключевых слов для KDE 4.0.0.

Листинг 1.2 Пример `package.unmask` :

```
# Unmask KDE 4.0.0
kde-base/amor:kde-4
kde-base/ark:kde-4
kde-base/blinken:kde-4
kde-base/bovo:kde-4
```

Gentoo и KDE

```
kde-base/cervisia:kde-4
kde-base/dolphin:kde-4
kde-base/drkonqi:kde-4
kde-base/gwenview:kde-4
kde-base/juk:kde-4
kde-base/kalgebra:kde-4
kde-base/kalzium:kde-4
kde-base/kamera:kde-4
kde-base/kanagram:kde-4
kde-base/kappfinder:kde-4
kde-base/kapptemplate:kde-4
kde-base/kate:kde-4
kde-base/katomic:kde-4
kde-base/kbattleship:kde-4
kde-base/kblackbox:kde-4
kde-base/kbounce:kde-4
kde-base/kbruch:kde-4
kde-base/kbugbuster:kde-4
kde-base/kcachegrind:kde-4
kde-base/kcalc:kde-4
kde-base/kcharselect:kde-4
kde-base/kcheckpass:kde-4
kde-base/kcminit:kde-4
kde-base/kcmshell:kde-4
kde-base/kcolorchooser:kde-4
kde-base/kcontrol:kde-4
kde-base/kcron:kde-4
kde-base/kde:kde-4
kde-base/kdeaccessibility:kde-4
kde-base/kdeaccessibility-colorschemes:kde-4
kde-base/kdeaccessibility-iconthemes:kde-4
kde-base/kdeaccessibility-meta:kde-4
kde-base/kdeaccounts-plugin:kde-4
kde-base/kdeadmin:kde-4
kde-base/kdeadmin-meta:kde-4
kde-base/kdeartwork:kde-4
kde-base/kdeartwork-colorschemes:kde-4
kde-base/kdeartwork-emoticons:kde-4
kde-base/kdeartwork-iconthemes:kde-4
kde-base/kdeartwork-ksscreensaver:kde-4
kde-base/kdeartwork-kworldclock:kde-4
kde-base/kdeartwork-meta:kde-4
kde-base/kdeartwork-sounds:kde-4
kde-base/kdeartwork-styles:kde-4
kde-base/kdeartwork-wallpapers:kde-4
kde-base/kdebase:kde-4
kde-base/kdebase-data:kde-4
kde-base/kdebase-kioslaves:kde-4
kde-base/kdebase-meta:kde-4
kde-base/kdebase-startkde:kde-4
kde-base/kdebugdialog:kde-4
kde-base/kdeedu:kde-4
kde-base/kdeedu-meta:kde-4
kde-base/kdegames:kde-4
kde-base/kdegames-meta:kde-4
kde-base/kdegraphics:kde-4
kde-base/kdegraphics-meta:kde-4
kde-base/kde-l10n:kde-4
kde-base/kdelibs:kde-4
kde-base/kde-menu:kde-4
kde-base/kde-menu-icons:kde-4
kde-base/kde-meta:kde-4
kde-base/kdemultimedia:kde-4
kde-base/kdemultimedia-kioslaves:kde-4
kde-base/kdemultimedia-meta:kde-4
kde-base/kdenetwork:kde-4
kde-base/kdenetwork-filesharing:kde-4
kde-base/kdenetwork-meta:kde-4
kde-base/kdepasswd:kde-4
kde-base/kdepimlibs:kde-4
kde-base/kdesdk:kde-4
kde-base/kdesdk-kioslaves:kde-4
kde-base/kdesdk-meta:kde-4
kde-base/kdesdk-misc:kde-4
```

kde-base/kdesdk-scripts:kde-4
kde-base/kdessh:kde-4
kde-base/kdesu:kde-4
kde-base/kdetoys:kde-4
kde-base/kdetoys-meta:kde-4
kde-base/kdeutils:kde-4
kde-base/kdeutils-meta:kde-4
kde-base/kde-wallpapers:kde-4
kde-base/kdf:kde-4
kde-base/kdialog:kde-4
kde-base/kdm:kde-4
kde-base/kdnssd:kde-4
kde-base/keditbookmarks:kde-4
kde-base/kfile:kde-4
kde-base/kfind:kde-4
kde-base/kfloppy:kde-4
kde-base/kfourinline:kde-4
kde-base/kgamma:kde-4
kde-base/kgeography:kde-4
kde-base/kget:kde-4
kde-base/kgoldrunner:kde-4
kde-base/kgpg:kde-4
kde-base/khangman:kde-4
kde-base/khelpcenter:kde-4
kde-base/khotkeys:kde-4
kde-base/kig:kde-4
kde-base/kioclient:kde-4
kde-base/kiriki:kde-4
kde-base/kiten:kde-4
kde-base/kjots:kde-4
kde-base/kjumpingcube:kde-4
kde-base/klettres:kde-4
kde-base/klines:kde-4
kde-base/klipper:kde-4
kde-base/kmag:kde-4
kde-base/kmahjongg:kde-4
kde-base/kmenuedit:kde-4
kde-base/kmilo:kde-4
kde-base/kmimetypefinder:kde-4
kde-base/kmines:kde-4
kde-base/kmix:kde-4
kde-base/kmousetool:kde-4
kde-base/kmouth:kde-4
kde-base/kmpplot:kde-4
kde-base/knetattach:kde-4
kde-base/knetwalk:kde-4
kde-base/knetworkconf:kde-4
kde-base/knewsticker:kde-4
kde-base/knewstuff:kde-4
kde-base/knotify:kde-4
kde-base/kolf:kde-4
kde-base/kolourpaint:kde-4
kde-base/kompare:kde-4
kde-base/konqueror:kde-4
kde-base/konquest:kde-4
kde-base/konsole:kde-4
kde-base/kopete:kde-4
kde-base/kpasswdserver:kde-4
kde-base/kpat:kde-4
kde-base/kpercentage:kde-4
kde-base/kppp:kde-4
kde-base/kquitapp:kde-4
kde-base/krdc:kde-4
kde-base/kreadconfig:kde-4
kde-base/kreversi:kde-4
kde-base/krfb:kde-4
kde-base/kruler:kde-4
kde-base/krunner:kde-4
kde-base/ksame:kde-4
kde-base/kscd:kde-4
kde-base/kscreensaver:kde-4
kde-base/kshisen:kde-4
kde-base/ksmserver:kde-4
kde-base/ksnapshot:kde-4

Gentoo и KDE

```
kde-base/kspaceduel:kde-4
kde-base/ksplash:kde-4
kde-base/ksquares:kde-4
kde-base/kstars:kde-4
kde-base/kstart:kde-4
kde-base/kstartupconfig:kde-4
kde-base/kstyles:kde-4
kde-base/ksudoku:kde-4
kde-base/ksysguard:kde-4
kde-base/ksystraycmd:kde-4
kde-base/kteatime:kde-4
kde-base/ktimer:kde-4
kde-base/ktimezoned:kde-4
kde-base/ktip:kde-4
kde-base/ktouch:kde-4
kde-base/ktraderclient:kde-4
kde-base/kttsd:kde-4
kde-base/ktuberling:kde-4
kde-base/kturtle:kde-4
kde-base/ktux:kde-4
kde-base/kuiserver:kde-4
kde-base/kuiviewer:kde-4
kde-base/kurifilter-plugins:kde-4
kde-base/kuser:kde-4
kde-base/kwallet:kde-4
kde-base/kweather:kde-4
kde-base/kwin:kde-4
kde-base/kwordquiz:kde-4
kde-base/kworldclock:kde-4
kde-base/kwrite:kde-4
kde-base/libkcdcd:kde-4
kde-base/libkcompactdisc:kde-4
kde-base/libkdeedu:kde-4
kde-base/libkdegames:kde-4
kde-base/libkmahjongg:kde-4
kde-base/libkonq:kde-4
kde-base/libkscan:kde-4
kde-base/libkworkspace:kde-4
kde-base/libplasma:kde-4
kde-base/libtaskmanager:kde-4
kde-base/lilo-config:kde-4
kde-base/lskat:kde-4
kde-base/marble:kde-4
kde-base/nepomuk:kde-4
kde-base/nsplugins:kde-4
kde-base/okular:kde-4
kde-base/parley:kde-4
kde-base/phonon:kde-4
kde-base/plasma:kde-4
kde-base/poxml:kde-4
kde-base/qimageblitz
kde-base/secpolicy:kde-4
kde-base/solid:kde-4
kde-base/soliduiserver:kde-4
kde-base/strigi-analyzer:kde-4
kde-base/superkaramba:kde-4
kde-base/svgpart:kde-4
kde-base/sweeper:kde-4
kde-base/systemsettings:kde-4
kde-base/umbrello:kde-4
```

Листинг 1.3 Пример `package.keywords` :

```
# Masked qca-2.0 and dependencies for further testing, bug #129536
app-crypt/qca:2 **
app-crypt/qca-cyrus:2 **
app-crypt/qca-gnupg:2 **
app-crypt/qca-logger:2 **
app-crypt/qca-openssl:2 **
app-crypt/qca-pkcs11:2 **

# KDE 4 dependencies
app-misc/strigi **
```

```

dev-libs/libzip **
dev-libs/soprano **
sci-libs/indilib **
sci-mathematics/gmm **

# Unmask KDE 4.0.0
kde-base/amar:kde-4 **
kde-base/ark:kde-4 **
kde-base/blinken:kde-4 **
kde-base/bovo:kde-4 **
kde-base/cervisia:kde-4 **
kde-base/dolphin:kde-4 **
kde-base/drkonqi:kde-4 **
kde-base/gwenview:kde-4 **
kde-base/juk:kde-4 **
kde-base/kalgebra:kde-4 **
kde-base/kalzium:kde-4 **
kde-base/kamera:kde-4 **
kde-base/kanagram:kde-4 **
kde-base/kapfinder:kde-4 **
kde-base/kapptemplate:kde-4 **
kde-base/kate:kde-4 **
kde-base/katomic:kde-4 **
kde-base/kbattleship:kde-4 **
kde-base/kblackbox:kde-4 **
kde-base/kbounce:kde-4 **
kde-base/kbruch:kde-4 **
kde-base/kbugbuster:kde-4 **
kde-base/kcachegrind:kde-4 **
kde-base/kcalc:kde-4 **
kde-base/kcharset:kde-4 **
kde-base/kcheckpass:kde-4 **
kde-base/kcminit:kde-4 **
kde-base/kcmshell:kde-4 **
kde-base/kcolorchooser:kde-4 **
kde-base/kcontrol:kde-4 **
kde-base/kcron:kde-4 **
kde-base/kde:kde-4 **
kde-base/kdeaccessibility:kde-4 **
kde-base/kdeaccessibility-colorschemes:kde-4 **
kde-base/kdeaccessibility-iconthemes:kde-4 **
kde-base/kdeaccessibility-meta:kde-4 **
kde-base/kdeaccounts-plugin:kde-4 **
kde-base/kdeadmin:kde-4 **
kde-base/kdeadmin-meta:kde-4 **
kde-base/kdeartwork:kde-4 **
kde-base/kdeartwork-colorschemes:kde-4 **
kde-base/kdeartwork-emoticons:kde-4 **
kde-base/kdeartwork-iconthemes:kde-4 **
kde-base/kdeartwork-ksscreensaver:kde-4 **
kde-base/kdeartwork-kworldclock:kde-4 **
kde-base/kdeartwork-meta:kde-4 **
kde-base/kdeartwork-sounds:kde-4 **
kde-base/kdeartwork-styles:kde-4 **
kde-base/kdeartwork-wallpapers:kde-4 **
kde-base/kdebase:kde-4 **
kde-base/kdebase-data:kde-4 **
kde-base/kdebase-kioslaves:kde-4 **
kde-base/kdebase-meta:kde-4 **
kde-base/kdebase-startkde:kde-4 **
kde-base/kdebugdialog:kde-4 **
kde-base/kdeedu:kde-4 **
kde-base/kdeedu-meta:kde-4 **
kde-base/kdegames:kde-4 **
kde-base/kdegames-meta:kde-4 **
kde-base/kdegraphics:kde-4 **
kde-base/kdegraphics-meta:kde-4 **
kde-base/kde-l10n:kde-4 **
kde-base/kdelibs:kde-4 **
kde-base/kde-menu:kde-4 **
kde-base/kde-menu-icons:kde-4 **
kde-base/kde-meta:kde-4 **
kde-base/kdemultimedia:kde-4 **
kde-base/kdemultimedia-kioslaves:kde-4 **

```

Gentoo и KDE

```
kde-base/kdemultimedia-meta:kde-4 **
kde-base/kdenetwork:kde-4 **
kde-base/kdenetwork-filessharing:kde-4 **
kde-base/kdenetwork-meta:kde-4 **
kde-base/kdepasswd:kde-4 **
kde-base/kdepimlibs:kde-4 **
kde-base/kdesdk:kde-4 **
kde-base/kdesdk-kioslaves:kde-4 **
kde-base/kdesdk-meta:kde-4 **
kde-base/kdesdk-misc:kde-4 **
kde-base/kdesdk-scripts:kde-4 **
kde-base/kdessh:kde-4 **
kde-base/kdesu:kde-4 **
kde-base/kdetoys:kde-4 **
kde-base/kdetoys-meta:kde-4 **
kde-base/kdeutils:kde-4 **
kde-base/kdeutils-meta:kde-4 **
kde-base/kde-wallpapers:kde-4 **
kde-base/kdf:kde-4 **
kde-base/kdialog:kde-4 **
kde-base/kdm:kde-4 **
kde-base/kdnssd:kde-4 **
kde-base/keditbookmarks:kde-4 **
kde-base/kfile:kde-4 **
kde-base/kfind:kde-4 **
kde-base/kfloppy:kde-4 **
kde-base/kfourinline:kde-4 **
kde-base/kgamma:kde-4 **
kde-base/kgeography:kde-4 **
kde-base/kget:kde-4 **
kde-base/kgoldrunner:kde-4 **
kde-base/kgpg:kde-4 **
kde-base/khangman:kde-4 **
kde-base/khelpcenter:kde-4 **
kde-base/khotkeys:kde-4 **
kde-base/kig:kde-4 **
kde-base/kioclient:kde-4 **
kde-base/kiriki:kde-4 **
kde-base/kiten:kde-4 **
kde-base/kjots:kde-4 **
kde-base/kjumpingcube:kde-4 **
kde-base/klettres:kde-4 **
kde-base/klines:kde-4 **
kde-base/klipper:kde-4 **
kde-base/kmag:kde-4 **
kde-base/kmahjongg:kde-4 **
kde-base/kmenuedit:kde-4 **
kde-base/kmilo:kde-4 **
kde-base/kmimetypefinder:kde-4 **
kde-base/kmines:kde-4 **
kde-base/kmix:kde-4 **
kde-base/kmousetool:kde-4 **
kde-base/kmouth:kde-4 **
kde-base/kmplot:kde-4 **
kde-base/knetattach:kde-4 **
kde-base/knetwalk:kde-4 **
kde-base/knetworkconf:kde-4 **
kde-base/knewsticker:kde-4 **
kde-base/knewstuff:kde-4 **
kde-base/knotify:kde-4 **
kde-base/kolf:kde-4 **
kde-base/kolourpaint:kde-4 **
kde-base/kompare:kde-4 **
kde-base/konqueror:kde-4 **
kde-base/konquest:kde-4 **
kde-base/konsole:kde-4 **
kde-base/kopete:kde-4 **
kde-base/kpasswdserver:kde-4 **
kde-base/kpat:kde-4 **
kde-base/kpercentage:kde-4 **
kde-base/kppp:kde-4 **
kde-base/kquitapp:kde-4 **
kde-base/krdc:kde-4 **
kde-base/kreadconfig:kde-4 **
```

```

kde-base/kreversi:kde-4 **
kde-base/krfb:kde-4 **
kde-base/kruler:kde-4 **
kde-base/krunner:kde-4 **
kde-base/ksame:kde-4 **
kde-base/kscd:kde-4 **
kde-base/kscreensaver:kde-4 **
kde-base/kshisen:kde-4 **
kde-base/ksmserver:kde-4 **
kde-base/ksnapshot:kde-4 **
kde-base/kspaceduel:kde-4 **
kde-base/ksplash:kde-4 **
kde-base/ksquares:kde-4 **
kde-base/kstars:kde-4 **
kde-base/kstart:kde-4 **
kde-base/kstartupconfig:kde-4 **
kde-base/kstyles:kde-4 **
kde-base/ksudoku:kde-4 **
kde-base/ksysguard:kde-4 **
kde-base/ksystraycmd:kde-4 **
kde-base/kteatime:kde-4 **
kde-base/ktimer:kde-4 **
kde-base/ktimezoned:kde-4 **
kde-base/ktip:kde-4 **
kde-base/ktouch:kde-4 **
kde-base/ktraderclient:kde-4 **
kde-base/ktttsd:kde-4 **
kde-base/ktuberling:kde-4 **
kde-base/kturtle:kde-4 **
kde-base/ktux:kde-4 **
kde-base/kuiserver:kde-4 **
kde-base/kuiviewer:kde-4 **
kde-base/kurifilter-plugins:kde-4 **
kde-base/kuser:kde-4 **
kde-base/kwallet:kde-4 **
kde-base/kweather:kde-4 **
kde-base/kwin:kde-4 **
kde-base/kwordquiz:kde-4 **
kde-base/kworldclock:kde-4 **
kde-base/kwrite:kde-4 **
kde-base/libkcddb:kde-4 **
kde-base/libkcompactdisc:kde-4 **
kde-base/libkdeedu:kde-4 **
kde-base/libkdegames:kde-4 **
kde-base/libkmahjongg:kde-4 **
kde-base/libkonq:kde-4 **
kde-base/libkscan:kde-4 **
kde-base/libkworkspace:kde-4 **
kde-base/libplasma:kde-4 **
kde-base/libtaskmanager:kde-4 **
kde-base/lilo-config:kde-4 **
kde-base/lskat:kde-4 **
kde-base/marble:kde-4 **
kde-base/nepomuk:kde-4 **
kde-base/nsplugins:kde-4 **
kde-base/okular:kde-4 **
kde-base/parley:kde-4 **
kde-base/phonon:kde-4 **
kde-base/plasma:kde-4 **
kde-base/poxml:kde-4 **
kde-base/qimageblitz-0.0.4 **
kde-base/secpolicy:kde-4 **
kde-base/solid:kde-4 **
kde-base/soliduiserver:kde-4 **
kde-base/strigi-analyzer:kde-4 **
kde-base/superkaramba:kde-4 **
kde-base/svgpart:kde-4 **
kde-base/sweeper:kde-4 **
kde-base/systemsettings:kde-4 **
kde-base/umbrello:kde-4 **

```

Непосредственно установка

Теперь Вы можете запустить одну из следующих команд `emerge`, для установки KDE.

- Первая установит Вам все из состава KDE 4.0.0, собранное из отдельных ebuilds.
- Вторая установит Вам минимальное окружение KDE 4.0.0 как базу для добавления только тех программ, которые Вы хотите.

Листинг 1.4: Установка KDE 4.0.0

```
emerge -avD kde-base/kde-meta:kde-4
emerge -avD kde-base/kdebase-startkde:kde-4
```

2. Изменения по сравнению с KDE 3.5

KDE 4 вносит впечатляющие изменения в архитектуру рабочего стола в сравнении с KDE 3. Одним из таких изменений является переход от autotools к `stake`. Это означает, что теперь Вы можете видеть прогресс компиляции KDE в процентах.

Скрипт startkde

Не рекомендуется позволять KDE 3.5 и KDE 4.0 использовать одни и те же файлы конфигурации. Для предотвращения этого скрипты **startkde** перезапишут символическую ссылку `~/kde` во время старта KDE. Это означает, что когда Вы будете находиться в сессии KDE 4.0, Ваша конфигурация для KDE 3.5 будет недоступна и наоборот.

Исходя из этого, Вам не следует запускать две разных версии KDE под одним пользователем в одно время. (Войдя в KDE 4.0, KDE 3.5 остается запущенной в фоне, под тем же пользователем !)

Выбор менеджера входа в систему

По умолчанию, самая последняя версия `kde-base/kdm` которая у Вас установлена, будет запускаться в момент входа в систему. Для настройки этого, откройте файл `/etc/conf.d/xdm` в Вашем любимом текстовом редакторе и измените переменную **DISPLAYMANAGER** на `kdm-3.5` или `kdm-4.0`. Вы можете посмотреть возможные версии `kdm`, запустив `ls /usr/kde` в консоли.

Листинг 2.1: Добавьте следующее в файл `/etc/conf.d/xdm`

```
DISPLAYMANAGER=kdm-3.5
```

3. FAQ (Часто задаваемые вопросы)

В: Могу я установить KDE 4.0 и KDE 3.5 одновременно?

О: Да, обе версии помещаются по слотам, и могут быть установлены одновременно. Обе версии KDE используют различные `KDEHOME`, таким образом Вы можете попробовать KDE 4.0 под тем же пользователем, без перезаписи Ваших настроек из KDE 3.5.

В: Вы сказали, что я могу установить вместе KDE 3.5 и KDE 4.0, но они блокируют друг друга !?

О: Вам необходимо установить последнюю ревизию (не версию!) пакета kde-base/kdebase (если Вы используете монолитные ebuilds) или kde-base/kdebase-startkde (если Вы используете отдельные ebuilds). Эти ревизии включают патчи, позволяющие множествам версиям KDE сосуществовать.

Примечание: Эти ревизии пока не были стабилизированы, таким образом Вам необходимо прописать ключевые слова для них.

В: Почему KDE 4.0.0 жестко замаскирован (**hard-masked**)?

О: KDE 4.0.0 является очень технологической версией. Он пока не предназначен для полной замены серии KDE 3. Множество возможностей отсутствуют и множество ошибок ещё будет найдено. Только когда KDE 4.x перейдет на стадию, которая предложит полноценное обновление для пользователей, тогда с него будет снят статус: жестко замаскирован!

В: Почему KDE 4.0.0 замаскирован EAPI?

О: Ebuilds KDE 4.0.0 используют slot dependencies, такие как kde-base/kdebase-startkde:kde-4, с синтаксисом зависимости. Вам необходима версия портежей, которая понимает это. Если вы получаете сообщение о маске, тогда Вам следует просто обновить портежи до новой версии.

Листинг 3.1: Обновление до версии portage, которая поддерживает EAPI=1

```
# emerge --oneshot >=sys-apps/portage-2.1.3.12
```

В: Где монолитные KDE ebuilds?

О: Мы обнаружили, что иметь оба типа ebuild: отдельные и монолитные часто вводит пользователей в замешательство, а также ненужная никому сложность в eclasses и бесцельное дублирование усилий для команды. Так как теперь KDE использует stake, то основной недостаток отдельных ebuilds (долгое время компиляции, потому что дублируется ./configure-runs) незначителен. По причинам указанным выше, мы решили сопровождать только отдельные ebuilds. Для переключения от монолитных к отдельным ebuilds, удалите все установленные монолитные ebuilds:

Листинг 3.2: Удаление монолитных ebuilds KDE4

```
emerge -C ~kde-  
base/kde{,accessibility,admin,artwork,base,edu,games,graphics,multimedia,netw  
ork,sdk,toys,utils,webdev}-4.0.
```

Потом, установите отдельные ebuilds, как объяснено ранее в этом руководстве.

В: Где KDE-PIM в KDE 4?

О: KDE-PIM пока не был выпущен с KDE 4.0.0, поэтому, к сожалению, пока вы не найдете его в дереве портежей. Вы, конечно, можете использовать, версии 3.5 kmail, akregator и их друзей совместно с KDE 4.0.0.

В: Я попробовал KDE4 и осознал, что пока не хочу его использовать. Что мне

следует делать в этом случае?

О: Удалите KDE 4 из Ваших файлов: `package.keywords` и `package.unmask`, затем удалите пакеты, которые вы установили. Не забудьте обновить Вашу переменную `DISPLAYMANAGER` если Вы ранее изменяли её на `kde-4.0`.

В: Означает ли это, что KDE 3.5 будет убран?

О: Вовсе НЕТ! KDE 3.5 — это очень стабильная версия, и она имеет очень мало ошибок. Несмотря на то, что мы не увидим новых возможностей в KDE 3.5, мы планируем ещё долгое время сопровождать его.

Руководство по настройке GNOME

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/gnome-config.xml>

С версии: 1.2

1. Что такое GNOME?

О проекте

[GNOME](#) - это свободное ПО. Проект посвящен разработке GNOME - рабочего стола и пакет разработчика для Unix/Linux систем. [GNOME Foundation](#) координирует разработку и другие аспекты проекта GNOME.

О Программном обеспечении

GNOME представляет собой рабочую среду а также платформу для разработчиков. Это "открытое ПО" используется в качестве рабочей среды для рабочих станций, некоторыми промышленными лидерами. Эта среда представляет интерес, как для бизнес пользователей, домашних пользователей так и разработчиков.

Сообщество

Как и любого большого открытого проекта, у Gnome очень большое количество разработчиков, а также большая пользовательская база. [Footnotes](#) содержит новости для пользователей GNOME Desktop; [GnomePlanet](#) предназначена для хакеров/помощников, а [Developer.Gnome.Org](#) для разработчиков.

2. Установка GNOME

Что вам понадобится?

Перед тем как устанавливать **GNOME**, вы может быть захотите отредактировать переменные USE. Убедитесь что **gtk** и **gnome** прописаны как переменные USE в /etc/make.conf. Если вы не планируете использовать **KDE** (другая подобная рабочая среда) тогда, удалите **qt** и **kde**.

Листинг 2.1: Пример использования переменных USE в /etc/make.conf
USE="-qt -kde gtk gnome"

После этого, начните установку GNOME с помощью **emerging gnome** и **xscreensaver**:

Листинг 2.2: Установка GNOME
emerge gnome xscreensaver

Это займет какое-то время, за это время вы можете прочитать все те книги которые для вас приготовила мам, и которые вы никогда в жизни не открывали. Готово? Отлично, теперь вы должны обновить переменные рабочей среды.

Листинг 2.3: Обновление зависимостей рабочей среды
env-update && source /etc/profile

Если вы были внимательны, то после завершения выполнения команды **emerge**,

вам предлагают добавить **famd** по умолчанию, чтобы **nautilus** и **gnome-vfs** могли отслеживать изменения файлов:

Листинг 2.4: Добавление famd по умолчанию

```
# /etc/init.d/famd start  
# rc-update add famd default
```

Первые впечатления

Давай для начала посмотрим что у нас получилось. Выйдите из root shell и зайдите в систему как обычный пользователь. Сейчас мы сконфигурируем сессию GNOME с помощью команды **startx**:

Листинг 2.5: Использование рабочей среды GNOME по умолчанию

```
$ echo "exec gnome-session" > ~/.xinitrc
```

Запускаем графическую среду, набрав команду **startx**:

Листинг 2.6: Запуск GNOME

```
$ startx
```

Если все сделано правильно, GNOME приветствует вас. Поздравляем. Теперь давайте попробуем сконфигурировать рабочую среду GNOME.

3. Настройка GNOME

Графический менеджеры GNOME входа в систему

Если вы хотите, чтобы GNOME Display Manager (GDM) запускался автоматически при загрузке (таким образом вы сможете зайти в систему с помощью графического интерфейса), то вы должны установить xdm в default runlevel.

Листинг 3.1: Установка запуска xdm по умолчанию

```
# rc-update add xdm default
```

Проверьте содержание /etc/X11/Sessions:

Листинг 3.2: Проверка содержания /etc/X11/Sessions /etc/X11/Sessions

```
# ls /etc/X11/Sessions  
Xsession Gnome
```

Как видите, теперь доступна сессия gnome. Теперь отредактируйте **/etc/rc.conf** и измените 2 переменные: **DISPLAYMANAGER** (должно быть gdm) и **XSESSION** (должно быть Gnome):

Листинг 3.3: Редактирование /etc/rc.conf

```
DISPLAYMANAGER="gdm"  
XSESSION="Gnome"
```

После перезагрузки, GNOME Display Manager спросит имя пользователя и пароль. GNOME будет рабочей средой по умолчанию (конечно же здесь есть возможность выбора).

Использование UTF-8 в Gentoo

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/utf-8.xml>

С версии: 1.0

1. Предупреждение или Напутствие переводчика

Путь Юникода в русском Gentoo-сообществе сложен. Мы постоянно сталкивались с проблемами неполной поддержки или же неподдержки вовсе отдельными приложениями стандарта UTF-8.

Но всё меняется к лучшему. Большинство приложений так или иначе стало поддерживать региональные настройки UTF-8. А для не поддерживающих уже разработаны способы исправления этого недостатка. Но всё же, если у вас возникли трудности с настройкой Юникода в Gentoo, то вы всегда найдете помощь и поддержку (иногда сквозь стиснутые зубы :)) на следующих страницах:

- [Статья в gentoo-wiki о поддержке Юникода](#)
- [Форум русского сообщества Gentoo](#)
- [Русскоязычный форум на официальном сайте Gentoo](#)

Если вы заметили неточность или хотите дополнить данное руководство, пожалуйста свяжитесь переводчиком или одним из редакторов перевода.

2. Кодировки

Что такое кодировка?

Компьютеры неспособны воспринимать текст. Вместо этого они каждый символ представляют как число. Традиционно каждый набор чисел, соответствующий алфавиту и символам (известный как кодировка или набор символов), ограничен в размере в силу ограничений оборудования.

История кодировок символов

Самой распространенной (или наиболее принятой повсеместно) кодировкой является ASCII (Американский стандартный код для обмена информацией, American Standard Code for Information Interchange). Повсеместное использование говорит о том, что ASCII — наиболее успешный программный стандарт современности. Нынешний ASCII стандартизован в 1986 году (ANSI X3.4, RFC 20, ISO/IEC 646:1991, ECMA-6) Американским национальным институтом по стандартизации (American National Standards Institute, ANSI).

ASCII является строго семибитной кодировкой, из чего следует, что она использует семь двоичных цифр, то есть интервал от 0 до 127. ASCII начинается с 32 невидимых управляющих символов (с кодами от 0 до 31) и заканчивается

управляющим символом DEL (или delete) с кодом 127. Символы в диапазоне от 32 до 126 видимы — это пробел, знаки препинания, латинские буквы и цифры.

Восьмой бит в ASCII изначально использовался как бит целостности для проверки ошибок при передаче информации. Если не указывалось особо, он всегда равнялся 0. Это означает, что в ASCII каждый символ равняется одному байту.

Хотя ASCII было достаточно для передачи информации на английском, для других европейских языков, содержащих символы с ударениями, это было не так просто. Для них были разработаны стандарты семейства ISO 8859. Они были обратно совместимы с ASCII, но также использовали восьмой бит для дополнения таблицы дополнительными 127 символами для каждой кодировки. Ограничения ISO 8859 были незначительными, поэтому стандарт получил распространение. На данный момент существует 15 вариантов стандарта ISO 8859 (от 8859-1 до 8859-15). Однако для каждого символа вне ASCII-совместимого диапазона между этими стандартами возникал конфликт. Чтобы еще больше усилить конфликт между различными кодировками, для восточноевропейских языков корпорацией Microsoft был введен набор Windows-1252, использованный в ранних версиях Windows. Этот набор является надстройкой ISO 8859-1, обладающий собственными изменениями. Однако все же эти наборы были совместимы с ASCII.

Необходимость разработки совершенно отличных от ASCII однобайтовых кодировок для нелатинских алфавитов, как например EUC (Extended Unix Coding), используемый в японском и корейском (и в меньшей степени китайском) алфавитах, породила еще большую неразбериху с кодировками, из-за чего ряд операционных систем все еще использует различные наборы символов для одного и того же языка, например японские Shift-JIS и ISO-2022-JP. А пользователи, желающие видеть кириллицу, вынуждены были выбирать между KOI8-R (для русского и болгарского языков) и KOI8-U (для украинского языка), неудачной ISO 8859-5 и популярной Windows-1251. Все три семейства кодировок не полностью совместимы с ASCII (хотя в KOI8 фонетически созвучные кириллические символы расположены так же, как и латинские, благодаря чему даже при отбрасывании восьмого бита текст оставался читабельным в ASCII-терминалах в виде транслита).

Все это привело к путанице и полной невозможности многоязычного общения, в особенности с использованием различных алфавитов. Переходим к Юникоду.

Что такое Юникод?

Юникод отбрасывает лимит традиционных однобайтовых кодировок. Он использует 17 «плоскостей», содержащих по 65 536 кодов символов. Таким образом, максимальное возможное число символов равно 1 114 112. Поскольку в первой плоскости («Basic Multilingual Plane» или BMP) содержится почти всё, что вам может понадобиться, многие ошибочно посчитали, что Юникод — это 16-битный набор символов.

Юникод реализован несколькими способами, но распространены только два — UTF (Unicode Transformation Format) и UCS (Universal Character Set). Число после UTF обозначает число бит на каждый символ, когда как число после UCS

обозначает число байт. UTF-8 стала наиболее распространенной при обмене текста Юникод из-за малых (сравнительно) потерь при хранении текста, Поэтому она и является основной темой данного документа.

UTF-8

UTF-8 — это кодировка переменной длины. Это означает, что для каждого символа она может использовать от одного до четырех байт. Первый байт UTF-8 используется для кодирования ASCII, благодаря чему UTF-8 полностью обратно совместима с ASCII. UTF-8 подразумевает, что ASCII и латинские символы равнозначны с небольшим увеличением размера данных, так как используется только первый байт. Пользователи восточных кодировок (например, японского), которым назначены старшие диапазоны, не так рады, так как в результате их данные разбухают на 50 %.

Что UTF-8 может сделать для вас

UTF-8 позволяет вам работать в совместимой со всеобщими стандартами и принятой по всему миру многоязычной среде, правда с небольшими потерями при обработке данных. UTF-8 — это наилучший способ для передачи не-ASCII символов через интернет, будь то электронная почта, IRC-сети или что-нибудь еще. Несмотря на это, множество людей расценивают использование UTF-8 при передаче данных неприличным и неуважительным. Всегда следует выяснять, поддерживает ли определенный канал, группа Usenet или список рассылки UTF-8 перед тем, как использовать символы из не-ASCII диапазона UTF-8.

3. Включение UTF-8 в Gentoo Linux

Поиск или создание локалей UTF-8

Теперь, когда вы поняли основы Юникода, вы готовы начать использование UTF-8 в вашей системе.

Главным требованием для UTF-8 является наличие установленной библиотеки glibc с поддержкой национального языка. При этом рекомендуется использовать файл `/etc/locale.gen`. Описание использования этого файла, однако, выходит за рамки данного документа. Его использование описано в Руководстве по локализации Gentoo.

Далее, нужно определить, доступна ли нам локаль UTF-8 для нашего языка или придется её создать.

Листинг 3.1: Поиск существующей локали UTF-8
(Замените "ru_RU" своим регионом)

```
# locale -a | grep 'ru_RU'
ru_RU
ru_RU.UTF-8
```

На выходе этой команды мы должны получить хотя бы одну строку, содержащую суффикс .UTF-8. Если таковых нет, то нам придётся создать локаль, совместимую с UTF-8.

Примечание: Запускайте следующую команду, если у вас нет UTF-8 локали для вашего языка.

Листинг 3.2: Создание локали UTF-8
(Замените "ru_RU" своим регионом)
localedef -i ru_RU -f UTF-8 ru_RU.UTF-8

Другим способом включить локаль UTF-8 является добавление её в файл /etc/locale.gen и генерация нужных локалей командой locale-gen.

Листинг 3.3: Одна из строк в /etc/locale.gen
ru_RU.UTF-8/UTF-8

Настройка локали

Есть одна переменная среды, которую необходимо установить, чтобы использовать UTF-8 локали: LANG (эту переменную также можно перезаписывать переменной LC_ALL). Есть множество способов сделать это. Некоторые люди предпочитают использовать UTF-8 только для определенного пользователя, поэтому они устанавливают эту переменную в своём ~/.profile (если используется /bin/sh), ~/.bash_profile или ~/.bashrc (если используется /bin/bash).

Другие предпочитают установить локаль глобально. Есть по крайней мере один весомый аргумент в пользу этого подхода — при использовании /etc/init.d/xdm, так как сценарии инициализации запускают диспетчер окон до того, как будут загружены конфигурационные файлы командной оболочки.

При установке глобальной локали должен быть использован файл /etc/env.d/02locale. Он должен выглядеть следующим образом:

Листинг 3.4: Вид /etc/env.d/02locale
(Как и всегда, замените "ru_RU.UTF-8" вашей локалью)
LANG="ru_RU.UTF-8"

Примечание: Вы можете также указывать LC_ALL вместо LANG. Так локаль устанавливается для всех категорий, включая числовые и денежные значения. На небольшом количестве систем это может вызывать некоторые проблемы. Однако, большинство пользователей может использовать LC_ALL без проблем. За дополнительными сведениями о категориях, затронутых использованием LC_ALL, пожалуйста, обращайтесь к странице локалей GNU.

Далее следует обновить среду.

Листинг 3.5: Обновление среды


```
# env-update
>>> Regenerating /etc/ld.so.cache...
* Caching service dependencies ...
# source /etc/profile
```

Теперь запустите `locale` без аргументов, чтобы увидеть, что у вас верные переменные среды:

Листинг 3.6: Проверка новой применённой локали

```
# locale
LANG=
LC_CTYPE="ru_RU.UTF-8"
LC_NUMERIC="ru_RU.UTF-8"
LC_TIME="ru_RU.UTF-8"
LC_COLLATE="ru_RU.UTF-8"
LC_MONETARY="ru_RU.UTF-8"
LC_MESSAGES="ru_RU.UTF-8"
LC_PAPER="ru_RU.UTF-8"
LC_NAME="ru_RU.UTF-8"
LC_ADDRESS="ru_RU.UTF-8"
LC_TELEPHONE="ru_RU.UTF-8"
LC_MEASUREMENT="ru_RU.UTF-8"
LC_IDENTIFICATION="ru_RU.UTF-8"
LC_ALL=ru_RU.UTF-8
```

Вот и всё. Теперь вы используете локаль UTF-8, и следующим этапом является настройка повседневно используемых вами приложений.

4. Поддержка приложениями

Пока Юникод делает первые шаги в мире программного обеспечения, и многобайтовые кодировки еще не включены в языки программирования, наподобие C, и множество повседневно используемых программ. Даже сейчас некоторые программы не способны работать с UTF-8 как надо. К счастью, таких не так много!

Имена файлов, NTFS и FAT

В конфигурационном меню ядра Linux есть множество параметров NLS, но главное — не паниковать! В большинстве случаев всё, что вам нужно — включить поддержку UTF-8 NLS в ваше ядро и изменить NLS по умолчанию на `utf8`.

Листинг 4.1: Конфигурация ядра для UTF-8 NLS

```
File Systems -->
Native Language Support -->
(utf8) Default NLS Option
<*> NLS UTF8
```

(также <*> для других кодировок, используемых в ФС FAT или Joilet CD-ROM)

Если вы планируете монтировать разделы NTFS, то вам понадобится указать параметр при монтировании `nls=`. Если вы будете монтировать разделы FAT, вам понадобится указать при монтировании параметр `codepage=`. Также вы можете установить для FAT кодовую страницу по умолчанию при конфигурации ядра. Помните, что параметр `codepage`, указываемый при монтировании, будет иметь приоритет над настройками ядра.

Листинг 4.2: Настройки FAT при конфигурировании ядра

```
File Systems -->  
DOS/FAT/NT Filesystems -->  
(866) Default codepage for fat
```

Не устанавливайте `Default iocharset for fat` в UTF-8, так как это не рекомендуется. Вместо этого укажите параметр `utf8=true` при монтировании раздела FAT. Для более детальной информации, смотрите `man mount` и документацию по ядру `/usr/src/linux/Documentation/filesystems/vfat.txt`.

Чтобы сменить кодировку имен файлов, используйте `app-text/convmv`.

Листинг 4.3: Пример использования `convmv`

```
# emerge --ask app-text/convmv  
(Формат команды)  
# convmv -f <current-encoding> -t utf-8 <filename>  
(Замените koi8-г кодировкой, с которой хотите конвертировать)  
# convmv -f koi8-r -t utf-8 filename
```

Для изменения содержимого файлов, используйте утилиту `iconv`, поставляемую вместе с `glibc`:

Листинг 4.4: Пример использования `iconv`

```
(Замените koi8-г кодировкой, с которой хотите конвертировать)  
(Проверка верности вывода)  
# iconv -f koi8-r -t utf-8 filename  
(Конвертация файла, вы должны указать другое имя файла)  
# iconv -f koi8-r -t utf-8 filename > newfile
```

Также для этой цели может быть использована утилита `app-text/recode`.

Системная консоль

Важно: Вам необходим `>=sys-apps/baselayout-1.11.9` для поддержки Юникода в консоли.

Для включения UTF-8 в консоли вы должны отредактировать `/etc/rc.conf` и установить переменную `UNICODE="yes"`. Также прочтите комментарии в этом файле — важно, чтобы в системе были шрифты с нужным диапазоном символов (если вы хотите выжать из Юникода всё).

В файле `/etc/conf.d/keymaps` переменной `KEYMAP` должна соответствовать раскладка Unicode.

Листинг 4.5: Пример файла `/etc/conf.d/keymaps`
 (Замените "ru4" на вашу любимую раскладку)
 (Для русского языка нужен ключ -и — прим. переводчика)
`KEYMAP="-и ru4"`
 (Подмена кодировки для русского языка — прим. переводчика)
`DUMPKEYS_CHARSET="koi8-r"`

Важно: Помните, что для правильного отображения вам понадобятся шрифты с поддержкой Юникода. Подробнее смотрите раздел X11 и шрифты этого руководства.

А теперь выберем подходящий шрифт, поддерживающий Юникод.

Листинг 4.6: Файл `/etc/conf.d/consolefont`
 (Отредактируйте переменную `CONSOLEFONT`)
`CONSOLEFONT="ter-k14n" # шрифт из пакета terminus-font)`

ncurses и slang

Примечание: Если вы не устанавливали или не используете `slang`, не принимайте во внимание упоминания о нём в этом разделе.

Будет хорошим решением добавить `unicode` к глобальным USE-флагам в файле `/etc/make.conf`, а затем при необходимости пересобрать `sys-libs/ncurses` и `sys-libs/slang`. Portage это сделает автоматически при обновлении системы:

Листинг 4.7: Обновление системы
`# emerge --update --deep --newuse world`

Также понадобится пересобрать пакеты, зависящие от них, раз изменения USE вступили в силу. Используемая утилита (`revdep-rebuild`) входит в пакет `gentoolkit`.

Листинг 4.8: Пересборка программ, скомпонованных с `ncurses` или `slang`
`# revdep-rebuild --soname libncurses.so.5`
`# revdep-rebuild --soname libslang.so.1`

KDE, GNOME и Xfce

Все основные графические оболочки полностью совместимы с Юникодом и не требуют дополнительной настройки, кроме той, что описана в этом документе. Все это благодаря тому, графические библиотеки (Qt и GTK+2) совместимы с UTF-8. Следовательно, все приложения, работающие на основе этих библиотек, также должны поддерживать UTF-8 без дополнительных настроек.

Исключением в этом правиле является Xlib и GTK+1. GTK+1 требует iso-10646-1 FontSpec в ~/.gtkrc, например -misc-fixed-**-**-**-iso10646-1. Также приложения, использующие Xlib или Xaw, нуждаются в подобной FontSpec, иначе они не будут работать.

Примечание: Если у вас установлен Центр управления gnome1, используйте его. Воспользуйтесь любым из шрифтов семейства iso10646-1.

Листинг 4.9: Пример ~/.gtkrc (для GTK+1), определяющий Юникод-совместимый шрифт

```
style "user-font"
{
    fontset="-misc-fixed-**-**-**-iso10646-1"
}
widget_class "*" style "user-font"
```

Важно: В Xorg 6.8.2 для славянских языков эта проблема подобным образом не решается. Обратитесь к руководствам на сайтах Fantoo или Gentoo-wiki для решения данной проблемы либо откатите Xorg до версии 6.8.0.

Если приложение одинаково поддерживает Qt и GTK+2, то обычно GTK+2 GUI выдает лучшие результаты при работе с Юникодом.

Х11 и шрифты

Важно: По сравнению с XFree86 x11-base/xorg-x11 лучше поддерживает Юникод, и поэтому настоятельно рекомендуется использовать именно его.

Шрифты TrueType обычно совместимы с Юникодом, и большинство шрифтов, поставляемых с Xorg, имеют великолепную поддержку кодировок, хотя не все глифы Юникода могут быть отображены для конкретного шрифта. Чтобы собрать шрифты с поддержкой Восточноазиатских алфавитов (включающие набор Bitstream Vera) для X-сервера, добавьте USE-флаг cjk. Много приложений используют этот флаг, поэтому неплохо бы внести его как постоянный флаг.

Также несколько пакетов шрифтов в Portage совместимы с Юникодом.

Листинг 4.10: Необязательно: установка некоторых Юникод-совместимых шрифтов

```
# emerge terminus-font intlfonts freefonts cronyx-fonts corefonts
```

Диспетчеры окон и терминальные эмуляторы

Диспетчеры окон, не использующие GTK или Qt, обычно очень хорошо поддерживают Юникод, так как чаще всего для отображения шрифтов используют библиотеку Xft. Если ваш диспетчер окон не использует Xft, то вы все еще можете использовать FontSpec, указанный в предыдущем разделе в качестве шрифта Юникода.

Эмуляторы терминала, использующие Xft, в большинстве случаев поддерживают Юникод. Кроме Konsole и gnome-terminal, лучшим выбором в Portage будет x11-terms/rxvt-unicode, xfce-extra/terminal, gnustep-apps/terminal, x11-terms/mlterm или

чистый x11-terms/xterm, собранный с USE-флагом unicode и запускаемый как uxterm. app-misc/screen тоже поддерживает UTF-8, если запускается с параметром screen -U, или внесите следующее в файл ~/.screenrc:

Листинг 4.11: ~/.screenrc с UTF-8

```
defutf8 on
```

Vim, Emacs, Xemacs и Nano

Vim полностью поддерживает UTF-8 и к тому же автоматически определяет файлы с UTF-8. Для более детальной информации используйте в Vim :help mbyte.txt.

Emacs 22.x и выше полностью поддерживает UTF-8. Xemacs 22.x ещё не поддерживает различные кодировки.

Более старшим версиям Emacs и/или Xemacs понадобятся пакеты app-emacs/mule-ucs и/или app-xemacs/mule-ucs и добавление следующих строчек в файл ~/.emacs для поддержкиCJK-языков в UTF-8:

Листинг 4.12: Emacs с поддержкой CJK UTF-8

```
(require 'un-define)
(require 'jsx0213)
(set-language-environment "Japanese")
(set-default-coding-systems 'utf-8)
(set-terminal-coding-system 'utf-8)
```

Nano полностью поддерживает UTF-8 начиная с версии 1.3.6.

Командные оболочки

На данный момент только bash полностью поддерживает Юникод через библиотеку GNU readline. Пользователи оболочки Z находятся в промежуточном положении — пока ни одна из составных оболочки еще поддерживает Юникод, хотя на данный момент ведутся усиленные работы для поддержки многобайтовых кодировок.

Оболочки C, tcsh и ksh не поддерживают UTF-8.

Irssi

Irssi полностью поддерживает UTF-8, хотя для этого требуется дополнительная настройка пользователем.

Листинг 4.13: Включение UTF-8 в Irssi

```
/set term_charset UTF-8
```

Для каналов, где не-ASCII символы чаще всего передаются в не-UTF-8

кодировках, может пригодиться команда `/recode` для перекодировки символов. Наберите `/help recode` для большей информации.

Mutt

Почтовый агент Mutt очень хорошо справляется с Юникодом. Чтобы использовать UTF-8 для Mutt, вам нет необходимости что-либо добавлять в конфигурационные файлы. Mutt будет работать с Юникодом без модификации, при условии, что все ваши конфигурационные файлы (включая подпись) сохранены в UTF-8.

Примечание: Вы всё ещё можете видеть знаки вопроса при чтении почты с помощью Mutt. Это случается из-за того, что некоторые люди используют почтовые клиенты, не устанавливающие используемую кодировку письма. Вы ничего больше не можете сделать, кроме как попросить их правильно настроить свои клиенты.

См. [Mutt Wiki](#) для большей информации.

Man

Man-страницы являются неотъемлемой частью любой системы Linux. Чтобы удостовериться, что все страницы отображаются верно, отредактируйте `/etc/man.conf` и замените следующую строку.

Листинг 4.14: Изменения `man.conf` для поддержки Юникода

```
(Это старая строка)
NROFF      /usr/bin/nroff -Tascii -c -mandoc
(Замените ее этой)
NROFF      /usr/bin/nroff -mandoc -c
```

elinks и links

Это самые популярные текстовые браузеры, и вы узнаете, как установить поддержку UTF-8 для них. В `elinks` и `links` есть два способа сделать это — через меню Setup (Установки) браузера или отредактировав конфигурационный файл. Чтобы настроить параметры в самом браузере, откройте какой-нибудь сайт с помощью `elinks` или `links` и нажмите `Alt+S`. Вы попадете в меню настроек (Setup Menu), после чего выберите параметры терминала (Terminal options), или нажмите `T`. Пролистайте вниз и выберите последний параметр — UTF-8 I/O, нажав на `Enter`. После сохраните и выйдите из меню. В `links` вы можете сделать то же самое, нажав `Alt+S`, а затем `S` для сохранения. Пример конфигурационного файла показан ниже.

Листинг 4.15: Включение UTF-8 в `elinks/links`

(Для `elinks` добавьте в `/etc/elinks/elinks.conf` или `~/.elinks/elinks.conf` следующую строку)

```
set terminal.linux.utf_8_io = 1
```

(Для `links` добавьте `~/.links/links.cfg` следующую строку)

```
terminal "xterm" 0 1 0 us-ascii utf-8
```

Samba

Samba — это набор программ, реализующих протокол SMB (Server Message Block) для UNIX-систем (Mac, Linux и FreeBSD). Этот протокол также иногда упоминается как Common Internet File System (CIFS). Samba содержит также систему NetBIOS, используемую для предоставления доступа к файлам в Windows-сетях.

Листинг 4.16: Включение UTF-8 в Samba

(Отредактируйте /etc/samba/smb.conf и добавьте следующее в раздел [global])

```
dos charset = 866
```

```
unix charset = UTF-8
```

```
display charset = UTF-8
```

Проверка работоспособности

Есть множество сайтов, проверяющих UTF-8. net-www/w3m, net-www/links, net-www/elinks, net-www/lynx, все браузеры, основанные на движке Mozilla (включая Firefox) поддерживают UTF-8. Konqueror и Opera тоже полностью поддерживают UTF-8.

При использовании текстовых браузеров, удостоверьтесь, что используете совместимый с Юникодом терминал.

Если вы видите символы, отображаемые как квадратики с буквами или цифрами внутри, то это значит, что в ваших шрифтах нет соответствующего глифа для символа. Вместо этого, они отображают квадрат с шестнадцатиричным кодом символа UTF-8.

- [Тестовая страница W3C с UTF-8](#)
- [Тестовая страница Франкфуртского университета](#)

Ввод символов

Dead keys могут быть использованы для ввода символов, не отображенных на клавиатуре, в X-сервере. Для этого нужно, удерживая правый Alt (в некоторых странах он называется AltGr), нажать на дополнительную неалфавитную клавишу с символом, не являющимся буквой или цифрой (модификатор), а затем нажать на необходимую букву. Dead key изменит ее. Ввод может быть изменен также с помощью Shift при нажатых AltGr и модификаторе.

Чтобы включить dead keys в X, нужно иметь раскладку, поддерживающую их. Большинство европейских раскладок уже имеют dead keys по умолчанию. Тем не менее, они не являются настоящими североамериканскими раскладками. Хотя существуют незначительные отличия между различными раскладками, простейшим решением будет использование раскладки «en_US» вместо просто «us». Раскладка указывается в /etc/X11/xorg.conf:

Листинг 4.17: Пример раздела /etc/X11/xorg.conf

```
Section "InputDevice"
    Identifier "Keyboard0"
    Driver      "kbd"
    Option      "XkbLayout" "en_US" # Вместо просто "us"
    (Другие параметры Xkb)
EndSection
```

Примечание: Эти изменения нужны только в том случае, если вы используете североамериканскую раскладку либо любую другую, в которой dead keys не работают. Европейским пользователям нет нужды что-либо менять.

Изменения вступят в силу только при перезагрузке X-сервера. Чтобы изменения вступили немедленно, используйте утилиту `setxkbmap`, например, `setxkbmap en_US`.

Рассмотрим действия dead keys на примерах. Хотя результат зависит от текущей локали, сам принцип действия должен работать в нее зависимости от нее. Примеры содержат символы Юникода, так что убедитесь, что ваш браузер корректно отображает их.

При нажатии единожды нажатых AltGr и [, а затем нажатой «а» мы получаем «а». При единожды нажатых AltGr и [, а затем нажатой «е», получаем «е». Нажав на AltGr и ;, мы получим «а», а нажав AltGr и ;, а затем на «е», получаем «е».

Нажав на AltGr, Shift и [, отпустив их и нажав «а», мы получим скандинавскую «а». Также при нажатии AltGr, Shift и [, отпуская только [и повторном ее нажатии мы получим только «?». Хотя этот символ (U+02DA) и похож на символ градуса (U+00B0), на самом деле он им не является. Подобным образом работают и другие модификаторы — нажатие AltGr and [, отпускание [и повторное нажатие выводят «?».

AltGr может быть использована вместе с одной из алфавитных клавиш. Например, комбинация AltGr и m выводит греческую строчную букву мю: «ч». AltGr и s выводит немецкий символ эсцет: «?». Множество европейцев захотят установить сочетание AltGr и 4 (или E в зависимости от клавиатурной таблицы), выводящее символ Евро, «?» (так как на их клавиатурах он нанесен).

Руководство по русской локализации Gentoo Linux

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/guide-localization.xml>

С версии: 1.0

1. Введение

Что такое русская локализация

В понятие локализации входит обеспечение вывода и ввода символов национальных языков (в случае использования набора символов, отличных от чистого латинского) и определение переменных окружения, зависящих от страны, языка и набора символов (то есть собственно установка locale). В число таких переменных, кроме страны, языка и набора символов, входят также представление даты и времени, разделителей десятичных дробей, денежных единиц и так далее.

Вследствие различия подходов в этом документе будут отдельно рассмотрены локализация текстового (то есть Linux-консоли) и графического (оконной системы X) режимов.

О кириллических кодировках

В условиях России и некоторых других стран бывшего СССР локализация осложняется множественностью наборов символов (charsets), используемых для представления алфавитов, основанных на кириллице. Причём для Unix-систем получили распространение минимум три таких набора, в просторечии именуемых кодировками: традиционная для свободных Unix-клонов (в том числе и Linux) кодировка KOI8-R, кодировка ISO-8859-5 (именуемая кодировкой ГОСТ), используемая в проприетарных Unix-системах, и кодировка CP1251 (или кодировка Windows). Кроме того, для обеспечения экранного вывода используется кодировка CP866 (или кодировка DOS). Наконец, в последнее время всё большую популярность получает кодировка UTF-8.

В Gentoo Linux штатно (как говорится, из коробки) поддерживается кодировка KOI8-R, и всё сказанное ниже относится именно к ней. Локализацию для кодировок ISO-8859-5 или CP1251 пользователь может выполнить самостоятельно, получив из независимых источников некоторые дополнительные компоненты. О локализации для кодировки UTF-8, призванной унифицировать подход к интернационализации, будет говориться в следующих версиях этого документа, когда использование её станет общепринятым.

2. Базовая кириллизация консоли

Введение

Итак, базовая локализация консоли включает обеспечение вывода национальных символов на экран и ввода их с клавиатуры. Однако в случае кириллической локализации вследствие особенностей наборов символов и аппаратных особенностей PC для ввода и для вывода используются, как правило, разные кодировки. И это требует еще двух дополнительных шагов — установки карты соответствия между наборами символов ввода и вывода, и активизация её на каждой из виртуальных консолей.

Установка экранного шрифта

Экранный шрифт, загружаемый по умолчанию при старте системы, определяется в файле `/etc/rc.conf` строкой

Листинг 2.1: Строка, определяющая экранный шрифт

```
CONSOLEFONT="default8x16"
```

Для вывода русских букв "умолчальное" значение следует заменить на имя какого-либо файла шрифта с поддержкой кириллицы, например, `sr866-8x16` для представления кодировки DOS или `koi8r-8x16` — для кодировки KOI8-R. Например,

Листинг 2.2: Определение экранного шрифта для кодировки `sr866`

```
CONSOLEFONT="sr866-8x16"
```

Или —

Листинг 2.3: Определение экранного шрифта для кодировки KOI8-R

```
CONSOLEFONT="koi8r-8x16"
```

Примечание: Файлы шрифтов (и не только кириллических) находятся в каталоге `/usr/share/consolefonts/` и имеют вид `*.gz`, `*.psf.gz` или `*.psfu.gz`. При стандартном их размещении и форме ни полного пути, ни "расширения" имени указывать не обязательно.

Важно: В кириллическом Linux'е для экранного вывода, как правило, используется кодировка `sr866` (вне зависимости от того, какой набор символов принят для клавиатурного ввода). О причинах этого говорить можно было бы долго — достаточно указать, что в любом ином случае программы, использующие псевдографику (например, `Midnight Commander`) приобретут весьма уродливый вид.

Экранные кириллические шрифты, поставляемые с Gentoo Linux, далеки от эстетического совершенства (справедливости ради заметим, что это относится почти ко всем Unix-клонам). И потому в качестве источника дополнительных шрифтов можно рекомендовать пакет `console-tools-cyrillic` Виктора Вагнера. В результате приведенная выше в качестве примера строка может принять вид

вроде

Листинг 2.4: Один из лучших кириллических шрифтов для консоли
`CONSOLEFONT="UniCyr-sans"`

Установка раскладки клавиатуры

Загружаемая по умолчанию раскладка клавиатуры также определяется в файле `/etc/rc.conf` строкой

Листинг 2.5: Строка, определяющая раскладку клавиатуры
`KEYMAP="us"`

По аналогии с предыдущим случаем легко догадаться, что значение `us` следует заменить на имя файла какой-либо кириллической (на этот раз уже в кодировке KOI8-R) раскладки. Место этих файлов (имеющих вид `*.map.gz`) — в каталоге `/usr/share/keymaps/i386/qwerty`, где для русской кодировки KOI8-R имеется несколько раскладок, например, `ru1-ru4`. Различия между ними — в соответствии различной маркировке клавиш (DOS или Windows — не путать с одноименными кодировками) и положении переключателя латиница/кириллица. Рискну взять на себя смелость рекомендовать раскладку `ru4` — для Windows-маркированных клавиатур, в которой переключателем служит CapsLock (при этом фиксация верхнего регистра обеспечивается одновременным нажатием клавиш Shift+CapsLock). То есть строка примет вид:

Листинг 2.6: Один из примеров определения кириллической раскладки
`KEYMAP="ru4"`

Примечание: Как и в случае с экранными шрифтами, при стандартном расположении файлов раскладок указывать полный путь к ним и "расширение" имени нет необходимости.

Предупреждение: При желании вы можете выбрать и раскладку клавиатуры для кодировки `sr1251` (например, `ru_win`). Однако в этом случае придётся отыскать или самостоятельно изготовить таблицу перекодировки (`mapscreen`) `sr1251->sr866`.

Примечание: Нет ничего более уродливого, чем программы, использующие псевдографику, при экранном представлении в кодировке `sr1251`.

Установка карты соответствия (`mapscreen`)

Теперь остаётся только обеспечить соответствие между кодировками ввода

(KOI8-R) и вывода (cp866), то есть загрузить соответствующую таблицу перекодировки (mapscreen). Для этого снова обращаемся к файлу /etc/rc.conf. Строка в нём, отвечающая за эту операцию, имеет по умолчанию такой вид:

Листинг 2.7: Строка, определяющая раскладку клавиатуры

```
#CONSOLETRANSLATION="cp437_to_iso01"
```

Необходимо, сняв с неё символ комментария (#), заменить "умолчальное" значение на необходимое нам, и подсмотреть его можно в каталоге /usr/share/consoletrans — это koi2alt:

Листинг 2.8: Установка таблицы соответствия cp866->KOI8-R

```
CONSOLETRANSLATION="koi2alt"
```

Предупреждение: Пожалуйста, не забудьте снять символ комментария со строки CONSOLETRANSLATION — кто только на моей памяти не наступал на эти грабли :-)

Примечание: Если вы по каким-либо причинам остановились на представлении экранных шрифтов в кодировке KOI8-R, загрузка и активизация карты соответствия, разумеется, не нужна.

Активизация карты соответствия

Результатом выполненной операции будет то, что после перезагрузки машины мы получили бы нормальный вывод кириллического текста (записанного в кодировке KOI8-R), но только — на первой (т.н. системной) консоли. Чтобы иметь то же самое на остальных виртуальных консолях (а в Gentoo Linux их по умолчанию еще 5), та же карта соответствия должна быть активизирована на каждой из них с помощью направления Escape-последовательности \033(K (символ \ в данном случае показывает, что следующие за ним символы рассматриваются как специальные). Достигнуть этого можно разными способами. Например, создать в каталоге /etc/init.d/ файл consoletrans и в любом текстовом редакторе внести в него следующие строки:

Листинг 2.9: Сценарий активизации mapscreen

```
for i in 1 2 3 4 5 6; do
echo -ne '\033(K' > /dev/vc/$i
done
```

Примечание: Это — для случая 6-ти консолей по умолчанию. Если количество их изменялось пользователем, значения in, очевидно, должны быть приведены в соответствие с реальностью.

Примечание: Следует отметить, что начиная с baselayout-1.8.x необходимость в этом отпала, поскольку этот код был встроен в /etc/init.d/consolefont.

Теперь командой `chmod a+x /etc/init.d/consoletrans` созданному скрипту следует присвоить бит исполнения, после чего он добавляется к уровню исполнения по умолчанию (default runlevel):

Листинг 2.10: Сценарий активизации mapscreen
`# rc-update add consoletrans default`

Примечание: Механизм автоматического обновления стартовых скриптов подробно описан в соответствующем разделе документации.

Вот теперь после перезагрузки машины мы получим полноценно локализованную консоль.

Примечание: Все действия, описанные в данной главе, могут быть выполнены как на заключительных шагах (final steps) установки Gentoo Linux, так и в произвольный момент времени в дальнейшем.

3. Установка русской locale

Установка locale ru_RU.KOI8-R

Установить локально-зависимые переменные можно различными способами, в том числе и просто в профильных файлах отдельных пользователей. Однако в Gentoo Linux предусмотрен автоматический механизм установки любых переменных окружения в качестве общесистемных — `env-update`, подробно описанный в соответствующих разделах документации.

Для использования этого механизма следует в файл /etc/env.d/02locale добавить строку

Листинг 3.1: Установка переменной LANG
`LANG="ru_RU.KOI8-R"`

которая определяет не только язык locale, но и все остальные локально-зависимые переменные. Далее выполняем команду `env-update`, дополняющую глобальный профильный файл /etc/profile, после чего остаётся только перечитать последний командой `source /etc/profile`. Теперь в ответ на команду `locale` последует вывод

Листинг 3.2: Вывод команды `locale` после установки переменной `LANG`

```
LANG=ru_RU.KOI8-R
LC_CTYPE="ru_RU.KOI8-R"
LC_NUMERIC="ru_RU.KOI8-R"
LC_TIME="ru_RU.KOI8-R"
LC_COLLATE="ru_RU.KOI8-R"
LC_MONETARY="ru_RU.KOI8-R"
LC_MESSAGES="ru_RU.KOI8-R"
LC_PAPER="ru_RU.KOI8-R"
LC_NAME="ru_RU.KOI8-R"
LC_ADDRESS="ru_RU.KOI8-R"
LC_TELEPHONE="ru_RU.KOI8-R"
LC_MEASUREMENT="ru_RU.KOI8-R"
LC_IDENTIFICATION="ru_RU.KOI8-R"
LC_ALL=
```

свидетельствующий, что все локально зависимые переменные приняли значения для страны России (`ru`), русского языка (`_RU`) и набора символов `KOI8-R`. Те же значения этих переменных будут устанавливаться и в сеансах всех пользователей, вошедших в систему после выполнения команды `env-update`.

Предупреждение: Механизм `env-update` однозначно работает в том случае, если пользовательской оболочкой (`login shell`) является `/bin/bash`. При использовании иных командных оболочек могут потребоваться некоторые дополнительные действия. Например, если в качестве `login shell` выступает оболочка `zsh`, следует отказаться от создания файла `~/.zshenv` в домашнем каталоге пользователя.

Важно: Переменная `LANG=ru_RU.KOI8-R` должна быть установлена в любом случае, даже если вы не любите сообщений на русском языке: в противном случае во многих программах (в том числе для `XFree86`) окажется просто невозможным, например, ввод символов кириллицы.

Как уже сказано, установка переменной `LANG` автоматически определяет и все прочие локально-зависимые переменные. Что в ряде случаев может быть нежелательным. Например, некоторые программы расчетного характера требуют, чтобы в качестве десятичного разделителя использовался символ точки, как в `locale POSIX`, а не запятой, как это принято в `locale ru_*`. Чтобы добиться этого, достаточно в конец файла `/etc/env.d/02locale`, после определения `LANG`, добавить строку вида:

Листинг 3.3: Установка переменной `LC_NUMERIC`

```
LC_NUMERIC="POSIX"
```

Аналогичным образом могут поступить пользователи, испытывающие стойкую неприязнь к русскоязычным меню и сообщениям программ:

Листинг 3.4: Установка переменной LC_NUMERIC
 LC_MESSAGES="POSIX"

4. Установка часового пояса

Системные часы по Гринвичу

Если системные часы машины в BIOS Setup установлены на время по Гринвичу (UTC), для настройки часового пояса достаточно создать символическую ссылку /etc/localtime на файл, описывающий данный часовой пояс. Например, для установки московского времени это будет выглядеть так:

Листинг 4.1: Установка московского времени
 ln -sf /usr/share/zoneinfo/Europe/Moscow /etc/localtime

Примечание: Точное обозначение часовых поясов для России можно посмотреть в каталогах /usr/share/zoneinfo/Europe (имеет силу также для Украины и Белоруссии) и /usr/share/zoneinfo/Asia (для азиатской части России). Например, пользователям из Петропавловска-Камчатского указанная команда потребует в следующем виде:

Листинг 4.2: Установка времени для Петропавловска-Камчатского
 ln -sf /usr/share/zoneinfo/Asia/Kamchatka /etc/localtime

Системные часы по местному времени

Если системные часы данной машины установлены по местному времени, кроме создания указанной ссылки потребуются внести еще и изменения в файл /etc/rc.conf, а именно: строку CLOCK="UTC" заменить на CLOCK="local".

5. Локализация оконной системы X

Введение

Локализация оконной системы X (конкретно — её свободной реализации XFree86) может быть выполнена двумя путями: с помощью специальных утилит конфигурирования (обычно — xf86config, используемой для настройки графического режима вообще) или прямым редактированием конфигурационного файла /etc/X11/XF86Config.

Локализация с использованием утилиты xf86config

Утилита `xf86config` в большинстве случаев позволяет корректно выполнить кириллизацию системы XFree86 уже на стадии начальной её настройки. Для этого необходимо дать (в консольном режиме) одноименную команду и правильно ответить на вопросы, относящиеся к определению языково-зависимых параметров, и следующие после указания выбора типа клавиатуры.

Первый из таких вопросов — выбор страны (Enter a number to choose the country), ответ на который, на самом деле, определяет базовую раскладку клавиатуры. Очевидными ответами являются 8 Belarusian для Белоруссии и 70 Ukrainian для Украины, тогда как для России предлагается два варианта: 53 Russian и 54 Russian (cyrillic phonetic). Первый — это традиционная раскладка `qwerty`, второй же — фонетическая раскладка `uwertj`, в которой символы кириллицы расположены на месте созвучных латинских. Она использовалась на старых терминалах и ныне практически не встречается. Так что на самом деле и тут выбор 53 Russian практически однозначен.

Далее следует предложение указать вариант русской раскладки (Please enter a variant name for 'ru' layout). Если, просто нажав Enter, отказаться от этого, будет установлен так называемый DOS-вариант раскладки (вернее, маркировки клавиш); маркированные таким образом клавиатуры практически вышли из употребления, поэтому следует, введя `winkeys`, выбрать Windows-вариант (не следует путать его с Windows-кодировкой, иначе говоря — `cp1251`).

Затем, введя `y`, следует положительно ответить на вопрос об указании дополнительных опций расширения XKB (Do you want to select additional XKB options (group switcher, group indicator, etc.)?): это позволит определить переключатель латиница/кириллица (в силу исторических причин русская раскладка клавиатуры в XFree86 по умолчанию такового не имеет вообще).

В результате следующим шагом будет предложение выбрать клавишу или клавишную комбинацию для такого переключателя. Возможные варианты включают:

Листинг 5.1: Варианты выбора переключателя латиница/кириллица

- 1 R-Alt switches group while pressed
- 2 Left Win-key switches group while pressed
- 3 Right Win-key switches group while pressed
- 4 Both Win-keys switch group while pressed
- 5 Right Alt key changes group
- 6 Caps Lock key changes group
- 7 Menu key changes group
- 8 Left Win-key changes group

- 9 Right Win-key changes group
- 10 Both Shift keys together change group
- 11 Control+Shift changes group
- 12 Alt+Control changes group
- 13 Alt+Shift changes group

Из них следует выбирать, руководствуясь своими предпочтениями. Однако по вполне понятным причинам можно рекомендовать, чтобы переключатель раскладок был идентичен для текстового (консольного) и графического режимов. ИМХО, лучше всего на эту роль подходит клавиша CapsLock (для этого в консоли должна быть установлена раскладка ru4). При этом первичная функция этой клавиши (фиксация верхнего регистра), как и в консоли, будет выполняться одновременным нажатием Shift+CapsLock

Примечание: Не менее удобным представляется и использование в качестве переключателя какой-либо из т.н. Windows-клавиш, имеющих на всех современных клавиатурах. Однако при этом для достижения идентичности переключения в консоли и в XFree86 потребуются расширенные раскладки клавиатуры из упомянутого выше пакета console-tools-cyrillic.

Следующие несколько вопросов позволяют указать переключатель для третьей клавиатурной раскладки, дополнительные позиции управляющей клавиши Control, индикацию текущей раскладки (например, индикаторами CapsLock или ScrollLock) и ещё некоторые опции. Если в них нет необходимости — ответы на эти вопросы можно просто пропустить, нажимая клавишу Enter.

Покончив с настройкой клавиатуры, следует завершить конфигурирование XFree86 (включая настройку видеорежимов) и выйти из программы xf86config с сохранением результатов в файле /etc/X11/XF86Config.

Установка кириллических шрифтов

Программа xf86config обеспечивает настройку клавиатурного ввода, но не подключает по умолчанию шрифтов с символами кириллицы — это достигается редактированием файла /etc/X11/XF86Config. Его следует открыть в любом текстовом редакторе и отыскать в Section "Files" описание путей к шрифтовым файлам.

Предупреждение: При вызове файла /etc/X11/XF86Config необходимо отключить режим переноса строк в используемом редакторе.

Пути к файлам шрифтов описываются следующим образом:

Листинг 5.2: Описание путей к файлам шрифтов

```
FontPath "/usr/X11R6/lib/X11/fonts/local/"
FontPath "/usr/X11R6/lib/X11/fonts/misc/"
FontPath "/usr/X11R6/lib/X11/fonts/75dpi:unscaled"
FontPath "/usr/X11R6/lib/X11/fonts/100dpi:unscaled"
FontPath "/usr/X11R6/lib/X11/fonts/Type1/"
FontPath "/usr/X11R6/lib/X11/fonts/Speedo/"
```

```
FontPath "/usr/X11R6/lib/X11/fonts/75dpi"
```

```
FontPath "/usr/X11R6/lib/X11/fonts/100dpi"
```

В этот список следует внести путь к файлам шрифтов с поддержкой кириллицы. По умолчанию в любой современной версии XFree86 имеется один набор кириллических шрифтов — Cronyx, расположенный в каталоге `/usr/X11R6/lib/X11/fonts/cyrillic/`. И потому первая строка указанной секции должна принять вид

Листинг 5.3: Описание пути к файлам кириллических шрифтов

```
FontPath "/usr/X11R6/lib/X11/fonts/cyrillic/"
```

Важно: Строка с кириллическими шрифтами должна быть первой в списке путей, дабы именно они обнаруживались бы любыми программами в первую очередь при совпадении их имен или псевдонимов с таковыми шрифтов, не имеющих символов кириллицы.

Примечание: Шрифты производства Cronyx содержат ограниченное число гарнитур. Кроме того, они являются растровыми и вообще далеки от эстетического совершенства. Поэтому их по возможности лучше дополнить (или заменить) шрифтами от независимых производителей. В качестве таковых можно рекомендовать коллекцию масштабируемых (TTF и ATM) лицензионно чистых шрифтов Валентина Филиппова, входящих в российскую сборку OpenOffice и в последние версии дистрибутивов Altlinux.

Ручная настройка клавиатуры для ввода кириллицы

К кириллизации XFree86 посредством программы `xf86config` целесообразно прибегать только при начальной настройке этой системы. Если же в ходе использования Gentoo Linux возникнет необходимость дополнительного конфигурирования (например, изменения положения переключателя раскладок), проще обратиться к прямому редактированию файла `/etc/X11/XF86Config`.

За настройку клавиатуры в нём отвечает Section "InputDevice". В ней, после указания идентификатора клавиатуры (а оконная система X в состоянии поддерживать, даже на локальной машине, более одной физической клавиатуры) и её драйвера, следует описание т.н. правил (rules), определяющих раскладку клавиатуры. Оно имеет вид вроде следующего:

Листинг 5.4: Описание раскладки клавиатуры

```
Option "XkbRules" "xfree86"
```

```
Option "XkbModel" "pc105"
```

```
Option "XkbLayout" "ru"
```

```
Option "XkbVariant" "winkeys"
```

```
Option "XkbOptions" "grp:caps_toggle,grp_led:caps"
```

Здесь, разумеется, можно изменить всё, что угодно. Однако практически имеет смысл рассмотреть только возможные варианты значений для строки Option "XkbOptions". Значение, приведенное в примере, соответствует переключателю раскладок CapsLock (grp:caps_toggle) и индикатору Capslock как показателю включения русской раскладки (grp_led:caps). И то, и другое можно изменить, если не устраивает. Доступные значения содержатся (вместе с их краткими описаниями) в секции ! option файла /usr/X11R6/lib/X11/xkb/rules/xfree86.lst

Руководство Gentoo Linux ALSA

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/alsa-guide.xml>

С версии: 1.0

Обновлено: 1.2

1. Введение

Что такое ALSA?

ALSA, или **Advanced Linux Sound Architecture** — продвинутая звуковая архитектура Linux, позволяет работать аудио и **MIDI (Musical Instrument Digital Interface** — Цифровой интерфейс музыкальных инструментов) в операционной системе Linux. ALSA является основной звуковой подсистемой в ядрах 2.6, заменяя собой OSS (Open Sound System — Открытая звуковая система), которая использовалась в ядрах 2.4.

Главные преимущества ALSA включают эффективную поддержку всех типов аудио интерфейсов, начиная от широко распространённых аудио карт и заканчивая профессиональным звуковым оборудованием, полностью модульные драйверы, поддержку многопроцессорных систем и потоковую безопасность, обратную совместимости с OSS, а также пользовательскую библиотеку `alsa-lib`, делающую разработку приложений лёгкой.

ALSA в Gentoo

Одна из сильных сторон Gentoo заключается в предоставлении пользователю максимального контроля над тем, как система установлена/сконфигурирована. ALSA в Gentoo следует этому принципу. Существуют два способа, с помощью которых вы можете установить и запустить ALSA на вашей системе. Мы подробно рассмотрим их в следующей главе.

2. Установка ALSA

Варианты

Предупреждение: Способы, описываемые ниже, являются взаимно исключающими. Вы не можете одновременно собрать ALSA в ядре и установить `media-sound/alsa-driver`. Это не получится.

Два варианта установки ALSA драйверов:

1. Использовать ALSA драйверы, предоставляемые вашим ядром. Этот метод предпочтительный и рекомендуемый.
2. Использовать пакет `media-sound/alsa-driver`.

Драйверы, предоставляемые ядром, могут немного отличаться от предоставляемых пакетом **alsa-driver**; возможности и исправления одного могут оказаться всё ещё не включёнными в другой. Разработчики ALSA осознают ситуацию, но эти два драйвера по существу являются отдельными ветвями проекта ALSA; они не идентичны. Вы должны понимать, что они могут по-разному функционировать, поэтому если один из них у вас не работает, попробуйте другой! Мы бегло рассмотрим оба варианта перед принятием окончательного решения.

Преимущества и недостатки использования ALSA драйверов, предоставляемых ядром:

ALSA в ядре за и против

- + Нет необходимости устанавливать ещё один пакет; драйверы включены в ядро
- + Единое решение, никаких повторных команд emerge
- Может немного отличаться от alsa-driver

А если вы выберете alsa-driver, то:

Драйверы ALSA за и против

- + Самые свежие драйверы от проекта ALSA
- + Удобно, если вы собираетесь разрабатывать драйверы для аудио устройств
- Каждая пересборка ядра требует повторной переустановки alsa-driver
- Определенные параметры конфигурации ядра должны быть отключены

Итак...

Как сказано выше, отличия между драйверами из пакета alsa-driver и ALSA драйверами, поставляемыми с ядром, очень незначительны. Так как между ними нет большой разницы, сначала попробуйте ALSA драйвера, поставляемые ядром, так как их проще использовать. Перед тем как сообщить о любой проблеме, связанной со звуком, в [Gentoo Bugzilla](#), пожалуйста, попробуйте её воспроизвести, используя alsa-driver, и создайте запрос об ошибке вне зависимости от результата.

Перед тем как вы продолжите

Какие драйверы использует ваша карта. В большинстве случаев звуковые карты (встроенные и подключаемые) основаны на PCI, и lspci поможет вам раскопать необходимую информацию. Пожалуйста, если вы ещё не установили lspci, установите командой emerge sys-apps/pciutils. Если у вас USB звуковая карта, вам может помочь lsusb из sys-apps/usbutils. Для карт ISA попробуйте sys-apps/isapnptools. Кроме того, следующие ресурсы могут помочь владельцам ISA звуковых карт:

- [Страница ISAPNPTOOLS](#)
- [Статья о PnP в LinuxJournal](#)

- [TLDP Sound HOWTO](#)

Примечание: Ради простоты в оставшейся части руководства мы предположим, что у пользователя звуковая карта, основанная на PCI.

Теперь мы попробуем найти информацию о звуковой карте.

Теперь мы попробуем найти информацию о звуковой карте.

Листинг 2.1: Подробности звуковой карты

```
# lspci -v | grep -i audio
0000:00:0a.0 Multimedia audio controller: Creative Labs SB Live! EMU10k1 (rev 06)
```

Теперь мы знаем, что звуковая карта, установленная в компьютере, — Sound Blaster Live!, а производителем является Creative Labs. Зайдём на страницу с [таблицей звуковых карт ALSA](#) и выберем Creative Labs из списка. В результате вы попадёте на страницу, содержащую таблицу продуктов Creative Labs, из которой вы можете узнать, что SB Live! использует модуль emu10k1. Эта та информация, которая нам и нужна. Если вам интересна более подробная информация, то вы можете перейти по ссылке «Details» на страницу, посвящённую emu10k1.

Если вы намереваетесь использовать MIDI, то перед установкой любых пакетов ALSA необходимо добавить midi к USE-флагам в файле /etc/make.conf. Ниже в руководстве мы продемонстрируем, как [настроить MIDI](#).

Использование ALSA драйверов, предоставляемых ядром

Если вам нравится идти по пути наименьшего сопротивления, то этот способ для вас.

Примечание: Начиная с выпуска 2005.0, Gentoo Linux в качестве основного ядра использует ядра ветки 2.6. Пожалуйста, удостоверьтесь, что у вас ядро ветки 2.6. Этот способ не применим для ядер ветки 2.4.

А теперь давайте сконфигурируем ядро так, чтобы включить в нём ALSA.

Важно: Пользователи genkernel должны запустить genkernel --menuconfig all и следовать инструкциям из раздела [Параметры ядра для ALSA](#).

Листинг 2.2: Погружение в исходные коды

```
# cd /usr/src/linux
# make menuconfig
```

Примечание: В только что приведённом примере предполагалось, что символическая ссылка /usr/src/linux указывает на исходные коды используемого вами ядра. Пожалуйста, перед тем как продолжить, проверьте, что у вас так оно и есть.

Теперь рассмотрим часть параметров конфигурации ядра 2.6, которые гарантируют работоспособность ALSA с нашей звуковой картой.

Обратите внимание, что во всех примерах мы собираем ALSA модулями. Мы советуем вам поступать так же, так как в дальнейшем это позволит использовать `alsacnf`, упрощающую настройку звуковой карты. Пожалуйста, не пропустите раздел [Настройка](#). Если вы всё же решаете не использовать модули, удостоверьтесь, что вы соответствующим образом изменили вашу конфигурацию.

Листинг 2.3: Параметры ядра для ALSA

Device Drivers --->

Sound --->

(Это необходимо включить)

<M> Sound card support

(Убедитесь, что OSS отключена)

Open Sound System --->

< > Open Sound System (DEPRECATED)

(Вернитесь на шаг назад и войдите в раздел ALSA)

Advanced Linux Sound Architecture --->

<M> Advanced Linux Sound Architecture

(Выберите, если вам нужен MIDI sequencing и routing)

<M> Sequencer support

(Поддержка старых `/dev/mixer*` и `/dev/dsp*`. Рекомендуется.)

<M> OSS Mixer API

<M> OSS PCM (digital audio) API

(Теперь вы можете выбрать устройства, поддержка которых вам требуется. Обычно в системе есть только одна звуковая карта. Если у вас их несколько, включите поддержку для каждой.)

(Для тестирования и разработки, обычным пользователям они не требуются, только если вы знаете, что делаете...)

Generic devices --->

(Для звуковых карт ISA)

ISA devices --->

(Если у вас Gravis, включите этот параметр)

<M> Gravis UltraSound Extreme

(Перейдите на один уровень назад и войдите в раздел PCI-устройств.

Большинство современных звуковых карт являются таковыми)

PCI devices --->

(Теперь выберем драйвер `emu10k1` для нашей карты)

<M> Emu10k1 (SB Live!, Audigy, E-mu APS)

(Или для карты Intel)

<M> Intel/SiS/nVidia/AMD/ALi AC97 Controller

(А может у вас карта на чипсете VIA?)

<M> VIA 82C686A/B, 8233/8235 AC97 Controller

(Вернитесь на один уровень назад и, если у вас звуковая карта USB, включите)

USB Devices --->

Теперь параметры вашего ядра установлены, и вы можете пересобрать ядро. Поддержка ALSA для вашей карты будет доступной сразу, после того как вы загрузитесь с новым ядром. Чтобы использовать новое ядро, не забудьте обновить конфигурацию вашего загрузчика. Теперь, чтобы проверить, что всё работает как должно, переходите к разделу [Утилиты ALSA](#).

Использование пакета драйверов ALSA

Итак, вы решили использовать пакет `alsa-driver`. Тогда начнём. Нужно выполнить несколько небольших действий для того, чтобы был скомпилирован только необходимый вашей аудио карте драйвер. Хотя это и не требуется, это сократит количество лишних драйверов, которые в противном случае были бы собраны.

Если вы не знаете, какие драйверы для звуковой карты вам могут понадобиться, ознакомьтесь с разделом, посвящённым [lspci](#) этого руководства. Как только вы узнаете имя драйвера (`emu10k1` в нашем примере), добавьте переменную `ALSA_CARDS` в файл `/etc/make.conf`.

Листинг 2.4: Добавление `ALSA_CARDS` в `make.conf`
 (Для одной звуковой карты)
`ALSA_CARDS="emu10k1"`
 (Для нескольких карт разделите имена пробелами)
`ALSA_CARDS="emu10k1 via82xx"`

Если вы уже собирали своё ядро и хотите использовать `alsa-driver`, пожалуйста, удостоверьтесь в следующем, перед тем как продолжить, иначе `alsa-driver`, скорее всего, не установится. Следующий перечень поможет вам провести проверку.

Примечание: Пользователи `genkernel` могут продолжить с [установки alsa-driver](#), так как конфигурация их обновлённого ядра по умолчанию соответствует нижеприведённой.

1. `CONFIG_SOUND` включён. (Общая поддержка звука включена)
2. `CONFIG_SOUND_PRIME` выключен. (Встроенная поддержка OSS отключена)
3. `CONFIG_SND` выключен. (Встроенная поддержка ALSA отключена)
4. Символическая ссылка `/usr/src/linux` указывает на то ядро, в котором ALSA будет работать.

Листинг 2.5: `.config` checks
 (Проверяем, что символическая ссылка указывает на нужное ядро.)
`# cd /usr/src/linux`
`# grep SOUND .config`
 (Первый пункт выполнен)
`CONFIG_SOUND=y`
 (Второй пункт выполнен)


```
CONFIG_SOUND_PRIME is not set
# grep SND .config
(Третий пункт выполнен)
CONFIG_SND is not set
```

Теперь всё, что вам нужно сделать, это набрать магические слова... нет, не абракадабру.

Листинг 2.6: Установка alsa-driver
emerge alsa-driver

Важно: Помните, что вам придётся выполнять `emerge alsa-driver` каждый раз после (пере)сборки ядра, так как предыдущие драйверы будут удалены. Чтобы упростить эту задачу, вы можете воспользоваться пакетом `module-rebuild`, который будет вести учёт всем пакетам с модулями ядра и по необходимости пересобирать их. Сначала, чтобы создать список пакетов, выполните `module-rebuild populate`, а затем после (пере)сборки ядра просто запускайте `module-rebuild rebuild`, и все внешние модули ядра будут пересобраны.

3. Настройка/тестирование ALSA

Утилиты ALSA

Пакет **alsa-utils** составляет неотъемлемую часть ALSA и содержит множество крайне полезных программ, в том числе и сценарий инициализации ALSA. Поэтому мы настоятельно советуем вам установить **alsa-utils**.

Листинг 3.1: Установка alsa-utils
emerge alsa-utils

Примечание: Если вы включили поддержку ALSA в вашем [ядре](#), а не собрали её в качестве модулей, пожалуйста, перейдите в раздел [сценарий инициализации ALSA](#). Всё, что осталось сделать всем остальным, это настроить ALSA. Сделать это очень просто благодаря утилите, входящей в состав `alsa-utils` — `alsacnf`.

Настройка

Последние версии `udev` (`>=udev-103`) в некоторой степени предоставляют автоматическое конфигурирование вашей аудио карты на уровне ядра. По возможности старайтесь полагаться на автоматическую настройку и позвольте ядру настроить вашу звуковую карту. В противном случае, чтобы настроить карту, используйте `alsacnf`, как это показано ниже.

Примечание: Пожалуйста, завершите все программы, которые могут обратиться к звуковой карте во время работы **alsacnf**.

Чтобы настроить вашу звуковую карту просто наберите `alsacnf` в командной оболочке с правами `root`.

Листинг 3.2: Запуск `alsacnf`

```
# alsacnf
```

Вы увидите изящный управляемый с помощью меню интерфейс программы, которая автоматически исследует ваши устройства и попытается найти вашу звуковую карту. Вас попросят выбрать вашу звуковую карту из списка. Как только это будет сделано, вас спросят разрешения автоматически сделать необходимые изменения в `/etc/modules.d/alsa`. После этого программа изменит настройки громкости на оптимальный уровень, выполнит `update-modules` и запустит службу `/etc/init.d/alsasound`. Как только `alsacnf` завершит работу, вы сможете продолжить с настройки сценария инициализации ALSA.

Сценарий инициализации ALSA

Мы почти завершили настройку. Вне зависимости от выбранного вами способа установки ALSA, вам понадобится что-то, что будет загружать модули или инициализировать ALSA и восстанавливать настройки громкости при загрузке системы. Сценарий инициализации ALSA, называемый `alsasound`, делает всё это для вас. Добавьте его в загрузочный уровень исполнения.

Листинг 3.3: Добавление ALSA в загрузочный уровень исполнения

```
# rc-update add alsasound boot
* alsasound added to runlevel boot
* rc-update complete.
```

Теперь проверьте файл `/etc/conf.d/alsasound` и убедитесь, что переменная `SAVE_ON_STOP` установлена в значение `yes`, тогда ваши настройки звука при выключении системы будут сохраняться.

Группа audio

Перед тем как начать проверку звука, необходимо сделать одну последнюю важную вещь. Главное правило в операционных системах *nix гласит: «не запускай ничего с правами `root`, если этого не требуется». И в данной ситуации это правило тоже применимо ;) Каким образом? Пожалуй, большую часть времени вы будете работать под учётной записью пользователя, при этом слушать музыку или иметь доступ к звуковой карте. Чтобы вы могли это делать, вы должны быть в группе `audio`. Тут то мы и добавим пользователей в эту группу, чтобы у них не было проблем с доступом к аудио устройствам. Мы воспользуемся `gpasswd`, так что вы должны иметь права `root`, чтобы сделать это.

Листинг 3.4: Добавление пользователей в группу `audio`

(Замените `<имя_пользователя>` на необходимое имя)

```
# gpasswd -a <имя_пользователя> audio
Adding user <имя_пользователя> to group audio
```

Проверка громкости!

Теперь все настройки и необходимые предпосылки выполнены, так что давайте заставим ALSA работать. Если вы запускали утилиту **alsacnf**, то можете пропустить этот шаг, так как **alsacnf** уже всё сделал за вас.

Листинг 3.5: Запуск сервиса
`/etc/init.d/alsasound start`

Теперь мы позаботились обо всём, что может быть необходимым, нам нужно проверить громкость, так как в определённых случаях звук выключен. Для этого мы воспользуемся **alsamixer**.

Листинг 3.6: Запуск **alsamixer**
 (Откройте в терминале. Будут отображены только необходимые настройки)
`# alsamixer`

Важно: Если у вас проблемы с запуском **alsamixer** и вы получаете ошибки, подобные этой: «**alsamixer: function snd_ctl_open failed for default: No such file or directory**», то, скорее всего, проблема в том, как демон **udev** инициализировал устройства. Выполните `killall udevd; udevstart` для перезагрузки устройств в `/dev` и попробуйте снова **alsamixer**. Это должно решить проблему.

Вот так может выглядеть микшер ALSA при первом запуске. Обратите внимание, что уровни каналов Master и PCM занижены и что под ними буквы MM. Это означает, что они выключены. Если вы попытаете что-нибудь проиграть, в то время пока **alsamixer** в таком состоянии, то вы ничего не услышите из ваших динамиков.

Для начала включим каналы и установим необходимые уровни громкости.

Предупреждение: Оба канала Master и PCM должны быть включены и установлены в слышимый уровень громкости, если вы хотите услышать что-нибудь из ваших динамиков.

- Для перемещения между каналами используйте клавиши влево и вправо. (<- и ->).
- Для того чтобы выключить/включить канал, например Master, выберите его и нажмите клавишу m.
- Чтобы увеличить или уменьшить уровень громкости, используйте клавиши вверх и вниз.

Примечание: Будьте осторожны при установке значений для Bass и Treble. Обычно для обоих оптимально значение 50. Слишком высокие значения Bass могут вызватьдребезжание неспособных воспроизводить глубокие басы динамиков.

После того как вы всё сделаете, ваш микшер ALSA должен выглядеть, так, как это показано ниже. Заметьте, что вместо MM стоит 00 и уровни громкости в оптимальном значении.

Проверка звука!

Наконец-то. Какие-нибудь звуки. Если всё прошло успешно, то теперь вы сможете услышать музыку. Быстрый способ проверить — это воспользоваться консольной командой, наподобие media-sound/madplay. Вы также можете использовать что-нибудь более известное, например, mpg123. Если вы поклонник формата OGG, используйте программу ogg123 из пакета media-sound/vorbis-tools. Используйте любой удобный для вас плеер. Как всегда, emerge — всё что вам нужно.

Листинг 3.7: Получение программ
(Установка необходимых приложений)
emerge madplay mpg123
(Для проигрывания ogg-файлов)
emerge vorbis-tools

А теперь проиграем ваш любимый саундтрек...

Листинг 3.8: Проигрывание музыки
madplay -v /mnt/shyam/Music/Paul\ Oakenfold\ -\ Dread\ Rock.mp3
MPEG Audio Decoder 0.15.2 (beta) - Copyright (C) 2000-2004 Robert Leslie et al.
Title: Dread Rock
Artist: Paul Oakenfold
Album: Matrix Reloaded
Year: 2003
Genre: Soundtrack
Soundtrack
00:04:19 Layer III, 160 kbps, 44100 Hz, joint stereo (MS), no CRC

ogg123 Paul\ Oakenfold\ -\ Dread\ Rock.ogg
Audio Device: Advanced Linux Sound Architecture (ALSA) output

Playing: Paul Oakenfold - Dread Rock.ogg
Ogg Vorbis stream: 2 channel, 44100 Hz
Genre: Soundtrack
Transcoded: mp3;160
Title: Dread Rock
Artist: Paul Oakenfold
Date: 2003
Album: Matrix Reloaded
Time: 00:11.31 [04:28.75] of 04:40.06 (200.6 kbps) Output Buffer 96.9%

ALSA и USE

Теперь, для того чтобы приложения, поддерживающие ALSA, были собраны с поддержкой оной, вам нужно добавить USE-флаг `alsa` в файл `/etc/make.conf`. Некоторые архитектуры, например, `x86` и `amd64`, включают этот флаг по умолчанию.

Проблемы?

Если по каким-либо причинам вы не слышите звука, перво-наперво проверьте настройки [alsamixer](#). 80% всех проблем заключаются в выключенных каналах или низкой громкости. Кроме того, проверьте звуковой апплет вашего оконного менеджера и удостоверьтесь, что громкости каналов установлены на слышимом уровне.

`/proc` — ваш друг. В этом случае, `/proc/asound` — ваш лучший друг. Мы бегло посмотрим, сколько полезной информации доступно нам здесь.

Листинг 3.9: Развлечение с `/proc/asound`

(Прежде всего, если `/proc/asound/cards` отображает вашу карту, то ALSA распознала вашу звуковую карту.)

```
# cat /proc/asound/cards
```

```
0 [Live      ]: EMU10K1 - Sound Blaster Live!
                Sound Blaster Live! (rev.6, serial:0x80271102) at 0xb800, irq 11
```

(Следующая команда отобразит текущую версию ALSA)

```
# cat /proc/asound/version
```

```
Advanced Linux Sound Architecture Driver Version 1.0.8 (Thu Jan 13 09:39:32 2005 UTC).
```

(Подробности эмуляции OSS в ALSA)

```
# cat /proc/asound/oss/sndstat
```

```
Sound Driver:3.8.1a-980706 (ALSA v1.0.8 emulation code)
```

```
Kernel: Linux airwolf.zion 2.6.11ac1 #2 Wed May 4 00:35:08 IST 2005 i686
```

```
Config options: 0
```

Installed drivers:

```
Type 10: ALSA emulation
```

Card config:

```
Sound Blaster Live! (rev.6, serial:0x80271102) at 0xb800, irq 11
```

Audio devices:

```
0: EMU10K1 (DUPLEX)
```

Synth devices: NOT ENABLED IN CONFIG

Midi devices:

```
0: EMU10K1 MPU-401 (UART)
```

Timers:

```
7: system timer
```

Mixers:

Другой очень распространённой проблемой среди пользователей является странная ошибка «Unknown symbol in module» («Неизвестный символ в модуле»).

Пример её появления показан ниже.

Листинг 3.10: Ошибка: неизвестный символ в модуле

```
# /etc/init.d/alsasound start
```

```
* Loading ALSA modules ...
*   Loading: snd-card-0 ...           [ ok ]
*   Loading: snd-pcm-oss ...
WARNING: Error inserting snd_mixer_oss
(/lib/modules/2.6.12-gentoo-r6/kernel/sound/core/oss/snd-mixer-oss.ko): Unknown
symbol in module, or unknown parameter (see dmesg) FATAL: Error inserting
snd_pcm_oss
(/lib/modules/2.6.12-gentoo-r6/kernel/sound/core/oss/snd-pcm-oss.ko): Unknown
symbol in module, or unknown parameter (see dmesg)
[ !! ]
*   Loading: snd-mixer-oss ...
FATAL: Error inserting snd_mixer_oss
(/lib/modules/2.6.12-gentoo-r6/kernel/sound/core/oss/snd-mixer-oss.ko): Unknown
symbol in module, or unknown parameter (see dmesg)
[ !! ]
*   Loading: snd-seq ...               [ ok ]
*   Loading: snd-emu10k1-synth ...     [ ok ]
*   Loading: snd-seq-midi ...          [ ok ]
* Restoring Mixer Levels ...           [ ok ]
```

И если, как советуют, просмотреть вывод `dmesg`, то, скорее всего, можно увидеть следующее:

Листинг 3.11: Вывод `dmesg`

(Показаны только относящиеся к делу части вывода)

```
# dmesg | less
```

```
ACPI: PCI Interrupt 0000:02:06.0[A] -> Link [APC3] -> GSI 18 (level, low) -> IRQ 209
snd_mixer_oss: Unknown symbol snd_unregister_oss_device
snd_mixer_oss: Unknown symbol snd_register_oss_device
snd_mixer_oss: Unknown symbol snd_mixer_oss_notify_callback
snd_mixer_oss: Unknown symbol snd_oss_info_register
snd_pcm_oss: Unknown symbol snd_unregister_oss_device
snd_pcm_oss: Unknown symbol snd_register_oss_device
snd_pcm_oss: Unknown symbol snd_mixer_oss_ioctl_card
snd_pcm_oss: Unknown symbol snd_oss_info_register
snd_mixer_oss: Unknown symbol snd_unregister_oss_device
snd_mixer_oss: Unknown symbol snd_register_oss_device
snd_mixer_oss: Unknown symbol snd_mixer_oss_notify_callback
snd_mixer_oss: Unknown symbol snd_oss_info_register
```

Эта проблема вызвана переключением с `alsa-driver` на драйверы, предоставляемые ядром, потому что когда вы удаляете `alsa-driver`, то файлы модулей сохраняются системой защиты конфигурации. И поэтому, когда вы переходите на встроенные в ядро драйверы, попытка `modprobe` выдаст вам смесь модулей из `alsa-driver` и встроенных в ядро, вызывая приведённые выше ошибки.

Решение очень простое. Удалите вызывающий проблемы каталог после удаления `alsa-driver`. Проверьте, что удаляете модули правильной, но не текущей версии ядра!

Листинг 3.12: Удаление модулей alsa-driver

```
# rm -rf /lib/modules/$(uname -r)/alsa-driver
```

Другой причиной подобных сообщений может являться файл в `/etc/modules.d`, содержащий параметр `device_mode`, в то время как он не требуется. Проверьте, так ли это, и выясните, какой файл является источником проблем.

Листинг 3.13: Подтверждение проблемы и поиск `device_mode`
 (Проверим вывод `dmesg` для идентификации проблемы)

```
# dmesg | grep device_mode
```

 snd: Unknown parameter `device_mode'
 (И теперь найдём источник проблемы)

```
# grep device_mode /etc/modules.d/*
```

Обычно это файл с именем `alsa`, в котором присутствует строка `options snd device_mode=0666`. Удалите эту строку и перезапустите службу `alsasound`. Это должно решить проблему.

4. Другие возможности ALSA

Настройка поддержки MIDI

Сначала проверьте, что у вас включён USE-флаг **midi** в файле `/etc/make.conf`. Если вы до сих пор этого не сделали, то сделайте это сейчас. Помимо этого вам потребуется пересобрать все пакеты ALSA, использующие флаг `midi`: `alsa-lib`, `alsa-utils` и `alsa-driver`.

Если ваша карта имеет встроенный MIDI синтезатор и вы хотите слушать *.mid файлы, вам придётся установить пакет `awesfx`, содержащий основной набор программ для управления драйвером AWE32. Сначала вам надо установить его. Если у вас нет синтезатора, вы можете использовать виртуальный. Обратитесь к разделу [виртуальные синтезаторы](#) для дополнительной информации.

Листинг 4.1: Установка `awesfx`

```
# emerge awesfx
```

Примечание: Вам понадобится скопировать файлы банка звуков (SoundFont: SF2) с компакт-диска с драйверами для вашей звуковой карты или установленные в Windows в каталог `/usr/share/sounds/sf2/`. Например, файл банка звуков для карты Creative SBLive! может называться `8MBGMSFX.SF2`.

После копирования файлов банка звуков мы сможем проигрывать midi-файлы. Также, для того чтобы банк звуков загружался каждый раз при загрузке системы, вы можете добавить команду **asfxload** в `/etc/conf.d/local.start`.

Примечание: Использованные ниже пути, начинающиеся с `/mnt`, не применимы

для вашего компьютера. Они являются примерами. Пожалуйста, будьте осторожны при изменении путей на соответствующие вашей системе.

Листинг 4.2: Загрузка сэмплов

(Сначала копируем файл с банком звуков)

```
# cp /mnt/win2k/Program\ Files/CreativeSBLive2k/SFBank/8MBGMSFX.SF2 /usr/share/sounds/sf2/
```

(Или берём его с компакт-диска SoundBlaster)

```
# cp /mnt/cdrom/AUDIO/ENGLISH/SFBANK/8MBGMSFX.SF2 /usr/share/sounds/sf2/
```

(Загружаем определённый банк звуков)

```
# asfxload /usr/share/sounds/sf2/8MBGMSFX.SF2
```

Теперь вы можете проигрывать midi-файлы, используя программу, подобную aplaymidi. Запустите aplaymidi -l, чтобы получить список доступных портов, а затем задействуйте один из них для проигрывания файла.

Листинг 4.3: Проигрывание MIDI

(Проверка открытых портов)

```
# aplaymidi -l
```

Port	Client name	Port name
64:0	EMU10K1 MPU-401 (UART)	EMU10K1 MPU-401 (UART)
65:0	Emu10k1 WaveTable	Emu10k1 Port 0
65:1	Emu10k1 WaveTable	Emu10k1 Port 1
65:2	Emu10k1 WaveTable	Emu10k1 Port 2
65:3	Emu10k1 WaveTable	Emu10k1 Port 3

(Выбираем порт и проигрываем mid-файл)

```
# aplaymidi --port=65:0 /mnt/shyam/music/midi/mi2.mid
```

Виртуальные синтезаторы

Если у вашей карты отсутствует встроенный синтезатор, вы можете использовать виртуальный, например, timidity++. Установка происходит на одном дыхании.

Листинг 4.4: Установка timidity++

```
# emerge timidity++
```

Чтобы **timidity** воспроизводил звуки, ему нужен набор сэмплов (или банк звуков). К счастью, вместе с пакетом устанавливаются несколько банков звуков. Есть ещё несколько пакетов с банками звуков в Portage, например, timidity-freepats и timidity-eawpatches. Вы можете установить несколько банков звуков, а также можете разместить собственный банк звуков в каталог /usr/share/timidity/. Для переключения timidity между разными банками звуков используйте eselect.

Листинг 4.5: Изменение конфигураций

```
# eselect timidity list
```

```
# eselect timidity set eawpatches
```

Не забудьте добавить timidity в основной уровень исполнения.

Листинг 4.6: Добавление timidity в основной уровень исполнения


```
# rc-update add timidity default
# /etc/init.d/timidity start
```

Теперь вы можете попробовать [проиграть MIDI](#) файлы.

Утилиты и Firmware

Для некоторых звуковых карт могут быть полезны утилиты из пакетов **alsa-tools** и **alsa-firmware**. Вы можете установить любой из этих пакетов, просто запустив emerge.

Листинг 4.7: Установка утилит ALSA

```
# emerge alsa-tools
```

Несколько звуковых карт

Вы можете использовать больше одной звуковой карты, при условии что вы собрали ALSA как модули ядра (или из пакета alsa-driver). Сначала в файле /etc/modules.d/alsa вам следует лишь указать, какая из карт должна быть запущена первой. В этом файле карты идентифицируются по именам своих драйверов. 0 означает первую карту, 1 — вторую, и так далее. Вот пример для системы, в которой присутствуют две звуковые карты.

Листинг 4.8: Две карты в файле /etc/modules.d/alsa

```
options snd-emu10k1 index=0
options snd-via82xx index=1
```

Или если у вас две карты, использующие один и тот же драйвер, то их следует указать на одной строке, разделяя цифры запятой. Ниже приведён пример системы, в которой установлено три звуковые карты, две из которых являются картами Intel High Definition Audio.

Листинг 4.9: Несколько звуковых карт в /etc/modules.d/alsa

```
options snd-ymfpcl index=0
options snd-hda-intel index=1,2
```

Модули расширения

Для расширения возможностей ALSA вы можете установить дополнительные расширения. Пакет alsa-plugins является собранием полезных модулей расширения, который включает: модуль вывода PulseAudio, конвертер частоты дискретизации, jack (аудио сервер с малым временем задержки) и кодек, позволяющий вам выводить шестиканальный звук через цифровой S/PDIF-вывод (оптический или коаксиальный). Вы можете выбрать те расширения, которые вы хотите установить, добавив соответствующие USE-флаги в /etc/portage/package.use.

Листинг 4.10: Установка alsa-plugins

```
# emerge -avt alsa-plugins
```

Огромное спасибо вам всем...

Всем, кто принимал участие в написании ранних версий руководства ALSA в Gentoo: Vincent Verleye, Grant Goodyear, Arcady Genkin, Jeremy Huddleston, John P. Davis, Sven Vermeulen, Benny Chuang, Tiemo Kieft и Erwin. А также DrJ[aM за

перевод ранней версии руководства на русский язык.

Ссылки

- [Домашняя страница проекта ALSA](#)
- [Linux Sound/MIDI Software](#)

Советы пользователям Gentoo (вариант 2)

Ссылка на оригинал: <http://lxj.endofinternet.net/column/gentoo-tips/>

С версии: 1.5

Автор: [Алексеев Алексей](#)

Дата: 12.02.2008

В своё время я написал две статьи для [Linux.com](#) с советами для пользователей Gentoo ([Gentoo Portage secrets](#) и [Tips for new Gentoo users](#)), причём вторая оказалась такой удачной, что не только попала в топ digg'a, но и была в тот же день переведена на русский язык!

Я решил собрать в одной статье всю актуальную информацию из обеих статей (они, в принципе, остались актуальными на 90%), и добавить немного нового материала. Ну и перевести, в конце концов, так, как я считаю нужным :)

Если вы только собираетесь начать работу с Gentoo

Приготовьтесь читать **много** документации, особенно поначалу. Собственно, вы и установить Gentoo не сможете, не изучив [Gentoo Handbook](#).

Есть, конечно, графический инсталлятор, есть [Sabayon](#), но без понимания основ вы далеко не уйдёте. Да и вообще, при работе с Gentoo не рассчитывайте на графические средства настройки — зато консольные средства настройки и организация конфигурационных файлов в Gentoo весьма удобны.

Handbook освещает установку и основы работы с Gentoo. Когда придёт время настраивать иксы и ALSA, переходите к [Gentoo Documentation Resources](#). Если вы не нашли информации по интересующей вас теме и там, то она скорее всего есть в [Gentoo wiki](#).

Найти что-либо, что не описано в Gentoo wiki довольно сложно, так что не забывайте туда заглянуть, прежде чем посетить официальный [форум](#) и [IRC-каналы](#).

Выше я приводил ссылки на англоязычные источники, как и в оригинале статьи. Если у вас нет проблем с английским языком, то лучше их и придерживаться, когда вы что-либо ищите — англоязычная документация практически всегда более актуальная и более полная. Но и документации на русском есть немало: и [Handbook](#), и [официальный перевод прочей документации](#), [русскоязычный раздел Gentoo Wiki](#). Большой русскоязычный форум о Gentoo вы можете найти на [Gentoo.ru](#) (там же вы найдёте ссылки на IRC-каналы), ну и крупнейший русскоязычный форум о Linux — [LinuxForum](#).

Основы Portage

Шутка. Тут этот раздел вообще не нужен. Вы ведь прочли Handbook, как это было сказано в предыдущем разделе? Хотя в оригинальную статью этот раздел по настоятельной просьбе редакторов пришлось всё-таки включить.

Настройка Portage

Первое правило Gentoo — внимательно следите за вашими USE-флагами. Множество глупых вопросов снимается, если вы перепроверите свои USE-флаги. Почему Midnight Commander тянет за собой в зависимостях X.org? Потому что у вас установлен флаг X. Почему вместе с KDE не установился KPDF? Потому что вы забыли про флаг pdf. *(Почему же всё-таки был установлен KGhostScript? Понятия не имею, да и вообще — KDE4 на носу)*

Для того, чтобы увидеть, с какими флагами будет установлен новый пакет, запускайте emerge с параметрами `—ask` и `—verbose (-av)` (`—ask` лучше `—pretend` тем, что зависимости не пересчитываются лишний раз). Обязательно установите пакет `gentoolkit` — среди прочих полезностей, он содержит утилиту `euse`, которая облегчает управление USE-флагами. Запустите `euse -i flag`, чтобы увидеть описание флага, и узнать, активирован ли он. Прочитайте `man euse`, чтобы узнать о всех возможностях этой команды (нет, правда, прочитайте — там немного).

Чтобы установить какую-либо программу с другими USE-флагами, иногда советуют запустить `USE="some flags" emerge moo`, но ни в коем случае так не делайте: при следующем обновлении пакет будет перекомпилирован со старым набором флагов. Для настройки USE-флагов для отдельных пакетов используйте `/etc/portage/package.use` (это или файл, или каталог с файлами, см. ниже). То есть если вы хотите установить пакет `moo` категории `category` с флагами `some flags`, то наберите `echo "category/moo some flags" >> /etc/portage/package.use` перед установкой.

То же самое относится и к `ACCEPT_KEYWORD` — используйте `/etc/portage/package.keywords`. Начиная с версии Portage 2.1, вы можете хранить такие настройки не в одном большом файле `package.keywords`, а создать несколько небольших файлов, и положить их в каталог `/etc/portage/package.keywords`. Это относится ко всем файлам в `/etc/portage`.

Когда нужно использовать оверлей

Иногда бывает нужно установить пакет (или версию пакета), которого ещё (или уже) нет в Portage. На пример, вы можете найти `ebuild`, который уже был удалён из официального дерева в [CVS-репозитории](#). Или же вы нашли `ebuild` для нового пакета в [Gentoo Bugzilla](#). А может быть, вас чем-то не устраивает официальный `ebuild`, и вы решили внести в него свои изменения.

В таком случае, вы должны создать собственный локальный оверлей, и положить свои `ebuild`'ы туда. Оверлей — это набор `ebuild`'ов, которые вы хотите «подмешать» к основной базе Portage. Ни в коем случае не храните свои `ebuild`'ы в `/usr/portage` — при первой же синхронизации они будут удалены. Создайте каталог для своего оверлея, и создайте в нём структуру каталогов, подобную `/usr/portage`. Расположите ваши `ebuild`'ы соответственно. Затем откройте `/etc/make.conf`, и добавьте строчку `PORTDIR_OVERLAY="/path/to/local/overlay"`. Обычное место для оверлеев — `/usr/local/overlays`, так что свой оверлей вы можете сделать в `/usr/local/overlays/local`. Что ещё может оказаться в `/usr/local/overlays`?

Есть множество неофициальных оверлеев, с которыми можно синхронизироваться. Для любого из них можно создать оверлей в `/usr/local/overlays`, и синхронизировать их (по тому же принципу, как вы

синхронизируете официальное дерево Portage, когда запускаете `emerge --sync`). Разные неофициальные оверлеи используют для синхронизации утилиты `rsync`, `svn` или `git`, что оговаривается в описании оверлея.

В прочем, большинство из этих оверлеев вы можете использовать при помощи утилиты `layman`. В таком случае вам не нужно самостоятельно создавать каталог для оверлея и запускать `rsync/svn/git` для получения обновлений — вы просто вводите `layman -a название-оверлея`, чтобы «подключиться» к оверлею — то есть, чтобы `layman` сам создал каталог, скачал туда нужные файлы и добавил путь в файл настройки; а затем запускаете `layman -S`, чтобы обновить все оверлеи.

Подробнее об управлении оверлеями, об установке `layman` и о том, какие вообще есть полезные оверлеи, можно узнать [на Gentoo wiki](#). Оверлеи, как правило, обсуждаются в разделе [Unsupported software](#) официальных форумов Gentoo.

Как часто нужно обновляться?

Не стоит запускать `emerge --sync` и `layman -S` чаще раза в сутки. Да и этого, как правило, более чем достаточно. Вы можете отслеживать появление интересующих вас пакетов на сайтах <http://packages.gentoo.org/> или <http://gentoo-portage.com/> (или же подписавшись на соответствующие RSS-ленты), и, таким образом, синхронизировать дерево Portage и обновлять пакеты только тогда, когда в официальном дереве появятся новые версии используемого вами ПО.

Чтобы не пропускать важные обновления безопасности, используйте утилиту [glsa-check](#) из пакета `gentoolkit`. В настоящее время эта функциональность ещё дорабатывается, но в будущем `glsa-check` станет частью Portage. Чтобы узнать, есть ли уязвимости среди установленных пакетов, запустите `glsa-check -l affected`. Выполнив `glsa-check -d <код уязвимости>`, вы увидите подробную информацию по определённой уязвимости (вы можете также сразу запустить `glsa-check -d affected`).

В то же время, не обновляетесь слишком редко. Если вы не будете обновлять систему больше месяца, то когда вы в конце концов соберётесь обновляться, у вас возникнет множество проблем.

Не забывайте читать [Gentoo Monthly Newsletter](#), практически все важные вопросы связанные с нетривиальными обновлениями, там всегда освещаются. Да и вообще подписаться на RSS-ленту новостей с `gentoo.org` (или хотя бы `gentoo.ru`) — это очень хорошая идея.

[Этот скрипт](#) с простым названием `update` упрощает процедуру обновления:

- на экран не выводятся сообщения от компилятора — только сообщения от `emerge`;
- если какой-то пакет не удаётся установить, то выполнение обновления не прерывается: скрипт в конце работы пытается ещё раз установить этот пакет, затем выводит список всех пакетов, которые так и не удалось установить;
- в некоторых случаях автоматически разрешаются блокировки пакетов (пакет А версии ниже X блокируют установку пакета В версии Y: обычно `emerge` не сможет продолжить работу, пока вы сами не обновите пакет А, а `update` это делает сам)

Держите «мир» в чистоте

Каждый раз, когда вы устанавливаете какой-либо пакет, его название записывается в `/var/lib/portage/world`. Когда вы обновляете ваши пакеты, вы запускаете что-то вроде `emerge --update --deep world` — именно тогда этот список и используется. Старайтесь, чтобы он был как можно короче; только программы, которые вы используете напрямую, должны туда попадать. Если какой-либо пакет установлен как зависимость другого, то он скорее всего не должен попадать в `world`.

Например, когда вы устанавливаете пакет, использующий библиотеку `wxWidgets`, то она автоматически устанавливается среди прочих зависимостей этого пакета, но не добавляется в `world`. Если вы удалите все пакеты, которые зависят от `wxWidgets`, то она перестанет обновляться при `emerge --update (...) world`; и будет удалена из системы, если вы запустите `emerge --depclean`. Только если вы сами разрабатываете программы с использованием `wxWidgets`, вы захотите, чтобы она входила в `world`.

Утилита `udept` может помочь вам удалить лишние пакеты из списка `world`:

```
emerge udept
dep -aw
```

Чтобы установить или обновить пакет напрямую, но без добавления его в этот список, используйте опцию `--oneshot (-1)`. Это может быть полезно, на пример, для временной установки какого-либо пакета.

Об удалении пакетов.

Не забывайте, что `emerge --unmerge` не проверяет, используется ли удаляемый пакет другими пакетами. Также будьте осторожны с `emerge --depclean`, так как иногда она может удалить пакеты, которые вам нужны.

Иногда при обновлении одного пакета, перестаёт работать другой, который, хоть и не требует для установки первый пакет, но включает его в свои необязательные зависимости. В таких случаях необходимо просто переустановить второй пакет, чтобы он скомпилировался без связей с первым. Чтобы выявить все такие пакеты, вы можете использовать утилиту `revdep-rebuild` из пакета `gentoolkit`.

Логи Portage

Все ваши действия с `emerge` отражаются в `/var/log/emerge.log`. К примеру, чтобы узнать, какой пакет сейчас компилируется, запустите `tail /var/log/emerge.log`. Преобразовать временные отметки в читаемый формат поможет простой [python-скрипт](#) (с его использованием необходимо запустить `tail /var/log/emerge.log | formatlog`).

Утилита `genlop` (`emerge genlop`) позволяет извлечь статистическую информацию из этого лог-файла. На пример, `genlop -s` скажет, какой пакет сейчас компилируется и как долго, и оценит количество оставшегося до конца установки времени.

Дополнительно стоит также включить так называемые `eelog`. При установке многих пакетов показываются важные уведомления, которые вы можете пропустить, если не следите за выполнением `emerge`. Добавьте в `make.conf` следующие строки:


```
# This sets what to log
PORTAGE_ELOG_CLASSES="warn error log"
# And this is how to do it
PORTAGE_ELOG_SYSTEM="save"
```

и создайте каталог `/var/log/portage/elog`. Теперь уведомления для каждого пакета будут сохраняться в отдельном файле в созданном каталоге. Если вы хотите узнать больше о настройке `elog`, то изучите файл `/etc/make.conf.example`. Существуют также [графические утилиты](#) для просмотра этих логов.

Удаление временных файлов

Если установка программы прервана ошибкой, в `/var/tmp/portage` остаётся временный каталог. Он будет автоматически удалён только после успешной установки этой программы, или же вы можете очистить этот каталог сами.

Также через некоторое время довольно много файлов может оказаться в `/usr/portage/distfiles`. Используйте утилиту `eclean` из пакета `gentoolkit`, чтобы удалить дистрибутивы, которые вам больше не нужны. Эта утилита также удаляет старые бинарные пакеты, если вы их создавали (запускайте как `eclean-dist` и `eclean-pkg` соответственно). В странице руководства рассказывается о дополнительных параметрах.

Быстрый поиск пакетов

Чтобы найти нужный пакет в локальном дереве Portage (и/или оверлеях), вы можете использовать `emerge --search` или `emerge --searchdesc`, но этот способ очень медленный. Поэтому лучше использовать [eix](#). Eix индексирует список пакетов после каждой синхронизации (для этого необходимо набрать `update-eix`), а затем очень быстро выполняет поиск по пакетам (к примеру, `eix moo` выдаст список всех пакетов, которые содержат `moo` в названии). К тому же, вывод `eix` более информативен, чем вывод `emerge`.

Eix позволяет искать по названию пакета, категории или описанию, использовать регулярные выражения, неточный поиск, ограничивать область поиска только установленными пакетами и многое другое. Как обычно, читайте `man`.

Уменьшение использования трафика

Обновление системы может быть связано с тратой большого количества сетевого трафика. Но есть способы уменьшить аппетиты Portage. Наиболее эффективно в этом преуспевает знаменитая утилита [Deltup](#), которая позволяет загружать не весь архив с исходными кодами, а только разность между новой и старой версией. Такой подход экономит до 90% размера загрузок. Установка Deltup [описана](#) в Gentoo Wiki.

У Deltup, конечно, есть свои недостатки. Эта утилита не поддерживает докачку, а кроме того, вы зависите от загруженности сервера, генерирующего новые патчи (помнится, были плохие времена у `deltup`, когда дожидаться очереди на генерацию было практически невозможно. Потом стало лучше, а после этого я уже давно `deltup` не пользуюсь — как там сейчас, не знаю). Также иногда Deltup «путается» в версиях некоторых пакетов, если у них запутанная система нумерации версий. Ну и

конечно, пересоздание нового пакета требует процессорного времени. Однако же, никаких катастрофических последствий использование `deltup` иметь не может, так что если вы нуждаетесь в экономии трафика — обязательно её попробуйте.

Похожий подход, но для синхронизации дерева Portage использует утилита `emerge-delta-webrsync`: она загружает ежедневные патчи для обновления вашей локальной копии дерева, что гораздо экономнее, чем `emerge --sync` (особенно если у вас уже есть архив с копией дерева портежей — вроде того, который вы использовали при установке). Просто запустите `emerge emerge-delta-webrsync`, а затем `emerge-delta-webrsync` для синхронизации дерева портежей (мне всё ещё надо напоминать, что подробности — в `man`е?).

На Gentoo wiki есть [специальный раздел](#) с советами для пользователей с плохим интернет-соединением.

Ещё одна возможность не уменьшит использование трафика, но позволит уменьшить время загрузки пакетов: добавьте `parallel-fetch` в переменную `FEATURES` в `/etc/make.conf`, чтобы `emerge` скачивал дистрибутив следующего пакета, устанавливая предыдущие в очереди. Это уменьшит время установки, если вы запускаете `emerge` для установки сразу нескольких пакетов.

Ускорение компиляции

Компиляция из исходных кодов — основной способ установки программ в Gentoo. Неудивительно, что появились средства, позволяющие ускорить этот процесс.

Например, [Distcc](#) позволяет распределить задачи по компиляции на несколько компьютеров в сети. Даже компьютеры с Windows [сгодятся](#) для этого!

[Ccache](#) кэширует результаты компиляции, таким образом, ускоряя компиляцию одинаковых участков кода.

Упомянутые здесь утилиты можно использовать не только с `emerge`, но и вообще при любых компиляциях. Подробнее читайте в документации [Distcc](#) и [Ccache](#).

Обновление конфигурационных файлов

После установки новых версий пакетов иногда необходимо обновить их конфигурационные файлы. В Handbook описана эта процедура с использованием утилиты `etc-update`, но существует более удобная и продвинутая утилита — [dispatch-conf](#). Эта утилита имеет возможность автоматически обновлять конфигурационные файлы, в которых произошли простые изменения (на пример, изменились только комментарии), вы также можете указать ей автоматически обновлять конфигурационные файлы, в которые не вносились изменения пользователем. При этом конфигурационные файлы помещаются в контроль системой контроля версий, чтобы вы с лёгкостью могли откатить все изменения.

Ну и [цитата](#) из `bash.org`’а напоследок:

```
<@insomnia> it only takes three commands to install Gentoo
<@insomnia> cfdisk /dev/hda && mkfs.xfs /dev/hda1 && mount /dev/hda1
/mnt/gentoo/ && chroot /mnt/gentoo/ && env-update && . /etc/profile &&
emerge sync && cd /usr/portage && scripts/bootsrap.sh && emerge system
&& emerge vim && vi /etc/fstab && emerge gentoo-dev-sources && cd
/usr/src/linux && make menuconfig && make install modules_install &&
emerge gnome mozilla-firefox openoffice && emerge grub && cp
```



```
/boot/grub/grub.conf.sample /boot/grub/grub.conf && vi /boot/grub/grub.conf  
&& grub && init 6  
<@insomnia> that's the first one
```

Ещё раз о главном - жизнь после установки

Ссылка на оригинал: <http://gentoo.blog.ru/6345603.html#comments>

С версии: 1.5

Автор: gentoo

Дата: 21.02.2008

Пост навеян аськой (да, я гентушник, и я не пользуюсь джаббером)), а точнее до боли знакомой фразой "привет! я читал твой блог! Есть вопрос!") Нет, я совсем не против новых знакомств, у самого когда-то была туча вопросов, и спасибо всем тем, кто терпеливо отвечал и помогал мне в своё время во всём разобраться ;) Но на одних подсказках далеко не уедешь, нужно (нужно!) читать хендбук (благо он на русском) и другие материалы. Хотя по себе знаю, что решения возникшей проблемы всегда хочется "здесь и сейчас", а стоит открыть хендбук и увидеть СКОЛЬКО много там букв, как уже всякое желание читать пропадает)) Напишу здесь ещё несколько советов, которые могут помочь сориентироваться новичкам, может где-то повторюсь, неважно.

Итак, предположим, вы удачно установили систему и загрузились в неё. Может быть вы уже обновили дерево портежей (по emerge --sync), и может быть пересобрали все пакеты системы (по emerge -epv world). Вобщем готовы двигаться дальше, ставить X'сы, биться с дровами за direct rendering, компилировать свой любимый DE/WM. Но стоп. Если вы смутно представляете, зачем нужны use-флаги, что такое eix, elogv, как обновлять мир, то вам сначала под кат =)

Совет: Не стоит выбирать профиль "desktop", который включает в себя несколько десятков use-флагов, предположительно подходящих для любой desktop-системы. Все необходимые флаги в любом случае пропишем постепенно в процессе установок пакетов, а добавление на начальном этапе этой кучи флагов из профиля desktop приведёт к большим зависимостям, и вы просто замучаетесь просматривать сто пакетов (и их use-флаги), потянувшихся по зависимостям за установкой всего одного-единственного необходимого вам пакета.

Совет: Перед установкой любого пакета, запускать команду emerge сначала с дополнительными ключами "pv" (или "av" - кому как больше нравится). Это даст возможность оценить предстоящий процесс установки, проверить и по необходимости включить (отключить) интересующие USE-флаги устанавливаемых пакетов.

Полезные пакеты и команды:

emerge **emerge-delta-webrsync** - теперь обновлять дерево портежей будет гораздо быстрее (за счёт экономии трафика), минус - дерево портежей после обновления будет актуально на вчерашний день включительно (изменения за сегодня вы получите только завтра).

emerge **eix** - с помощью этого пакета гораздо удобнее (и быстрее) работать с

поиском по дереву портежей.

eix-sync -Wv - этой командой удобно обновлять дерево портежей, она сделает emerge-delta-webrsync && update-eix и выведет в конце список изменений в дереве portage (что добавилось, что исчезло, что готово проапгрейдиться).

emerge -uDNvp world (или -uDNva) - предлагаю эту команду для обновления мира (если не придумаете свою, более удобную). Совсем не обязательно обновляться как только хотя бы один пакет появится в выводе этой команды. Я, например, обновляю лишь некоторые пакеты, выборочно и достаточно редко.

emerge elogv - позволит просматривать информационные сообщения всех устанавливаемых пакетов после их установки, но сначала необходимо его настроить. Как настроить - читаем в строках после установки, у меня настроено так (/etc/make.conf):

```
PORTAGE_ELOG_CLASSES="warn error log qa"
```

```
PORTAGE_ELOG_SYSTEM="save"
```

А если также добавить строку:

PORT_LOGDIR=/var/log/portage, то вся информация выводимая на экран во время компиляции пакета будет помещаться в указанный каталог (на случай, если вас не устроит elogv).

Совет: не пренебрегайте этой командой, после установки крупных вещей (X'сы, Gnome, KDE) обязательно запустите elogv и внимательно просмотрите сообщения всех пакетов, что устанавливались. В выводе некоторых могут встречаться важные инструкции, которые необходимо будет выполнить. Удалить текущую запись из журнала elogv - dd, помощь — F1.

emerge gentoolkit - сюда входят eclean, equery, eread, euse, glsa-check, а главное - revdep-rebuild. Must have.

Обновление конфигов.

При обновлении некоторых пакетов может потребоваться вручную обновить их конфиги. Если после установки какого-либо пакета вы видите на экране такое сообщение:

```
* IMPORTANT: 2 config files in '...' need updating.
```

значит некоторые конфиги ждут своего обновления. Обновить их можно либо с помощью etc-update, либо с помощью dispatch-conf. Рекомендуется использовать вторую команду, как более мощную, а при соответствующей настройке, умеющую сохранять историю изменений. В любом случае здесь рекомендуется включать мозг и внимательно изучать, какие строчки будут удалены, а какие - добавлены. Если обновляется какой-либо конфиг, где вы вносили свои корректировки, они

могут быть предложены к удалению - поэтому нужно быть внимательным (сам по-первости попался - обновил конфиги не глядя, а потом удивлялся, почему настройки, которые я терпеливо прописывал в несколько файлов вдруг заменились на дефолтные).

USE-флаги

>с флагами я не совсем разобрался... зачем их прописывать, если проги и без них поставятся?

Предположим, ставим мы с тобой kopete (IM-клиент). Ты не включаешь никакие флаги и ставишь пакет, а я включаю "irc" и "yahoo". Так вот мой копете будет поддерживать также протоколы irc и yahoo, а твой - не будет.

>т.е. прогу надо будет заново ставить?

Да, тебе придётся прописать флаги и пересобрать пакет. USE-флаги нужны для того, чтобы включать некоторый функционал в собираемом пакете.

>вот мы дали команду emerge gnome -pv

>там появились строки..красного, синего и желтого цвета...что это все означает?

Жёлтым выделены изменения в наборе флагов пакета. Флаги, больше недействительные (удалённые) для новой версии пакета будут взяты в скобки, новые (добавленные) флаги, появившиеся для новой версии, будут без скобок.

Красные - включенные флаги.

Синие - соответственно не включенные (идут по минусу).

Зелёные* - флаг был включен в системе после сборки пакета, пакет был собран без этого флага, готов к пересборке с ним.

Например:

```
[ebuild UD] app-emulation/wine-0.9.49 [0.9.57]
```

предлагает понизить версию пакета wine (я ставил тестовую 0.9.57, у которой были доступны новые use-флаги "gecko" и "samba", и которых не будет, если пересобрать пакет в версию 0.9.49).

Флаг "java" я включил в системе только что, и поэтому он подсветился во всех пакетах, где он может использоваться, зелёным цветом. USE-флаги добавляются (включаются) в системе по мере необходимости.

>ну вот я и пытаюсь узнать..как определить необходимость?

К примеру, поставил ты систему, чист твой /etc/make.conf, включено только несколько обязательных флагов (прописаны в профиле). Теперь хочешь установить Пакет_А, по-хорошему пишешь:

emerge < Пакет_А > -pv

и смотришь use-флаги всех пакетов, что будут устанавливаться. Какие флаги тебе кажутся нужными - включаешь (прописываешь). И ещё раз перезапускаешь

команду (могут появиться новые зависимости со своими флагами). Процесс повторять до полного удовлетворения =)

Если прописываешь флаги в `make.conf` - то они будут действовать глобально для всех пакетов, где встретятся. Если в `/etc/portage/package.use` в формате: категория/пакет флаг [флаг -флаг] (`app-misc/mc 7zip -X -gpm -ncurses slang`) то тогда эти флаги будут включены только для этого пакета.

Обычно в `make.conf` находится некоторое количество глобальных флагов (`X`, `kde`, `gnome`, `alsa`, etc...) и много-много строк в `package.use`. Так **постепенно** use-флаги и накапливаются. А если ты сразу пропишешь в `make.conf` "`X`, `kde`, `gnome`, `samba`" и т.д. (думая, что "вот сейчас я пропишу всё что мне надо и забуду про эти флаги"), то установка какого-нибудь `mc` потянет за собой столько пакетов по зависимостям... что ты будешь совсем не рад =) Причём, есть вероятность, что всё заикнется и выпадет ошибка.

Смотреть, что означает конкретный флаг удобно в другой консоли в `ufed`.

В `make.conf` можно прописать дефолтные опции, которые всегда будут ставится после `emerge`, к примеру, так:

`EMERGE_DEFAULT_OPTS="--ask --verbose"`

Еще стоит отметить, как более правильно распределять флаги между `make.conf` и `package.use`. У меня сейчас глобальные флаги, описывающие всю мою систему, в `make.conf`, а более локальные - относящиеся к конкретным программам - в `package.use`. Так наверно более удобно. Вообще я как-то стремлюсь сократить число флагов в `make.conf`.

Полезный пакет:

`emerge ufed` - интерактивный ncurses-based редактор use-флагов. В нём флаги можно включать (отключать) и `ufed` будет прописывать их в `/etc/make.conf`. Конечно информация о флагах здесь не настолько подробная, как хотелось бы (особенно для новичка), но всё же лучше чем ничего.

>но ведь можно сделать так:

>USE="флаг" emerge < пакет >

А при обновлении мира, флаг у этого пакета потеряется, ибо нигде не прописан, то есть при пересборке (обновлении пакета) получится

`USE="-флаг" emerge < пакет >` и пакет потеряет часть функциональности.

>понял, а как быть, если прога тянет за собой другие, а у тех свои зависимости?

Смотришь их флаги, включаешь, а они после этого потянут ещё что-нибудь... (смотришь теперь и их флаги)) пока не остановится процесс разрастания))

Если не планируется использовать `deltup`, можно прописать зеркало `yandex` в `make.conf`:

`GENTOO_MIRRORS="http://mirror.yandex.ru/gentoo-distfiles/";`

(хотя эта строка и при `дельтап`е не помешает).

Поддержание системы в чистоте и порядке =>

При установке пакета А, он прописывается в world (/var/lib/portage/world), все пакеты В С D, которые он потянул за собой по зависимостям никуда не прописываются, но они нужны пакету А и поэтому не болтаются в системе мёртвым грузом. Соответственно команда

```
"# emerge -a —depclean"
```

не сообщает ни о каких бесхозных пакетах.

Пакет А нам надоел и мы его удаляем: "# emerge -C пакет_А"

Но пакеты В С D всё ещё остаются в системе, совершенно ненужные никому. Удалить их можно лишь точно зная их имена, либо с помощью команды "# emerge -a —depclean".

Проверка целостности системы:

revdep-rebuild - команда проверит целостность зависимостей системы, и при необходимости (найденных ошибках) переустановит (доустановит) необходимые пакеты. Рекомендуется запускать всегда после выполнения команды "emerge -a —depclean".

"Masked" пакеты.

>а еще один вопрос - хочу проставить rawstudio

>а это программа под маской

>как правильно снять маску на эту программу

>в хелпе написано, что они могут быть замаскированы несколькими путями

По умолчанию используется стабильная ветвь ПО. Большое количество пакетов (версий) могут находиться в тестовой ветви. Это не означает, что эти пакеты ужасно нестабильные, отнюдь, вполне возможно, что они будут стабильно работать у вас 364 дня в году ;) Чтобы использовать такие пакеты, есть два варианта:

1. Можно перейти полностью на тестовую ветвь и иметь всегда последние версии ПО.
2. Либо можно комбинировать стабильную ветвь с некоторыми пакетами из тестовой.

В данном случае видим:

```
# eix rawstu
```

```
* media-gfx/rawstudio
```

```
Available versions: ~0.6 ~0.7
```

```
Homepage: http://rawstudio.org
```

```
Description:      a program to read and manipulate raw images from digital cameras.
```

Обе версии пакетов замаскированы тильдой, т.е. относятся к тестовой ветви ПО.

Снять такую маску можно прописав "категория/пакет ~x86" в

/etc/portage/package.keywords, например:

```
media-gfx/rawstudio ~x86
```

(вместо ~x86 может быть ~amd64, либо **)

Второй случай: [M] - пакет заблокирован.

```
# eix kde-meta
```

```
* kde-base/kde-meta
  Available versions:
    (3.5) 3.5.8 ~3.5.9
    (kde-4) [M]~4.0.1 [M]~4.0.2
```

Здесь kde-4 помечен как находящийся в тестовой ветви (~), плюс к тому же заблокирован от использования (M), т.е. разработчики не рекомендуют пока его использовать - обязательно будут ошибки. Но если очень хочется... ;) то можно разблокировать, прописав "категория/пакет" в /etc/portage/package.unmask, например:

```
kde-base/kde-meta
```

(и в данном случае нужно будет также снять тильду, как описано выше).

Установка пакетов. X'сы, DE.

Опишу свои личные соображения. Так как система будет использоваться на десктопе для повседневной работы, то обязательно понадобятся X'сы, какое-то DE, различные кодеки и прочие приложения. Можно сказать "emerge gnome" или "emerge kde" и потом долго ковыряться в use-флагах нескольких сотен пакетов - а их будет много, ибо потянутся и X'сы, а если повезёт, то захватятся и кодеки. Зато, если хватит сил, вся система будет установлена одной командой =)

Я же предпочитаю разбивать этот процесс на подпроцессы:

```
# emerge xorg-x11 - ставим только X'сы, драйвер на видеокарту, добиваемся его работы и direct rendering'a.
```

```
# emerge ffmpeg imagemagick xine-lib mplayer - в каком порядке не особо важно, главное не одновременно (а то опять увязнете в сотне-другой зависимостей). После установки этих пакетов и всех нужных зависимостей, в системе будут стоять все необходимые видео/аудио кодеки, а также будет поддержка графических форматов. Здесь же может потянуться в зависимостях samba и cups. И уже последним делом:
```

```
# emerge gnome (или kde-meta) - Гном, KDE, либо что вам там нравится.
```

Наблюдать за процессом установки можно в другой консоли, запустив:

```
tail -f /var/log/emerge.log
```

Надеюсь, эта информация помогла вам разобраться хоть в каких-то вопросах =)
Внимание! Этот пост не отменяет прочтение хендбука, так что вам теперь сюда:
<http://www.gentoo.org/doc/ru/handbook/>

[HandBook 2006.1 ru](#), [HandBook 2007.0 en](#), [Советы пользователям Gentoo](#)

Gentoo и Java

Руководство по настройке Java в Gentoo

Ссылка на оригинал: <http://www.gentoo.org/doc/en/java.xml>

+ <http://ru.wikipedia.org/wiki/Java>

Перевод + дополнения: [Бамогов Е.В а.к.а. JohnBat26](#)

С версии: 1.0.

Обновлено в: 1.4

Дата оригинала: 01.01.2008

Дата перевода: 28.01.2008



1. Что такое Java ?

Введение

Java — (произносится /'dʒɑ:və/) — объектно-ориентированный язык программирования, разрабатываемый компанией Sun Microsystems с 1991 года и официально выпущенный 23 мая 1995 года. Многие люди, говоря на русском, называют язык «Ява» по аналогии с географическим названием.

Изначально новый язык программирования назывался Oak (James Gosling) и разрабатывался для бытовой электроники, но впоследствии был переименован в Java и стал использоваться для написания апплетов, приложений и серверного программного обеспечения.

Этот объектно-ориентированный язык, разработанный для запуска на множестве платформ без необходимости повторной компиляции кода для каждой платформы. Хотя программа, написанная на Java, может быть скомпилирована как родная для конкретной платформы, наибольшую популярность Java принесла её переносимость, наряду с другими функциями, такими как сборка мусора (garbage collection).

Программы на Java могут быть транслированы в байт-код, выполняемый на виртуальной джава-машине (JVM) — программе, обрабатывающей байтовый код и передающей инструкции оборудованию, как интерпретатор, но с тем отличием, что байтовый код в отличие от текста обрабатывается значительно быстрее.

Достоинство подобного способа выполнения программ — в полной независимости байт-кода от ОС и оборудования, что позволяет выполнять Java приложения на любом устройстве, которое поддерживает виртуальную машину. Другой важной особенностью технологии **Java** является гибкая система безопасности, благодаря тому, что исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или

соединения с другим компьютером) вызывают немедленное прерывание. Это позволяет пользователям загружать программы, написанные на Java, на их компьютеры (или другие устройства, например, мобильные телефоны) из неизвестных источников, при этом не опасаясь заражения вирусами, пропажи ценной информации, и т. п.

Часто к недостаткам этого подхода относят то, что исполнение байт-кода виртуальной машиной может снижать производительность программ и алгоритмов, реализованных на языке Java. Данное утверждение справедливо для первых версий виртуальной машины Java, однако в последнее время оно практически потеряло актуальность. Этому способствовал ряд усовершенствований: применение технологии JITs (Just-In-Time compiler), позволяющей переводить байт-код в машинный код во время исполнения программы с возможностью сохранения версий класса в машинном коде, широкое использование native-кода в стандартных библиотеках, а также аппаратные средства, обеспечивающие ускоренную обработку байт-кода (например, технология [Jazelle](#), поддерживаемая некоторыми процессорами фирмы [ARM](#)).

Внутри Java существуют 3 основных семейства технологий:

- [J2EE](#) или Java EE (начиная с v1.5) — Java Enterprise Edition, для создания программного обеспечения уровня предприятия;
- [J2SE](#) или Java SE (начиная с v1.5) — Java Standard Edition, для создания пользовательских приложений, в первую очередь — для настольных систем;
- [J2ME](#), Java ME или Java Micro Edition, для использования в устройствах, ограниченных по вычислительной мощности, в том числе [мобильных телефонах](#), PDA, встроенных системах

Последней версией является [версия 1.6](#), в которой было произведено улучшение системы безопасности, улучшение поддержки XML и скриптового языка Mozilla Rhino, улучшена интеграция с рабочим столом, добавлены некоторые новые возможности в создании графических интерфейсов.

Таким образом байт-код запускается на так называемой JRE (Java Runtime Environment — Java-окружение времени выполнения), а не напрямую операционной системой. Следовательно, для выполнения Java-байткода мы нуждаемся в установленной JDK (Java Runtime Environment). JRE включает в себя основные библиотеки, независимую от платформы - виртуальную Java-машину, плагины для браузеров и многое другое. JDK (Java Development Kit — набор для Java-разработчика) добавляет средства программирования, такие как компилятор байткода и отладчик.

2. Перед тем как начать

Если у Вас уже установлена Java, тогда удостоверьтесь, что Вы следовали [Java Upgrade Guide](#).

Новая установка не требует дополнительных приготовлений.

3. Установка виртуальной машины

Варианты

Gentoo сопровождает многочисленные Runtime Environments (JREs) и Development Kits (JDKs). Среди всех предлагаемых вариантов имеются:

Поставщик	JDK	JRE
The Blackdown Java Kit	dev-java/blackdown-jdk	dev-java/blackdown-jre
Sun's Java Kit	dev-java/sun-jdk	dev-java/sun-jre-bin
The IBM Java Kit	dev-java/ibm-jdk-bin	dev-java/ibm-jre-bin
BEA WebLogic's J2SE Development Kit	dev-java/jrockit-jdk-bin	-----

Таблица 3.1: Выбор JDKs/JREs в Gentoo Linux

Пара Blackdown JRE/JDK используется по умолчанию для версии 1.4., т.к. Она свободна доступна без дополнительных регистраций. JREs и JDKs от [Sun](#), [IBM](#), и [BEA](#) конечно быстрее, но их получение требует немного больше усилий, таких как чтение и подтверждение лицензии перед загрузкой (IBM дополнительно требует Вашей регистрации).

Установка JRE/JDKs

Для установки JDK, принятой по умолчанию для Вашего профиля запустите:

emerge virtual/jdk.

Или для установки JRE, принятой по умолчанию для Вашего профиля, запустите:

emerge virtual/jre.

Недавно Sun провел повторное лицензирование своих JDK и JRE под более дружественной к Linux лицензией. В результате этого, Sun выпустил версию Java 1.5 и выше как свободно загружаемые, без дополнительных трудностей.

Примечание: JDK также включает и JRE, поэтому установив JDK, Вам не нужно устанавливать JRE

Установка других, немного ограничивающих их получение, виртуальных машин

Как уже упоминалось, некоторые из JDKs и JREs требуют, чтобы Вы совершили несколько дополнительных действий перед установкой. Просто запустите **emerge** для этих пакетов как Вы обычно это делаете. Ebuilds (сборочные файлы) в последствии сообщат Вам, куда идти и что скачать. Затем Вам следует загрузить указанные файлы в каталог: /usr/portage/distfiles. После этого, Вы можете повторно запустить команду **emerge**, которая на этот раз установит данную

JRE/JDK.

4. Настройка Вашей виртуальной машины

Введение

Gentoo способен иметь множество установленных JDKs и JREs без появления каких-либо конфликтов.

Используя средство **java-config**, Вы можете установить умолчания для всей системы (удостоверьтесь, что Вы имеете доступ под **root**). Пользователи могут также использовать **java-config**, для установки их персональных настроек по умолчанию.

Примечание: Вы также можете использовать **eselect** для изменения системной и пользовательской виртуальной машины. Смотрите **eselect java-vm help**.

Установка виртуальной машины по умолчанию

Запущенная команда **java-config --list-available-vm**s даст Вам список всех JREs и JDKs, установленных на Вашей системе. Здесь приведен пример вывода этой команды:

Листинг 4.1 Список доступных виртуальных машин.

```
# java-config --list-available-vm
The following VMs are available for generation-2:
1)  Blackdown JDK 1.4.2.03 [blackdown-jdk-1.4.2]
2)  Blackdown JRE 1.4.2.03 [blackdown-jre-1.4.2]
3)  Blackdown 32bit JRE 1.4.2.03 [emul-linux-x86-java-1.4.2]
4)  Sun 32bit JRE 1.5.0.08 [emul-linux-x86-java-1.5]
5)  Kaffe 1.1.7 [kaffe]
*)  Sun JDK 1.5.0.08 [sun-jdk-1.5]
```

Символ * показывает текущую активную виртуальную машину (system-vm или user-vm). Имя в скобках ([]) это описатель или номер для конкретной VM. Вы можете использовать описатель или номер для команды: **java-config --set-system-vm**. Здесь приведён пример по установке системной VM:

Листинг 4.2 Установка системной виртуальной машины (system-vm)

```
(По описателю (предпочтительней))
# java-config --set-system-vm blackdown-jdk-1.4
Now using blackdown-jdk-1.4 as your generation-2 system JVM
(По номеру)
# java-config --set-system-vm 6
Now using sun-jdk-1.5 as your generation-2 system JVM
```

Как обычный пользователь, Вы можете использовать: **java-config --set-user-vm**

Примечание: Вам нет необходимости изменять профиль для смены user/system VM.

Установка предпочтительной виртуальной машины

Во время установки различных Java пакетов, виртуальная машина может и будет

переключена так, как требуется.

Из-за того, что доступно большое множество виртуальных машин, мы не имеем ресурсов для тестирования и проверки работы каждого пакета их всех. Таким образом, для гарантии того, что все пакеты установятся гладко и правильно, мы определили список по умолчанию/поддерживаемых виртуальных машин на каждую архитектуру. Вы можете найти этот список в файле: **/usr/share/java-config-2/config/jdk-defaults.conf**. Когда Вы устанавливаете Java пакеты (через portage), то он обнаруживает одну из этих виртуальных машин в том файле и будет автоматически использовать ту VM, вместо системной VM. Во время установки VM переключается если это необходимо когда, например, ваша системная VM установлена на версию 1.4, но пакет, который Вы устанавливаете, требует как минимум VM версии 1.5. Во время установки он будет использовать предпочтительную версию 1.5 VM, оставляя вашу системную VM не тронутой.

Конечно, Gentoo всегда предлагает выбор, поэтому Вы можете переопределить эти установки по умолчанию в файле: **/etc/java-config-2/build/jdk.conf** и будете иметь полный контроль тем, какая VM будет использована. Несколько примеров:

Листинг 4.3 Пример файла: **/etc/java-config-2/build/jdk.conf**

(Я хочу всегда использовать sun-jdk, т.е sun-jdk-1.4 для 1.4, sun-jdk-1.5 для 1.5, и так далее)

```
*=sun-jdk
```

Листинг 4.4 Пример файла: **/etc/java-config-2/build/jdk.conf**

(Я хочу всегда использовать sun-jdk-1.5 когда возможно, за исключением случаев, когда 1.3 или 1.4 VM явно необходимы)

```
*=sun-jdk-1.5
```

Листинг 4.5 Пример файла: **/etc/java-config-2/build/jdk.conf**

Для 1.3 я предпочитаю sun-jdk 1.4, но когда она недоступна необходимо использовать ibm-jdk-bin,

Для 1.4, использовать blackdown-jdk, а для 1.5, использовать sun-jdk

```
1.3=sun-jdk-1.4 ibm-jdk-bin
```

```
1.4=blackdown-jdk
```

```
1.5=sun-jdk
```

Предупреждение: Вам нет необходимости изменять этот файл. Если вы измените эти параметры на не поддерживаемую VM, может случиться непредсказуемое. Сведения об ошибках в таком случае будут иметь гораздо меньший приоритет, чем сведения об ошибках с поддерживаемыми Vms.

5. Компиляторы

Стандартным Java-компилятором, используемым для сборки, является: **javac**. Он поставляется с каждым JDK. В дополнение к настройке виртуальной машины, используемой на этапе сборки, также возможно настроить какой компилятор использовать. Вы определяете список предпочтительных компиляторов в файле: **in /etc/java-config-2/build/compilers.conf**.

Листинг 5.1 файл **/etc/java-config-2/build/compilers.conf**

```
# If the ebuild supports it
```

```
# it will check the COMPILERS var front to back and
```

```
# use the first compiler that is installed
```

```
COMPILERS="ecj-3.1 jikes javac"
```

Некоторые компиляторы не поддерживают все возможные параметры **-target** и

-source. Как бы то ни было, каждый компилятор в этом списке проверяется, может ли он поддерживать желаемые -source/-target. **javac** будет работать во всех случаях, когда ни одного подходящего компилятора не найдено. Более детально о каждом компиляторе показано в таблице ниже:

Наименование	Описатель	Пакет	Подробности
javac	javac	N/A	Это компилятор по умолчанию, поставляется с каждой JDK
jikes	jikes	dev-java/jikes	Jikes был изначально разработан IBM. Это анекдот, но он обычно быстрее чем javac. Тем не менее он более педантичный и падает в тех случаях, в которых javac не имеет проблем. Jikes пока не поддерживает синтаксис Java 1.5
Eclipse Compiler for Java	ecj-3.1	=dev-java/eclipse-ecj-3.1*	Этот компилятор используется в Eclipse SDK. Он является достаточно функциональным и быстрым. Также поддерживает синтаксис Java 1.5

6. Установка переменной CLASSPATH по умолчанию

Предупреждение: Опции, рассмотренные в этом разделе являются устаревшими, и будут удалены в будущем. Мы очень рекомендуем не использовать их, так как Ваши Java проекты или приложения могут идеально управлять своими classpaths. Если Вы определили специфичный CLASSPATH по умолчанию, то некоторые приложения могут повести себя неожиданно, так как классы могут вдруг появиться в classpath.

java-config может использоваться для установки системного **CLASSPATH**, так же как и пользовательского **CLASSPATH**. Во-первых Вам необходимо получить список доступных java библиотек, установленных в Вашей системе, которые Вы желаете поместить в **CLASSPATH**. Далее приведен пример:

Листинг 6.1: Список классов

```
# java-config --list-available-packages
```

```
[xerces-2] The next generation of high performance, fully compliant XML parsers in the Apache Xerces family (/usr/share/xerces-2/package.env)
```

```
[junit] Simple framework to write repeatable tests (/usr/share/junit/package.env)
```

```
[bsh] BeanShell: A small embeddable Java source interpreter (/usr/share/bsh/package.env)
```

```
[bcel] The Byte Code Engineering Library: analyze, create, manipulate Java class files (/usr/share/bcel/package.env)
```

```
[log4j] A low-overhead robust logging package for Java (/usr/share/log4j/package.env)
```

```
...
```

И вновь, имена в скобках ([]) являются ID для передачи в **java-config --set-system-classpath**. Здесь пример:

Листинг 6.2 Установка CLASSPATHs.

```
# java-config --set-system-classpath log4j,xerces-2
```

Примечание: заметьте, что текущая директория (.) не является частью системного CLASSPATH, и должна быть добавлена в пользовательский профиль

Для обновления Вашего окружения, выйдите из системы, и зайдите вновь. Можно еще сделать

source /etc/profile

Для пользователей, `java-config --set-user-classpath` создаст `~/.gentoo/java-env-classpath`, который будет затем использоваться в вашем консольном профиле.

Листинг 6.3 Установка пользовательского classpath.

```
if [[ -f "${HOME}/.gentoo/java-env-classpath" ]]; then
    source "${HOME}/.gentoo/java-env-classpath"
fi
```

Если Вы действительно хотите иметь системный или пользовательский classpath, Вы можете добавить следующее в Ваш консольный профиль. Но мы советуем избегать этого

Листинг 6.4 Установка CLASSPATH

```
export CLASSPATH="${CLASSPATH}:${java-config --classpath log4j,xerces-2}"
```

7. Java-плагины для браузера

Установка плагина

Вы можете установить Java плагин для Вашего Web браузера путем запуска `emerge` для Java VM с включенным флагом `nsplugin`.

Примечание: `nsplugin` недоступен для всех архитектур. Для проверки доступности плагинов для вашей архитектуры, перед установкой Java VM выполните:

```
emerge -pv <java-vm>
```

Portage позволяет установить Вам множество версий плагинов, но только один будет использоваться браузером. Вы можете проверить список доступных плагинов, запустив:

Листинг 7.1: Просмотр доступных плагинов

```
# eselect java-nsplugin list
[1] sun-jre-bin-1.5
[2] blackdown-jre-1.4.2
```

В примере ниже, `sun-jre-bin` устанавливается в качестве плагина для браузера:

Листинг 7.2: Выбор плагина

```
# eselect java-nsplugin set sun-jre-bin-1.5
```

Проверка, что правильный плагин был выбран:

Листинг 7.3 Проверка правильности выбора плагина

```
# eselect java-nsplugin list
[1] sun-jre-bin-1.5 current
[2] blackdown-jre-1.4.2
```

Ресурс <http://java.com> также предоставляет ссылку для [проверки правильности установки плагина](#). Дополнительно, если Вы используете браузер, основанный на

Mozilla (например Firefox), тогда Вы можете проверить Ваш Java плагин, просто введя: **about:plugins** в адресной строке.

Плагины на системах с множеством библиотек.

Если Вы работаете на смешанной 64-битной и 32-битной системе (например: на AMD64), то можете использовать только 32-битные плагины.

Для использования 32-битного плагина, Вам необходимо выполнить:

emerge emul-linux-x86-java

с включенным USE-флагом **nsplugin**.

Листинг 7.4 Установка 32-битного плагина.

```
# echo "app-emulation/emul-linux-x86-java nsplugin" >> /etc/portage/package.use
# emerge emul-linux-x86-java
```

Далее проверьте, какие плагины вообще доступны:

Листинг 7.5 Просмотр доступных плагинов.

```
# eselect java-nsplugin list
```

Available 32-bit Java browser plugins

- [1] emul-linux-x86-java-1.4.2
- [2] emul-linux-x86-java-1.5

Хотя, Вы должны выбрать 32-битный браузер (такой как **mozilla-firefox-bin**) для использования с Вашим 32-битным плагином, 64-битная версия **konqueror** использует Вашу Java VM напрямую. Таким образом, возможно использовать 64-битную версию **blackdown** с этим браузером без дополнительных настроек.

Теперь выберите правильный плагин для Вашего 32-битного браузера:

Листинг 7.6 Выбор плагинов

```
# eselect java-nsplugin set 32bit emul-linux-x86-java-1.5
```

Проверка, что правильные плагины были выбраны.

Листинг 7.7 Проверка правильного плагина

```
# eselect java-nsplugin list
```

Available 32-bit Java browser plugins

- [1] emul-linux-x86-java-1.4.2
- [2] emul-linux-x86-java-1.5 current

8. USE флаги, используемые с Java

Установка USE флагов.

Для получения дополнительной информации относительно USE флагов, смотрите главу [USE флаги](#) в настольной книге Gentoo

Флаги*:

- Флаг **java** добавляет поддержку Java во множество программ.
- Флаг **nsplugin** добавляет поддержку для браузеров, основанных на Mozilla (включая Firefox). Это необходимо для просмотра апплетов в этих браузерах.
- Флаг **source** включает установку zip архива исходных кодов пакета. Это обычно используется в IDE для присоединения исходников.
- Флаг **jce** добавляет поддержку криптографического расширения Java.

* Для более удобной установки USE-флагов, используйте **ufed** (Use Flag Editor) (**emerge ufed**)

- Для Java пакетов, флаг **doc** включает сборку документации по API, используя **javadoc**.

9. Дополнительные ресурсы

Off-line ресурсы:

- `java-config man page`
- `java-config --help`

Online ресурсы

- [Java Project Page](#)
- [gentoo-java](#), [gentoo-user](#), и [gentoo-dev](#) списки рассылки
- [#gentoo](#) и [#gentoo-java](#) на [irc.freenode.net](#)
- [Wikipedia's entry for Java](#)
- If you have suggestions or questions regarding this document, please email the Gentoo Java team: java@gentoo.org

Руководство по обновлению Java в Gentoo

Ссылка на оригинал: <http://www.gentoo.org/proj/en/java/java-upgrade.xml>

Перевод: [Батогов Е.В а.к.а. JohnBat26](#)

С версии: 1.4.

Дата оригинала: 19.10.2006

Дата перевода: 29.01.2008

1. Введение

Привет и добро пожаловать. Теперь Вы можете спросить: "почему я могу захотеть обновлять Java?". Или, вероятно, вы начали процесс и получили ссылку на эту страницу после ошибки в течение выполнения emerge? Несмотря ни на что, целью этого документа является помощь Вам в процессе обновления на новую Java систему. Но чем же является эта новая Java система.

Новая Java система

Для тех, кто не знаком с новой Java системой, здесь приведен список новых возможностей:

- Способность переключать текущую VM на лету.
- Непосредственно изменять пользовательскую и системную VM, без необходимости трогать окружение шелла (т.е. не надо больше запускать `env-update && source /etc/profile` после переключения системной VM) .
- Теперь появилось понятие "сборочной VM", используемой для установки (через emerge) пакетов, и настраиваемой независимо от системной VM.
- Для каждой версии Java, т.е. 1.3, 1.4, 1.5, 1.6 и т.д., сборочная VM может быть настроена и по поставщику и по версии.
- Во время выполнения emerge, VM может переключаться на лету в соответствии с её настройками, а также из-за зависимостей пакетов. Например, некоторые пакеты не собираются с версией 1.5. В таких случаях версия 1.4 будет использована во время сборки.
- У Java пакетов, которые собираются с ant, во время компиляции будет перезаписан файл build.xml, для гарантии, что правильная версия байт-кода будет получена.

Терминология/ основные понятия

Теперь, когда Вы знаете идею того, что получаете... далее приводится несколько терминов и понятий, которые Вы можете найти полезными перед началом.

Поколение (Generation)

Это новое понятие. Смысл состоит в том, что **поколение** это множество средств и eclasses для сборки Java пакетов. В некоторое время мы начнем переходить с существующего поколения на новое. В течение этого времени, оба поколения существуют вместе в Вашей системе и в дереве портежей. Так например, Вы можете иметь системную VM для поколения 1 и системную

VM для поколения 2. Для достижения этого, пакеты которые используют поколение 1 и поколение 2 могут сосуществовать, пока происходит миграция на новое поколение.

Поколение 1 (Generation 1)

Это поколение состоит из существующих eclasses (java-pkg, java-utils, и java) и java-config-1. Поколение 1 это устаревшая, постепенно уходящая, система.

Поколение 2 (Generation 2)

Это поколение состоит из новых eclasses (java-pkg-2, java-pkg-opt-2, java-ant-2, и java-utils-2) и новой версии java-config. Это поколение, куда мы постепенно мигрируем.

Поколение 1 Системная VM

Эта VM используется для установки Java пакетов, используя eclasses из Поколения 1. Она устанавливается следующим образом:

java-config-1 --set-system-vm <choice of vm>.

Поколение 2 Системная VM

С Поколением 1, системная VM используется только для root и для пользователей, которые не имеют установленной пользовательской VM

Поколение 2 Сборочная VM

Поколение 2 вводит новый класс VM. Сборочная VM используется во время установки для сборки Java пакетов. Она изменяется, как того требует зависимости от пакетов. Например, если пакет собирается только с версией 1.4, то VM версии 1.4 и будет использоваться. Умолчания определены в файле: **/usr/share/java-config-2/config/jdk-defaults.conf**. Дополнительно сборочная VM может быть настроена в файле: **/etc/java-config-2/build/jdk.conf**.

2. Обновление java-config

Новый пакет **java-config-wrapper**, блокируется старыми версиями java-config, так что нам следует удалить его сначала:

Листинг 2.1: Удаление старого java-config

```
# emerge -C java-config
```

Теперь нам необходимо установить новую версию java-config:

Листинг 2.2: Установка нового java-config

```
# emerge -1 "=java-config-1*" "=java-config-2*"
```

3. Проверка вашего окружения

У нас теперь есть новый скрипт: **java-check-environment**. Как можно догадаться из его названия, он проверяет правильность Вашего окружения Java. Потом он советует, какие действия Вы можете предпринять для исправления найденных проблем. Таким образом, теперь запустите:

Листинг 3.1: Проверка Вашего окружения

```
# java-check-environment
```

Если **java-check-environment** обнаружит проблему, остановится и скажет Вам о

том как её исправить. Следуйте этим советам, и повторно запускайте **java-check-environment**, пока он не найдет никаких проблем.

4. Обновление ... завершено!

Если Вы дошли до этого места, значит Вы успешно перешли на новую Java систему. **Поздравляем !**

Теперь, когда Вы обновились, Вы можете захотеть взглянуть на нашу обновленную документацию:

- [Руководство пользователя \(Рус.\)](#)
- [Руководство разработчика \(Англ.\)](#)

5. Общие проблемы и вопросы

Для решения общих проблем, Java team имеет свою wiki страницу [здесь](#). Перед поиском помощи, или отправкой отчета о проблеме, пожалуйста, ознакомьтесь с этой страницей.

Gentoo и GPRS

GPRS через USB (Руководство)

Ссылка на оригинал: <http://linuxforum.ru/>

Автор: Nikoli (Linuxforum.ru)

С версии: 1.3

Появилась необходимость в GPRS соединении в качестве страховочного. Телефоны подключал по USB. Потратил пару часов на настройку, но в итоге получилось довольно простое решение. Здесь для Gentoo такого руководства не выкладывали, надеюсь моё будет полезно. Если считаете необходимым, добавьте его в FAQ.

1) Ядро должно быть собрано с:

```
Device Drivers --->
  USB support --->
    <M> USB Modem (CDC ACM) support
```

2) Должен быть установлен **ppp**

3) В **/etc/conf.d/net** нужно добавить:

```
#*****/-----Nokia with Megafon
config_ppp1=( "ppp" )
link_ppp1="ttyACM0"
pppd_ppp1=(
"debug" # Comment this off, if you don't need more info
#460800 # Serial port line speed
"crtscts" # hardware flow control for cable
"local" # Ignore carrier detect signal from the modem:
"lcp-echo-failure 0"
"lcp-echo-interval 0"
# IP addresses:
":0.0.0.0"
# - accept peers idea of our local address and set address peer as 10.6.6.6
# (any address would do, since IPCP gives 0.0.0.0 to it)
# - if you use the 10. network at home or something and pppd rejects it,
# change the address
"noipdefault" # pppd must not propose any IP address to the peer!
"ipcp-accept-local" # Accept peers idea of our local address
"defaultroute" # Add the ppp interface as default route to the IP routing table
#replacedefaultroute # New route should be our default route to Internet
"usepeerdns" # User DNS returned by server
"noauth" # The phone is not required to authenticate
# Most phone do not support compression, so turn it off.
#noipv6
#novj
#nobsdcomp
#novjccomp
#nopcomp
#noaccomp
```

```

# Username and password:
# If username and password are required by the APN, put here the username
# and put the username-password combination to the secrets file:
# /etc/ppp/pap-secrets for PAP and /etc/ppp/chap-secrets for CHAP
# authentication. See pppd man pages for details.
'user "internet" # Change this
'persist' # Persistent connection
'maxfail 99999' # Retry and retry and retry if failed...
)
chat_ppp1=(
'TIMEOUT' '45'
'ABORT' 'BUSY'
'ABORT' 'NO CARRIER'
'ABORT' 'ERROR'
" 'ATE1'
'OK' 'AT+CGDCONT=1,"IP","internet"'
'OK' 'ATD*99***1#'
'CONNECT' 'c'
)

#*****/-----Motorola with Beeline
config_ppp2=( "ppp" )
link_ppp2="ttyACM0"
pppd_ppp2=(
"debug" # Comment this off, if you don't need more info
#460800 # Serial port line speed
"crtscts" # hardware flow control for cable
"local" # Ignore carrier detect signal from the modem:
"lcp-echo-failure 0"
"lcp-echo-interval 0"
# IP addresses:
":0.0.0.0"
# - accept peers idea of our local address and set address peer as 10.6.6.6
# (any address would do, since IPCP gives 0.0.0.0 to it)
# - if you use the 10. network at home or something and pppd rejects it,
# change the address
"noipdefault" # pppd must not propose any IP address to the peer!
"ipcp-accept-local" # Accept peers idea of our local address
"defaultroute" # Add the ppp interface as default route to the IP routing table
#replacedefaultroute # New route should be our default route to Internet
"usepeerdns" # User DNS returned by server
"noauth" # The phone is not required to authenticate
# Most phone do not support compression, so turn it off.
#noipv6
#novj
#nobsdcomp
#novjccomp
#nopcomp
#noaccomp
# Username and password:
# If username and password are required by the APN, put here the username
# and put the username-password combination to the secrets file:
# /etc/ppp/pap-secrets for PAP and /etc/ppp/chap-secrets for CHAP
# authentication. See pppd man pages for details.
'user "internet.beeline.ru" # Change this
'persist' # Persistent connection
'maxfail 99999' # Retry and retry and retry if failed...
)
chat_ppp2=(
'TIMEOUT' '45'

```

```
'ABORT' 'BUSY'
'ABORT' 'NO CARRIER'
'ABORT' 'ERROR'
" 'ATE1'
'OK' 'AT+CGDCONT=1,"IP","INTERNET.MSK'"
'OK' 'ATD*99***1#'
'CONNECT' 'c'
)
```

4) Сделайте ссылки на *net.lo*

```
ln -s /etc/init.d/net.lo /etc/init.d/net.ppp1
ln -s /etc/init.d/net.lo /etc/init.d/net.ppp2
```

5) Подключите телефон с помощью USB, появится */dev/ttyACM0*

6) Выполните

```
/etc/init.d/net.ppp1 start
```

Возможно у вас появится соединение с интернетом.

Полезно почитать вывод **tail -f /var/log/messages :**

```
Oct 27 01:18:03 kompyc pppd[9874]: pppd 2.4.4 started by root, uid 0
Oct 27 01:18:03 kompyc chat[9877]: timeout set to 45 seconds
Oct 27 01:18:03 kompyc chat[9877]: abort on (BUSY)
Oct 27 01:18:03 kompyc chat[9877]: abort on (NO CARRIER)
Oct 27 01:18:03 kompyc chat[9877]: abort on (ERROR)
Oct 27 01:18:03 kompyc chat[9877]: send (ATE1^M)
Oct 27 01:18:03 kompyc chat[9877]: expect (OK)
Oct 27 01:18:03 kompyc chat[9877]: ATE1^M^M
Oct 27 01:18:03 kompyc chat[9877]: OK
Oct 27 01:18:03 kompyc chat[9877]: -- got it
Oct 27 01:18:03 kompyc chat[9877]: send (AT+CGDCONT=1,"IP","internet"^M)
Oct 27 01:18:04 kompyc chat[9877]: expect (OK)
Oct 27 01:18:04 kompyc chat[9877]: ^M
Oct 27 01:18:04 kompyc chat[9877]: AT+CGDCONT=1,"IP","internet"^M^M
Oct 27 01:18:04 kompyc chat[9877]: OK
Oct 27 01:18:04 kompyc chat[9877]: -- got it
Oct 27 01:18:04 kompyc chat[9877]: send (ATD*99***1#^M)
Oct 27 01:18:04 kompyc chat[9877]: expect (CONNECT)
Oct 27 01:18:04 kompyc chat[9877]: ^M
Oct 27 01:18:05 kompyc chat[9877]: ATD*99***1#^M^M
Oct 27 01:18:05 kompyc chat[9877]: CONNECT
Oct 27 01:18:05 kompyc chat[9877]: -- got it
Oct 27 01:18:05 kompyc chat[9877]: send (c^M)
Oct 27 01:18:05 kompyc pppd[9874]: Serial connection established.
Oct 27 01:18:05 kompyc pppd[9874]: Using interface ppp1
Oct 27 01:18:05 kompyc pppd[9874]: Connect: ppp1 <--> /dev/ttyACM0
Oct 27 01:18:06 kompyc pppd[9874]: local IP address 10.205.145.43
Oct 27 01:18:06 kompyc pppd[9874]: remote IP address 10.6.6.6
Oct 27 01:18:06 kompyc pppd[9874]: primary DNS address 10.77.48.2
Oct 27 01:18:06 kompyc pppd[9874]: secondary DNS address 10.77.48.3
```

Проверял на том, что было. С отключённым сжатием у меня заработало, у вас его возможно нужно выключить. Если у вас есть замечания по улучшению, буду рад.

По мотивам:

http://gentoo-wiki.com/Nokia_S60_and_N_series
http://gentoo-wiki.com/HOWTO_Phone_sync
http://www.opennet.ru/base/modem/gprs_linux.txt.html

Настройка GPRS в Linux

Ссылка на оригинал: <http://ru.gentoo-wiki.com>

С версии: 1.3

Введение

Это руководство поможет настроить мобильный телефон для связи с провайдером через GPRS (на примере beeline). Для того, чтобы настроить мобильный телефон для связи через GPRS, вам понадобится подключить на вашем телефоне услугу GPRS. Обзавестись Mini-USB кабелем. Поправить ручками кое-что в ядре и собрать дополнительные пакеты из набора portage.

Подготовка

Для наших задач нам понадобится изменить несколько файлов настроек. Я использовал ядро gentoo-sources-2.4.28-gentoo-r5 из portage, дозвонщик wvdial оттуда же и ppp-2.4.3-r1 тоже соответственно из portage.

Изменяем файлы:

Открываем /etc/make.conf для редактирования nano -w /etc/make.conf

Добавляем mppe-mprc (Нужно для подключения с шифрованием Microsoft MS-CHAPv2)к вашему USE Файл: /etc/make.conf

USE="mppe-mprc"

Сборка пакетов

Собираем pppd:

emerge /usr/portage/net-dialup/ppp/ppp-2.4.3-r1.ebuild

собираем wvdial:

emerge wvdial

редактируем ядро

cd /usr/src/linux && make menuconfig

(подразумевается что linux это символическая ссылка на исходники ядра)

изменяем:

Network Device Support --->

<M> (PPP point-to-point protocol) support

[] PPP multilink support (EXPEREMRNTAL) (Вы же не хотите объединять несколько телефонов в один канал)

[M] PPP filtering

<M> PPP support for async serial ports

<M> PPP support for sync tty ports

<M> PPP Deflate compression

<M> PPP BSD-Compress compression

<M> PPP over Ethernet (EXPEREMENTAL)

Примечание - т.к. у меня ядро 2.6.12-gentoo-r6, то выше названные модули я просто включил в ядро, и не возился в дальнейшем с их загрузкой. Пока проблем не было.

ISDN Subsystem --->

<*> ISDN Support

[*] Support asynchronus PPP

[*] PPP filtering for ISDN

[*] Use VJ-compression with asynchronus PPP

USB supprot --->

[*] Support for USB

--- USB Host Controller Driver

<*> ECHI HCD (USB2.0) support (EXPEREMENTAL)

(так как у меня чип материнской платы Intel то)

<*> UHCI (Intel PIIX4, VIA, ...) support

--- USB Device Class drivers

<M> USB Modem (CDC ACM) support

Примечание - для ядра 2.6.12-gentoo-r6 нужно установить:

Device Drivers ---> ISDN subsystem --->

ISDN subsystem --->

[*] CAPI2.0 Middleware support (Может и не нужно, но с ним работает)

Для C380 в данном разделе остальное у меня отключено.

И по меньшей мере для C380 дополнительно требуется установить:

Device Drivers ---> USB support ---> USB Serial Converter support --->

<*> USB Serial Converter support

[*] USB Serial Console device support

Сохраняем конфигурацию ядра и выполняем сборку. После сборки устанавливаем новое ядро

(Процесс установки ядра описан во многих источниках так что не буду повторяться)

и перезагружаем машину.

Настройка

После перезагрузки пытаемся загрузить модули

перед всеми манипуляциями проверить есть ли в /dev устройство ttyACM0, иначе скрипт и терминалка не увидят модема (телефона) по крайней мере в Fedore Core 2.0 если с ttyUSB0(1,2,3,4,5) ничего не вышло наберите находясь в /dev "mknod ttyACM0 с 160 0" (спасибо Pooh) и все должно заработать, естественно что модемом после этого будет устройство ttyACM0 (Yuri г. Barnaul)

- стандартный модуль PPP modprobe ppp_generic
- Модуль для асинхронной передачи modprobe ppp_async
- Модуль для нашего телефона modprobe acm

Проверяем загрузились модули **lsmod**

Вывод должен выглядеть примерно так --->

```
ppp_async      6848  0 (unused)
ppp_generic    22628  0 [ppp_async]
slhc           4624  0 [ppp_generic]
acm            45353  0 [xxxxxxxxxxx] (не помню точно что должно быть)
```

Производим первичную конфигурацию wvdial

cd /etc && wvdialconf wvdial.conf

открываем для редактирования /etc/wvdial.conf

nano -w wvdial.conf Файл: /etc/wvdial.conf

и изменяем следующие строки

Init2 = AT+CGDCONT=1,"IP","internet.beeline.ru"

Phone = *99***1#

Username = beeline

Password = beeline

сохраняем полученный файл. Если значение Init2 уже занато (для C380 в нем дополнительные команды инициализации), то используйте значение Init3 (Init3 =

```
AT+CGDCONT=1,"IP","internet.beeline.ru").
```

На некоторых тарифах имя пользователя и пароль пустые, тогда файл для C380 и ядра 2.6.12-r6 будет выглядеть примерно так: Файл: /etc/wvdial.conf

```
Modem = /dev/ttyACM0
Baud = 460800
Init1 = ATZ
Init2 = ATQ0 V1 E1 S0=0 &C1 &D2 +FCLASS=0
Init3 = AT+CGDCONT=1,"IP","internet.usi.ru"
ISDN = 0
Modem Type = Analog Modem
Phone = *99***1#
Username = ""
Password = ""
```

Установка соединения

запускаем wvdial

wvdial

Потираем ручки и юзаем GPRS на Linux

Автоматическая установка соединения

Если вы используете udev, можно настроить автоматическое подключение GPRS при подключении телефона к USB.

Для этого создайте такой вот файл: Файл: /etc/udev/rules.d/90-gprs-autodial.rules

```
KERNEL=="ttyACM0", RUN="/usr/bin/wvdial"
```

(немного некрасиво, но работает на ура)

..а при отключении телефона соединение оборвётся и без наших скриптов..

Примечание: А вообще, на домашних компьютерах можно использовать и графическую звонилку kppp, там и в командной строке набирать ничего не надо, да и удобнее она чем wvdial.

Заключение

Данная статья ни в коем случае не претендует на то что это является правильным и единственным способом подключения GPRS. Вся ответственность за действия ложится на ваши плечи. Я описал то что у меня работает. Допускаю что мог в чем то ошибиться так как писал это все на память так что милости прошу ваши комментарии и исправления можно отправлять по email darkman@rusgate.org

Экономим трафик, с помощью Toonel. (Linux)

Ссылка на оригинал: <http://doc.3dn.ru/publ/9-1-0-12>

Автор: Дмитрий а.к.а. doc (<http://doc.3dn.ru>)

С версии: 1.3

Бесплатный сервис toonel.net. Клиент написан на java, поэтому необходима виртуальная машина java.

Оф.сайт: www.toonel.net

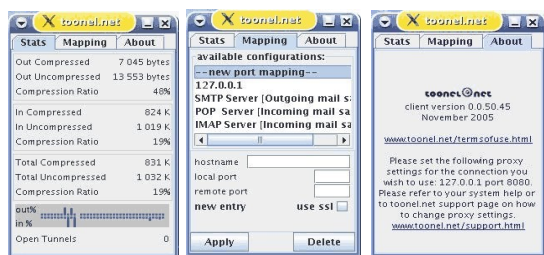
Убедитесь, что у вас установлена виртуальная машина JAVA. Посмотрите в /usr/lib/jre-xxxxx (x – номер версии), если не стоит то установите с диска дистрибутива (практический во всех дистрибутивах JAVA клиент есть), обычно называется j2re-xxxxx (x – номер версии), если Java стоит, то можно приступить к установке toonel.

Прежде чем запускать файл toonel, поменяйте у него права на чтение и запись.

Теперь выполняем простую команду:

\$ java -jar /указываем путь/toonel.jar

Появляется окно программы.



Вкладка **Stats** – показывает статистику,
вкладка **Mapping** – настройка программы,
вкладка **About** – о программе.

Теперь переходим к настройке, вкладка Mapping.

Чтобы настроить сжатие в браузере ждем на 127.0.0.1

Здесь стоит hostname: 127.0.0.1

localport: 8080

web interface port: 7999

Устанавливаем нужный процент сжатия JPEG.

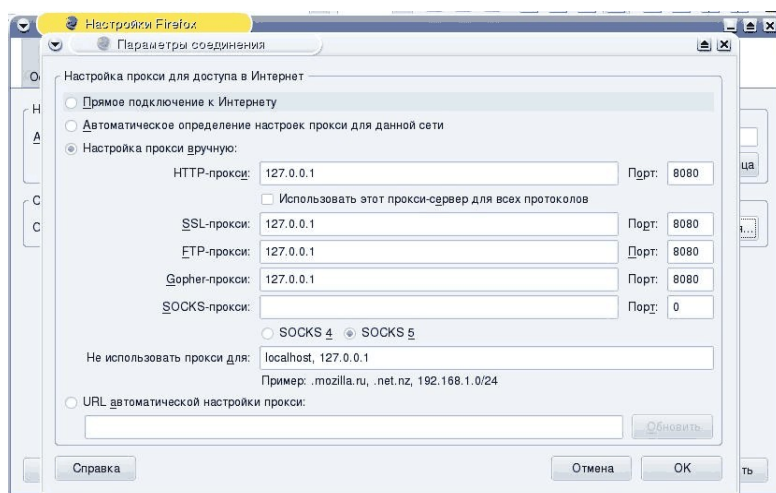
Теперь переходим в браузер, входим в Настройки, Параметры соединения.

Выбираем: Настройка прокси в ручную:

HTTP-прокси: 127.0.0.1 Порт:8080

SSL-прокси: 127.0.0.1 Порт:8080

FTP-прокси: 127.0.0.1 Порт:8080



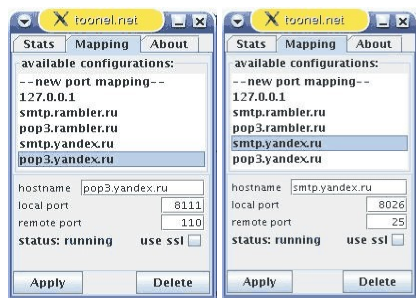
Gopher-прокси:127.0.0.1 Порт:8080

Все теперь сжатие трафика с проводника установлено.

Сжимаем трафик электронной почты.

В Toonel на вкладке Mapping жмем на – new port mapping – в hostname пишем pop3 вашего почтового сервера, например pop3.yandex.ru , в local port указываем порт для прокси сервера, например 8111 и в remote port указываем порт для входящей почты обычно 110, и жмем кнопку Apply.

Для исходящей почты тоже самое только меняем порты, ну и конечножи сервер на smtp.



Все toonel для сжатия почты готов, теперь переходим в почтовый клиент.

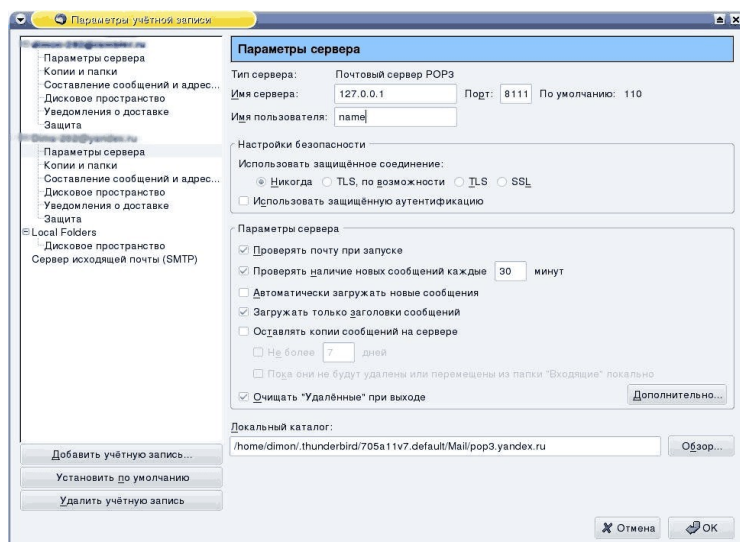
Входим в Настройки, Параметры соединения, выбираем Настройка прокси вручную:

HTTP-прокси: 127.0.0.1 Порт:8080
 SSL-прокси: 127.0.0.1 Порт:8080

После настройки прокси, входим в Параметры учетной записи и меняем Имя сервера: на 127.0.0.1 а порт 110, на порт который указали в toonel 8111.

Теперь и сжатие трафика входящей почты установлено.

Для исходящей почты Имя сервера: на 127.0.0.1 а порт 25, на порт который указали в toonel 8026.



Таким образом можно настроить сжатие любой программы которой вы пользуетесь выходя во всемирную паутину.

GPRS через KPPP

Ссылка на оригинал:

http://www.linuxcenter.ru/lib/articles/networking/kppp_gprs_hints.shtml

Автор: Валентин Сеницин

С версии: 1.3

Телефон с поддержкой GPRS сегодня есть практически у каждого. Руководств по настройке GPRS-соединения в Linux тоже предостаточно, но большая часть из них ориентирована на конкретных операторов сотовой связи или не выходит за рамки mini-HOWTO. В этой статье мы постарались дать общее представление о том, что представляет собой "GPRS в Linux" и привели примерную схему действий для настройки подключения "стандартными средствами" — через KPPP

С точки зрения повседневного использования сотовый телефон с поддержкой GPRS не слишком отличается от старого доброго модема на 14400 (да, и по скорости временами тоже). Значит, для подключения к "мобильному Интернету" можно использовать все привычные PPP-утилиты — ничего нового не потребуется. Впрочем, необходимо учесть кое-какие тонкости — о них мы и собираемся поговорить.

Как и всякое приличное "железо", GPRS-модем идентифицируется в системе файлом устройства в `/dev`. Его конкретное имя может отличаться в зависимости от дистрибутива и способа подключения: для дата-кабеля оно может называться `/dev/ttyUSB0` (необходимо подгрузить драйвер используемого конвертера USB-to-Serial, чем, по сути, и является дата-кабель. Весьма распространенный вариант — `pl2303`, если в вашем дата-кабеле используется другая микросхема — ознакомьтесь с соответствующей секцией конфигулятора ядра Linux или поищите на форумах. В современных дистрибутивах с `udev`, как правило, необходим только скомпилированный модуль — все остальное, в том числе, создание файла устройства происходит автоматически), для IrDA — `/dev/ircomm0`, для Bluetooth — `/dev/rfcomm0`. В двух последних случаях потребуется некоторая предварительная настройка, выходящая за рамки данной статьи — все подробности можно найти в соответствующих HOWTO.

Настройку GPRS-подключения, как уже упоминалось выше, можно произвести стандартными утилитами. Конкретные параметры можно найти на сайте вашего оператора сотовой связи, здесь же мы опишем стандартную схему действий на примере KPPP.

- Запустите KPPP и создайте новый модем (Configure->Modems->New). Укажите используемое имя устройства (если нужного вам имени в списке нет, создайте соответствующую символическую ссылку. Лучше делать это прямо из правил `udev`)
- Перейдите на вкладку "Modem" и нажмите кнопку "Modem commands". Введите дополнительную строку инициализации вида `AT+CGDCONT=1,"IP","APN_NAME"`. Точный вид этой строки должен быть приведен на сайте вашего оператора.
- Проверьте модем, нажав на кнопку Query Modem. Если вы не увидите сообщения об ошибке, значит, все прошло, как надо.
- Создайте новое подключение. В качестве телефона введите `*99***1#` — точный вариант, опять же, ищите на сайте оператора.
- Теперь можно задать дополнительные аргументы `pppd`. Рекомендуем добавить "debug" — будете хорошо видеть, если что-то пойдет не так. Если оператор требует отключить сжатие данных — введите "связку" `nodeflate + nobsdcomp`; если необходимо избавиться от сжатия заголовков TCP/IP — добавьте `noipv + novjccomp`.
- Отдельного упоминания заслуживает фраза: "имя пользователя и пароль не используются". Дело в том, что KPPP не позволяет оставить имя пользователя и пароль пустыми, если используется PAP-авторизация, а некоторые операторы не принимают никакие реквизиты, кроме пустой строки. Проблема решается следующим образом: откройте файл `/etc/ppp/options` (от имени `root`) и добавьте в него опцию `privgroup <ваша_группа>`. Это позволит вам передавать в командной строке привилегированные опции `pppd`, в частности, `noauth`. Ею и следует

воспользоваться.

- Закройте окно настроек. Введите имя пользователя и пароль (в случае, если они "не используются", ими может быть любой набор символов) и нажмите кнопку "Connect". Если все пройдет гладко, то через некоторое время KPPP свернется в трей — это значит, что подключение удалось. В противном случае проверьте файл журнала на предмет сообщений об ошибках.

GPRS через Bluetooth

Смотрите "[HOWTO Соединение с Internet !](#)"

HOWTO Соединение с Internet

Ссылка на оригинал: <http://ru.gentoo-wiki.com/>

С версии: 1.0

Изменено: 1.3

Настройка VPN

Это работает для ядер $\geq 2.6.15$

Проверка параметров ядра

Если ваш провайдер использует mppe-mprc проверьте следующее:

Linux Kernel Configuration: Параметры ядра

```
Device Drivers --->
  Network support --->
    <M>   PPP (point-to-point protocol) support
    <M>   PPP MPPE compression (encryption) (EXPERIMENTAL)
  Cryptographic options --->
<M>   SHA1 and RC4 algorithms
```

Теперь необходимо собрать наше ядро. Без разницы, как вы скомпилировали эти части. Вы можете вкомпилировать все в ядро или собрать модулями. В конце вы должны подгрузить модули:

- **libcrc32c**
- **ip_gre**
- **crc_ccitt**
- **ppp_async**
- и конечно **ppp_mppe_mprc**.

А затем запустите **modules-config**.

Установка пакетов

Это работает для ядер $\geq 2.6.15$

Установка очень проста:

Code: **Установка**

```
emerge sys-kernel/gentoo-dev-sources
emerge ppp
для ppp должны быть включены флаги ram,atp,dhcp (не достоверно, у меня
работает без atp и dhcp)
emerge pptpclient
```

Настройка клиента

Если все удалось, то теперь вам необходимо сконфигурировать клиента. Вот

пример конфигурации `/etc/ppp/peers/myvpn`:

```

pty                "/usr/sbin/pptp xxx.xxx.xxx.xxx --nolaunchpppd" #xxx...это
адрес сервера
name               myname
remotename         myvpn
lock
dump
noipx
#debug             #включите эту опцию, если хотите видеть отладочные
сообщения (пригодится если ничего не работает)
ipcp-accept-local
ipcp-accept-remote
lcp-echo-failure 4
lcp-echo-interval 30
asyncmap 0
crttscts
require-mppe
refuse-eap
nobsdcomp
nodeflate
proxyarp
ipparam myvpn

```

Осталось только дозвониться до провайдера: `pon myvpn` Для вывода отладочных сообщений на этой же консоли введите: `pon myvpn debug dump logfd 2`

Теперь необходимо настроить таблицы маршрутизации. Обычно это делается так: `route add default dev ppp0`

Настройка с помощью `pptpconfig`

Для графической (GTK+) настройки PPTP используйте

- `net-dialup/pptpconfig`

он пока под маской "`~x86`"

Code: установка `pptpconfig`

`ACCEPT_KEYWORDS="~x86" emerge pptpconfig`

Или более правильно в файл `/etc/portage/package.keywords` добавить следующие строки:

```
net-dialup/pptpconfig ~x86
```

```
dev-php4/php-gtk ~x86
```

А затем

Code: установка `pptpconfig`

`emerge pptpconfig`

Примечание: Для нормальной сборки необходимы флаги `USE=cli pcre session`

Наберите от рута, например в `xterm`: `pptpconfig &`

Создайте новое соединение и настройте в соответствии с параметрами указанными выше, плюс поправка на рекомендации провайдера.

Для настройки маршрутизации по умолчанию используется вкладка **Routing**, на ней используйте значение **All to tunnel**.

Выберите созданное соединение и нажмите Start. Все :)

Примечание: если возникнут проблемы с соединением на вкладке "Miscellanyous" в опциях для pppd напишите

```
noipdefault 50
```

Проверяем соединение

Теперь все готово. Можно навестить [Gentoo-Wiki](#) ;)

Проверить соединение можно командой:

```
# ifconfig ppp0
ppp0  Link encap:Point-to-Point Protocol
      inet      addr:xxx.xxx.xxx.xxx      P-t-P:xxx.xxx.xxx.xxx
Mask:xxx.xxx.xxx.xxx
      UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
      RX packets:6 errors:0 dropped:0 overruns:0 frame:0
      TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:3
      RX bytes:72 (72.0 b)  TX bytes:78 (78.0 b)
```

[Настройка VPN-соединения](#) для Gentoo

[Настройка маршрутизации](#) для VPN-соединений

Примечание (JohnBat26): Я использую perl скрипт: **pptp-command** для работы с VPN. Также можно воспользоваться **Webmin**. Взято с сайта провайдера [2Com](#):
/usr/sbin/pptp-command (появится после установки pptp-client)

Надо запустить этот скрипт из под пользователя root.

Затем выбрать:

Setup -> Manage CHAP secrets -> Add a New CHAP secret

Local Name: ВАШ_ЛОГИН

Remote Name [PPTP]: 2Com

Password: ВАШ_ПАРОЛЬ

Затем выбрать Quit

Затем выбрать:

Add a NEW PPTP Tunnel -> Other

Tunnel Name: 2Com

Server IP: 192.168.X.2 (здесь X - номер вашей сети)

route: add default ppp0

route:

Local Name: ВАШ_ЛОГИН

Remote Name [PPTP]: 2Com

Затем выбрать Quit

ВНИМАНИЕ!!!!

проверьте наличие файла /etc/ppp/options.pptp

Файл /etc/ppp/options.pptp должен выглядеть след. образом:

```
lock
noauth
nobsdcomp
nodeflate
novj
novjccomp
norpcomp
noaccomp
```

Для установки VPN соединения надо запустить скрипт /usr/sbin/pptp-command

Затем выбрать:

start -> 2Com

Для разрыва VPN соединения надо запустить скрипт /usr/sbin/pptp-command

Затем выбрать:

stop

Через pppoe

Code: Устанавливаем pppoe

emerge pppoe

В /etc/conf.d/net прописываем

```
modules=( "ifconfig" )
config_eth0=( "null" )
config_ppp0=( "ppp" )
link_ppp0="eth0"
plugins_ppp0=("pppoe")
username_ppp0='login'
password_ppp0='passwd'
pppd_ppp0=(
    "defaultroute" # Make this PPP interface the default route
    "usepeerdns"   # Use the DNS settings provided by PPP
    "lcp-echo-interval 15" # Send a LCP echo every 15 seconds
    "lcp-echo-failure 3"   # Make peer dead after 3 consecutive
)
#Если собираемся раздавать интернет в локалку с помощью маскарadingа то нужно
добавить такую строчку в IPTABLES:
#iptables -I FORWARD 1 -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-
to-pmtu
```

Code: Делаем симлинк

```
ln -s /etc/init.d/net.lo /etc/init.d/net.ppp0
```

Code: **Запускаем**

```
/etc/init.d/net.ppp0 start
```

Настройка выхода через DIAL-UP

Подключение модема

Последовательный порт

Подключение данного типа модема заключается в физическом "втыкании" в один из COM портов дата кабеля модема, включении модема в розетку и подсоединения к нему телефонной линии. Все это можно прочитать в руководстве к модему. Как известно последовательные порты в Линукс именуются `/dev/ttySx` (где x - номер порта, `/dev/ttyS0` = COM1 и т.д.) Вам нужно будет знать имя порта на который Вы "повесили свой модем"...скорее всего у вас их два или даже один, поэтому ошибиться здесь довольно сложно. Либо `/dev/ttyS0`, либо `/dev/ttyS1`. На этом подключение модема к последовательному порту можно считать законченным.

Подключение через USB

Linux Kernel Configuration: Установка пакетов

```
"USB support" >>
  <*>"USB Modem (CDC ACM) support"
```

устройство будет называться `/dev/ttyACM0`

Win модемы

Для работы Win-модема нужно установить драйвер. Не все win-модемы поддерживаются поэтому Вам остается надеяться на лучшее. За драйверами идем на <http://www.linmodems.org/> Там можно взять отличную софтинку под названием scanModem которая опросит Ваш модем и выдаст (или не выдаст) URL откуда взять драйвер. Тянем. Распаковываем. Читаем INSTALL. Ставим. Еще можно почитать эту статью: [linmodems](http://www.linmodems.org/)

Для всех типов модемов

Для удобства можно создать симлинк вида `/dev/modem` на соответствующее модему устройство:

```
ln -s /dev/ttyS0 /dev/modem
```

и в дальнейшем в настройках использовать устройство (`/dev/modem`)...хотя никто и не запрещает использовать устройство модема без симлинка `/dev/ttySx`

Проверка работоспособности, возможные грабли

После подключения модема и установки всех необходимых драйверов необходимо проверить его работоспособность. Я делаю это так: запускаю утилиту `minicom` и даю команду `AT`. Если все в порядке, я в ответ получаю `OK`, и не получаю если что-то не так. Если `OK` не получен, то самое первое, что нужно сделать - это проверить настройки `minicom`. Если модем ответил "OK", можно попробовать позвонить провайдеру:

ATDP 554554 - для пульсового набора
или
ATDT 554554 - для тонового набора

если модемы соединились и на экране появилось приглашение ввести логин или посыпался мусор в виде фигурных скобок ({}), можете себя поздравить и переходить к настройке программного обеспечения, модем у вас работает.

Настройка программного обеспечения

Включение поддержки PPP в ядре

в ядре должны быть включены опции отвечающие за `ppp`:

Linux Kernel Configuration: Установка пакетов

```
Device Drivers----->
Networking support----->
  <*>PPP (point-to-point protocol) support
  <*>PPP support for async serial ports
  <*>PPP Deflate compression
  <*>PPP BSD compression
```

Установка необходимых пакетов

во-первых нужно установить пакет который реализует поддержку протокола точка-точка (point-to-point protocol)

```
emerge ppp
```

во-вторых нужна звонилка или скрипт для автоматизирования процесса установления соединения.

Обзор звонилок и примеры скриптов смотри ниже...

Настройка соединения с помощью chat-script

Прежде чем писать скрипт, необходимо настроить `pppd`: в `/etc/ppp/pap-secrets` пропишем свои Логин/Пароль

Code: `/etc/ppp/pap-secrets`

```
login * password *
```

где `login` и `password` выданные Вашим провайдером Вам логин и пароль соответственно...

в `/etc/ppp/options` пропишем:

Code: `/etc/ppp/options`

```
nodetach
defaultroute
lock
crtstcts
```

в `/etc/resolv.conf` прописываем один или несколько ip-адресов DNS серверов (спросить у провайдера адрес его сервера)

Code: `/etc/resolv.conf`

```
nameserver xx.xx.xx.xx
```

xx.xx.xx.xx - заменить на ip-адресс DNS сервера

В каталоге `/usr/share/doc/ppp-*/scripts` лежат примеры скриптов, нам нужны:

```
ppp-on
ppp-off
ppp-on-dialer
```

подправляем в них логин, номер телефона, пароль...и вуаля...ppp-on поднимает соединение, ppp-off разрывает его, а ppp-on-dialer -вспомогательный скрипт для осуществления дозвона.

замечу только что для тонового набора используется AT-команда ATDT, а для пульсового - ATDP.

Обзор звонилки

Хорошая звонилка избавит Вас от процесса написания скриптов, вникания в опции rpprd, и сделает настройку соединения довольно тривиальной

chestnut-dialer

Замечательная звонилка которая к сожалению не поставляется в стандартных портах gentoo. ebuild можна взять [здесь](#)

Из плюсов

- Настройка тривиальна
- Умеет работать в консоли
- Умеет работать в GUI (имеет GTK+ и Qt интерфейсы)
- Умеет call back
- Нетребовательна к ресурсам
- "Местного разлива" - можно пообщаться с автором на родном языке, что для многих немаловажно.

Из минусов

- Не входит в официальное дерево портов

Настройка Call back

rpprd должен быть собран с поддержкой CALLBACK. В gentoo это делается по умолчанию, так что вам нужно просто собрать rpprd штатными средствами.

Самый простой скрипт для CALLBACK был обнаружен в /usr/doc/ppp-2.4.2/README.cbcp:

```
#!/bin/sh
pppd debug nodetach /dev/modem 115200 crtscts modem \
callback СВОЙ_НОМЕР name ЛОГИН \
connect 'chat -v "" atz OK atdtНОМЕР_ПРОВАЙДЕРА CONNECT "~--"'
sleep 1
pppd debug nodetach /dev/modem 115200 crtscts modem \
name ЛОГИН defaultroute \
connect 'chat -v RING ATA CONNECT "\c"'
```

ЛОГИН должен соответствовать записи в /etc/ppp/chap-secrets (/etc/ppp/pap-secrets). В /etc/resolv.conf нужно прописать адрес DNS-сервера (можно добавить опцию usepeerdns в /etc/ppp/options для получения списка DNS серверов от провайдера).

Также можно использовать kppp или chestnut-dialer для установки callback соединения. Настройки их достаточно тривиальны.

Самым сложным может оказаться правильно подобрать строку инициализации для модема. Рекомендуют отключить CallerID. Приведу несколько строк инициализации для примера, но лучше берите документацию от модема (часто идет только на сидироме хотя иногда бывает и в бумажном виде), смотрите что каждая из AT команд означает и экспериментируйте.

- Для модемов Lucent:

```
ats0=1s38=0s37=17-v90=0
ats0=1s38=0s37=17-v90=0+vcid=0
```

значение регистра s37 можна понижать до 14

- Для модемов GVC Vector K2D:

```
ATS0=1+VCID=0&U1
```

Я лично подобрав подходящую строку инициализации, сохранил ее прямо в профиле модема...

Настройка выхода через GPRS

Если вы подключаетесь через GPRS с помощью звонилки (kppp, chestnut-dialer), может быть важным использование опции pppd - noipdefault !!!

Требуемая аппаратура

Для соединения по GPRS потребуется следующее:

- Мобильный телефон, поддерживающий технологию GPRS
- Кабель COM или USB для подключения к компьютеру ("data-кабель") либо IrDA-интерфейс.

Подключение с помощью USB-кабеля

Убедитесь, что в конфигурационном файле Вашего ядра присутствуют следующие опции:

```
"Device Drivers" >>
```

```
[*]"Networking support" >>
[*] "Network device support"
<M> "PPP (point-to-point protocol) support"
<M> "PPP support for async serial ports"
<M> "PPP support for sync tty ports"
<M> "PPP Deflate compression"
<M> "PPP BSD-Compress compression"
```

Для data-кабелей на основе микросхем Prolific 2303 (используются для подключения большинства телефонов, в том числе Siemens, Motorola, Nokia и др.) проверьте наличие:

```
"USB support" >>
"USB Serial Converter support" >>
<M> "USB Prolific 2303 Single Port Serial Driver"
```

После компиляции ядра проверьте модуль:

```
modprobe pl2303
```

Теперь кабель представлен в системе как устройство /dev/usb/lts/0 (возможно /dev/ttyUSB0), которое может быть использовано в качестве обычного модема.

Для телефонов Motorola, которые подключаются кабелем mini-USB (например C380, C390) достаточно включить

```
"USB support" >>
<*>"USB Modem (CDC ACM) support"
```

устройство будет называться /dev/ttyACM0

Подключение с помощью COM-кабеля

Тут все просто, "втыкаем" и используем...как любой модем на последовательном порту.

Подключение через Bluetooth

Предполагается что вы уже настроили соединение телефона с компьютером через bluetooth, например по этой статье: [Работа с мобильными телефонами через bluetooth](#). Итак устройства сопряжены. Теперь нам необходимо подключить телефон как модем. Для этого мы выполним привязку bluetooth-сервиса 1(то есть Dial-Up) к устройству /dev/rfcomm0. Начнем по порядку.

- Поищем телефон

```
#hcitool scan
Scanning ...
    00:60:57:D5:3B:BE          Gtbear
```

- Подключим его к rfcomm0

```
#rfcomm bind 0 00:60:57:D5:3B:BE 1
```

Этой командой мы подключили в rfcomm0 канал 1. Теперь телефон готов к работе. Осталось настроить соединения в конфигах rppd.

- Создадим файл /etc/ppp/peers/gprs. В него запишем следующее:

```
lcp-echo-failure 0
```



```
lcp-echo-interval 0
/dev/rfcomm0
connect /etc/ppp/peers/suct-connect # Это скрипт соединения, такое имя выбрано
в честь моего оператора:
115200                                     #Южно-Уральский Сотовый Телефон :)
crtscts
local
ipcp-accept-local
noauth
usepeerdns
user suct
defaultroute
noipdefault
```

- Теперь сделаем скрипт соединения:

```
#!/bin/bash
exec chat -vS \
'' \rAT \
TIMEOUT 12 \
OK ATH \
OK ATET1 \
OK 'AT+CGDCONT=1,"IP","internet.usi.ru"' \
OK ATD*99***1# \
TIMEOUT 22 \
SAY "\nWaiting for connect...\n" \
CONNECT "" \
SAY "\nGPRS connected.Enjoy.."
```

- В принципе все готово. Для запуска соединения можно использовать команду **pppd call gprs nodetach**

Подключение через IrDA

Пакеты:

```
emerge irda-utils
```

Linux Kernel Configuration: Ядро 2.6

```
Device Drivers --->
[*] Networking support
  <*> IrDA (infrared) subsystem support --->
  <*> IrCOMM protocol
  Infrared-port device drivers --->
    <*> IrDA USB dongles
    <*> SigmaTel STIr4200 bridge (EXPERIMENTAL)
```

Когда ядро будет готово можно запускать команду:

```
irattach irda0 -s
```

Теперь если поднести телефон к инфракрасному порту он должен определиться...
Дальше можно поднимать ppp как для обычного модема

Настройка выхода через CDMA (SKYLINK)

Подключение аппаратуры

- [Модем AnyDATA ADU-E100A](#)

Настройка программного обеспечения

- Устанавливаем пакет который реализует поддержку протокола точка-точка (point-to-point protocol)

```
# emerge ppp
```

- Редактируем файл `/etc/ppp/chap-secrets`

```
Code: /etc/ppp/chap-secrets
```

```
# Secrets for authentication using CHAP
# client server secret IPaddresses
"mobile" * "internet" *
```

- Создаем файл `/etc/ppp/peers/cdma`

```
Code: /etc/ppp/peers/cdma
```

```
debug
/dev/ttyUSB0
921600
noipdefault
defaultroute
ipcp-accept-local
lcp-echo-interval 60
lcp-echo-failure 5
usepeerdns
nopcomp
noauth
noaccomp
nodetach
user "mobile"
connect "/usr/sbin/chat -s -S -V -t 5 -f /etc/ppp/cdma.chat"
```

- Создаем файл `/etc/ppp/cdma.chat`

```
Code: /etc/ppp/cdma.chat
```

```
" " # "Это не кавычки, а два знака, который на клавиатуре слева от единички
" 'ATZ' # "Это не кавычки, а два знака, который на клавиатуре слева от единички
'OK' 'ATI'
'OK' 'ATDT#777'
'CONNECT' 'ATO'
```

- Пробуем

```
# pon cdma
```

Настройка спутникового интернета в Gentoo

Автор: [tuxmaster](#)

С версии: 1.5

Введение

Спутниковый Интернет - это экономичный способ высокоскоростного подключения к интернету. Спутниковое соединение обеспечивает столь же быструю скорость, как и выделенная линия или DSL - до нескольких мегабит в секунду (это примерно в 100 раз быстрее обычного модема). Спутниковая связь очень надежна, и вопреки расхожему мнению, отлично работает при любой погоде - нужно просто использовать антенну достаточного диаметра.

Услугами спутникового Интернет можно пользоваться в любой географической точке, расположенной в обширной зоне обслуживания спутника. Если изменится место Вашего проживания, Вы всегда сможете перенести свой "спутниковый канал". Вам не придется снова тратить деньги на высокоскоростное подключение к интернету.

Как работает спутниковый Интернет?

Передача данных через спутник носит односторонний характер (асимметричный доступ): Вы можете только получать данные. Для передачи Ваших запросов на какую-либо информацию в Сети, а также исходящих от Вас данных (например, Ваших электронных писем) нужно использовать любой вид наземного соединения. Вся входящая к Вам информация будет поступать через высокоскоростной спутниковый канал.

Оборудование

Для приема информации со спутника используется комплект оборудования, состоящий из спутниковой антенны, конвертера и dvb-платы. Никаких разрешительных документов на установку принимающей антенны не требуется. В дальнейшем мы предполагаем, что Ваша спутниковая антенна уже установлена и настроена.

Подготовка Gentoo к использованию спутникового Интернета

Настройка ядра

Перейдите в каталог с исходными кодами ядра и введите команду `make menuconfig`. Этой командой Вы вызовете конфигурационное меню, использующее `ncurses`.

Листинг 1: Вызов конфигурационного меню

```
# cd /usr/src/linux
```

```
# make menuconfig
```

Перед Вами появятся несколько секций настроек. Прежде всего необходимо включить поддержку загружаемых модулей (если Вы использовали genkernel этот шаг можно пропустить).

Листинг 2: Включение поддержки модулей

```
Loadable module support --->
  [*] Enable loadable module support
  [*] Automatic kernel module loading
```

Включите поддержку Вашей DVB-платы.

Листинг 3: Включение поддержки DVB:

```
Device Drivers --->
  Multimedia devices --->
    <M> DVB for Linux
    [*] Load and attach frontends modules as needed
    [*] DVB/ATSC adapters --->
      <M> Ваша_DVB-плата
```

Теперь Вы можете приступить к компиляции ядра. Если Вы неуверенны в своих силах обратитесь к соответствующим разделам руководства.

Установка пакета утилит

Для проверки работоспособности оборудования, настройки, создания сетевого интерфейса, и т.п. Вам потребуются определенные утилиты, входящие в состав linuxtv-dvb-apps

Листинг 4: Установка пакета утилит

```
# emerge linuxtv-dvb-apps
```

Настройка параметров системы

dvb_core

По умолчанию после 5 секунд простоя DVB-плата отключается, при использовании ядер 2.6.23 и старше эта проблема решается загрузкой модуля dvb_core с параметром dvb_shutdown_timeout=0

Листинг 5: Установка параметров модуля

```
# echo 'alias dvb_core dvb_shutdown_timeout=0' >> /etc/modules.d/aliases
```

К сожалению, в ядрах 2.6.24 такой способ не работает.

/etc/channels.conf

В файле channels.conf хранятся параметры настройки dvb-карты на спутник. Для дальнейшей работы нам нужны значения частоты, скорости потока и

поляризации, узнать их можно на сайте провайдера.

Листинг 6: Создаем /etc/channels.conf

```
# nano -w /etc/channels.conf
```

Теперь заполним channels.conf параметрами транспондеров. Ниже приведен пример channels.conf для провайдера Raduga с подробными комментариями.

Листинг 7: channels.conf для абонетов Raduga

```
# Каждая строка описывает параметры отдельного транспондера в формате:
# Name:Frequency:Polarisation:diseqc:Symbolrate:VPID:APID:SID
# где
# Name – Желаемое название транспондера
# Frequency – Частота транспондера в мегагерцах (МГц, MHz)
# Polarisation – Поляризация, возможные значения
#   v-вертикальная (левая круговая), h-горизонтальная (правая круговая)
# diseqc – Номер входа diseqc. Если у Вас один конвертер выберите 0.
# Symbolrate – Символьная скорость в киლოსимволов/сек (Ksps)

# Express AM22 (53 East)
1:11096:v:0:6164:0:0:0

# Express AM1 (40 East)
2:11082:v:0:5064:0:0:0

# Intelsat 904 (60 East)
3:11595:v:0:29270:0:0:0
4:10983:v:0:3819:0:0:0

# Yamal 201 Ku (90 East)
5:11672:v:0:18200:0:0:0

# Yamal 201 C (90 East)
6:3980:h:0:38000:0:0:0

# ABS 1 (75 East)
7:12609:v:0:22000:0:0:0

# Sirius 4 (5 East)
8:12680:v:0:9404:0:0:0
```

Сетевой интерфейс

Так как при каждом включении или перезагрузке необходимо создавать dvb-сетевой интерфейс, рекомендуем создать скрипт, выполняющий эти действия автоматически.

Листинг 8: Создаем /bin/satup

```
# nano -w /bin/satup
```

Листинг 9: Вариант /bin/satup

```
#!/bin/bash

# Параметры подписки
# Ваш PID
PID=0x1234

# IP адрес подписки
IP_ADDR=000.111.222.333
```

```
# MAC адрес платы, на которую оформлена подписка
MAC_ADDR=00:11:22:33:44:55

# Поднимаем сетевой интерфейс
dvbnet -p $PID
/sbin/ifconfig dvb0_0 $IP_ADDR
/sbin/ifconfig dvb0_0 hw ether $MAC_ADDR

# Отключаем spoof-фильтрацию
echo 0 > /proc/sys/net/ipv4/conf/dvb0_0/rp_filter
```

Листинг 10: Добавление /bin/satup в автозапуск

```
# echo '/bin/satup' >> /etc/conf.d/local.start
```

Сетевой фильтр

Общий принцип, не зависящий от провайдера: необходимо разрешить входящий трафика с DVB-платы на порты ускорителя и исходящий на сервер провайдера.

Листинг 11: Пример настройки iptables для ускорителя Sprint (спутник Yamal-90 Ku)

```
# iptables -A INPUT -p udp -m udp -i dvb0_0 --dport 8093 --sport 1024:65535 -j ACCEPT
# iptables -A OUTPUT -p udp -m udp -m multiport -d 80.81.208.66 --dports 8093,8095 --sport 1024:65535 -j ACCEPT
# /etc/init.d/iptables save
```

Проверка работоспособности

Запустите в терминале szap с параметрами 'n номер_транспондера_в_файле_channels.conf', 'x' — прекратить работу после настройки и 'c путь_к_файлу_channels.conf'

Листинг 12: Проверка параметров трансподера Ku-диапазон

```
~ szap -n 5 -x -c /etc/channels.conf
reading channels from file '/etc/channels.conf'
zapping to 5 '5':
sat 0, frequency = 11671 MHz V, symbolrate 18200000, vpid = 0x0000, apid = 0x0000
using '/dev/dvb/adapter0/frontend0' and '/dev/dvb/adapter0/demux0'
status 00 | signal d689 | snr 9a83 | ber 00000117 | unc 00000000 |
status 1f | signal e625 | snr d587 | ber 00000088 | unc 00000000 |
FE_HAS_LOCK
```

Листинг 13: Проверка параметров трансподера C-диапазон

```
~ szap -n 6 -x -l C-BAND -c /etc/channels.conf
reading channels from file '/etc/channels.conf'
zapping to 6 '6':
sat 0, frequency = 3980 MHz H, symbolrate 38000000, vpid = 0x0000, apid = 0x0000
using '/dev/dvb/adapter0/frontend0' and '/dev/dvb/adapter0/demux0'
status 00 | signal a367 | snr 7543 | ber 00000129 | unc 00000000 |
status 1f | signal ea32 | snr e259 | ber 00000052 | unc 00000000 |
FE_HAS_LOCK
```

Ключевые слова: `status 1f` и `FE_HAS_LOCK` говорят о том, что настройка завершилась успешно. Теперь проверим, поддерживает ли Ваше ядро параметр `dvb_shutdown_timeout=0`

Листинг 14: Ядро поддерживает `dvb_shutdown_timeout=0`

```
~ femon
using '/dev/dvb/adapter0/frontend0'
FE: ST STV0299 DVB-S (SAT)
status 1f | signal e5df | snr d500 | ber 00000000 | unc 00000000 |
FE_HAS_LOCK
status 1f | signal e879 | snr d518 | ber 00000000 | unc 00000000 |
FE_HAS_LOCK
status 1f | signal e8a0 | snr d527 | ber 00000000 | unc 00000000 |
FE_HAS_LOCK
```

Листинг 15: Ядро не поддерживает `dvb_shutdown_timeout=0`

```
~ femon
using '/dev/dvb/adapter0/frontend0'
FE: ST STV0299 DVB-S (SAT)
status 03 | signal 9f5e | snr 0000 | ber 00008080 | unc 00000000 |
status 03 | signal 9f5e | snr 0000 | ber 00008080 | unc 00000000 |
status 03 | signal 9f5e | snr 0000 | ber 00008080 | unc 00000000 |
```

Во втором случае перед подключением к Интернету необходимо запустить `szarp` без параметра `'x'` (при этом окно терминала закрывать нельзя). Добавьте эту команду в скрипт запуска ускорителя спутникового интернета.

Листинг 16: Запуск `szarp` в фоновом режиме (пример для Ku-диапазона спутника Yamal-90)

```
~ szarp -n 5 -c /etc/channels.conf >> null &
```

Теперь, когда `dvb`-плата настроена, проверим принимает ли `dvb`-плата поток

Листинг 17: Проверка обработки `dvb`-платой транспортного потока

```
~ dvbtraffic
1039    53 p/s      9 kb/s      80 kbit
103a     7 p/s      1 kb/s      11 kbit
103b   271 p/s     49 kb/s     408 kbit
103c   102 p/s     18 kb/s     154 kbit
103d   713 p/s    130 kb/s    1073 kbit
104a  1190 p/s    218 kb/s    1790 kbit
104b   835 p/s    153 kb/s    1256 kbit
104c   635 p/s    116 kb/s     955 kbit
104d   843 p/s    154 kb/s    1267 kbit
104e  1187 p/s    217 kb/s    1786 kbit
104f   962 p/s    176 kb/s    1447 kbit
1051   167 p/s     30 kb/s     252 kbit
1057    91 p/s     16 kb/s     137 kbit
1101  4485 p/s    823 kb/s    6746 kbit
1105   195 p/s     35 kb/s     293 kbit
2000 14404 p/s   2644 kb/s   21664 kbit
-PID-FREQ-----BANDWIDTH-BANDWIDTH-
```

и функционирование сетевого интерфейса

Листинг 18: Проверка сетевого интерфейса

```
# tcpdump -c 3 -qn -i dvb0_0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on dvb0_0, link-type EN10MB (Ethernet), capture size 96 bytes
16:55:06.196218 IP 10.10.10.1 > 10.252.43.127: udp
16:55:06.196802 IP 10.10.10.1.52632 > 10.252.43.246.8093: UDP, length 884
16:55:06.196871 IP 10.10.10.1.52632 > 10.252.43.246.8093: UDP, length 884
3 packets captured
3 packets received by filter
0 packets dropped by kernel
```

Использование спутникового канала

Установите соединение с наземным провайдером, запустите ускоритель и пользуйтесь недорогим и качественным интернетом со спутника :)

Листинг 19: Пример скрипта для запуска ускорителя Sprint

```
#!/bin/bash
szap -n 5 -c /etc/channels.conf >> null &
cd /home/sprint
/home/sprint/sprint
```


Соединение нескольких офисов в одну сеть с помощью OpenVPN

Ссылка на оригинал: <http://ylsoftware.com/>

Автор: MooSE (moose@home.ylsoftware.com)

С версии: 1.4

Дата: 19.01.2008

Итак. Допустим что у некоторой фирмы есть несколько офисов в различных точках города (возможно даже земного шара - не суть важно) и нам нужно обеспечить максимально простой способ взаимодействия локальных сетей различных офисов между собой. Неплохим решением этой задачи будет объединение этих сетей посредством OpenVPN.

Итак. Уточним начальные условия:

Центральный офис (office-0):

Сервер под управлением Ubuntu Linux. Три сетевых интерфейса: eth0, eth1, eth2. Конфигурация следующая:

- eth0: внешний интерфейс, имеющий реальный ip-адрес a.b.c.d.
- eth1: первая локальная сеть: 192.168.1.1/24.
- eth2: вторая локальная сеть: 192.168.2.1/24.

Офис 1 (office-1):

Под управлением Mandriva Linux. Два интерфейса:

- eth0: внешний интерфейс, имеющий доступ к адресу a.b.c.d (каким либо образом).
- eth1: локальная сеть: 192.168.3.1/24.

Офис 2 (office-2)

Сервер полностью аналогичен серверу в первом офисе, за исключением eth1: там адрес 192.168.4.1/24.

Объединять мы будем сервера в виртуальную сеть 192.168.10.0/24. Поэтому на всех серверах должен быть настроен NAT не только для "своих" сетей, но и для сети 192.168.10.0/24.

Будем считать что всё это уже сделано. Приступаем к установке и настройке OpenVPN-сервера:

```
apt-get install openvpn
```

Создаём файл конфигурации /etc/openvpn/server.conf следующего содержания:

```
mode server
tls-server
daemon
```

```
ifconfig 192.168.10.1 255.255.255.0
```

```
port 1194
```

Соединение нескольких офисов в одну сеть с помощью OpenVPN

```
proto tcp-server
dev tap
ca /etc/openvpn/keys/ca.crt
cert /etc/openvpn/keys/office-0.crt
key /etc/openvpn/keys/office-0.key
dh /etc/openvpn/keys/dh1024.pem
client-config-dir /etc/openvpn/ccd
push "route 192.168.10.0 255.255.255.0 192.168.10.1"
keepalive 10 120
comp-lzo
persist-key
persist-tun
verb 3
log-append /var/log/openvpn.log
```

Создаём каталог, в котором будут храниться индивидуальные настройки клиентов:

```
mkdir /etc/openvpn/ccd
```

Копируем скрипты для генерации ключей и создаём ключи:

```
cp -vR /usr/share/doc/openvpn/examples/easy-rsa/2.0 /etc/openvpn/
mkdir /etc/openvpn/2.0/keys
ln -s /etc/openvpn/2.0/keys /etc/openvpn/keys
cd /etc/openvpn/2.0/keys
source ./vars
./clean-all
./build-ca
./build-dh
```

```
# Ключ для центрального офиса
./build-key office-0
```

```
# Ключ для первого офиса
./build-key office-1
```

```
# Ключ для второго офиса
./build-key office-2
```

В ходе выполнения этих команд будет задан ряд вопросов. Ответы на них в общем-то очевидны, поэтому заострять на них внимание не будем.

Далее создаём файлы /etc/openvpn/ccd/office-1 и /etc/openvpn/ccd/office-2. Содержание первого:

```
# присваиваем ip-адрес
ifconfig-push 192.168.10.101 255.255.255.0

# роутинг на сети центрального офиса
push "route 192.168.1.0 255.255.255.0 192.168.10.1"
push "route 192.168.2.0 255.255.255.0 192.168.10.1"

# роутинг на сеть второго офиса
push "route 192.168.4.0 255.255.255.0 192.168.10.102"
```

Содержание второго:

```
# присваиваем ip-адрес
ifconfig-push 192.168.10.102 255.255.255.0
```

```
# роутинг на сети центрального офиса
push "route 192.168.1.0 255.255.255.0 192.168.10.1"
push "route 192.168.2.0 255.255.255.0 192.168.10.1"

# роутинг на сеть первого офиса
push "route 192.168.3.0 255.255.255.0 192.168.10.101"
```

На этом настройка сервера завершена. Перезапускаем его:

```
/etc/init.d/openvpn restart
```

Убеждаемся что поднялся интерфейс tap0:

```
ifconfig tap0
```

Переходим к настройке офисов. Рассмотрим только один. Второй будет сделан аналогично, за исключением имён сертификатов.

Устанавливаем openvpn:

```
urpmi openvpn
mkdir /etc/openvpn/keys
```

Создаём файл конфигурации /etc/openvpn/client.conf:

```
client
dev tap
proto tcp

# адрес сервера в центрально офисе
remote a.b.c.d 1194
resolv-retry infinite
nobind
persist-key
persist-tun
comp-lzo
ns-cert-type server
ca ca.crt
cert /etc/openvpn/keys/office-1.crt
key /etc/openvpn/keys/office-1.key
log-append /var/log/openvpn.log
```

Далее нам нужно поместить файлы office-1.* и ca.crt из каталога /etc/openvpn/keys сервера в каталог /etc/openvpn/keys клиента.

После этого запускаем сервис:

```
chkconfig openvpn on
service openvpn start
```

Убеждаемся что поднялся интерфейс:

```
ifconfig tap0
```

После настройки обоих офисов можно убедиться в работе сети попробовав пинговать из одного офиса какой-нибудь компьютер, расположенный в другом офисе.

На этом всё. Более подробную информацию можно найти в документации по openvpn.

Управление питанием компьютера

Ссылка на оригинал: <http://www.gentoo.org/doc/en/power-management-guide.xml>

Автор: [Dennis Nienhüser](#)

Перевод: [Управление питанием компьютера](#)

Перевод: vah.

С версии: 1.5.

Дата оригинала: 21.01.2008

Дата перевода: 08.02.2008

Перевод статьи [Power Management Guide](#). В статье рассматриваются способы увеличения жизни ноутбука от батареи. Статья ориентирована на Gentoo, но думаю, сделав замену emerge на apt-get install/rpm -Uvh/.. и обладая общими знаниями о **runlevels** в вашей системе, можно всё написанное перенести и на другие дистрибутивы.

1. Введение

Ёмкость и время жизни батареи ноутбуков были значительно увеличены в последние годы. Тем не менее, современные процессоры потребляют намного больше энергии, чем более старые, и каждое новое поколение ноутбуков комплектуется всё более и более жадными до энергии устройствами. Поэтому Управление Питанием становится даже более важным, чем прежде. Увеличение времени автономной жизни совсем не обязательно влечёт покупку ещё одной батареи - многое можно выжать, применяя грамотные установки Управления Питанием.

Краткий обзор

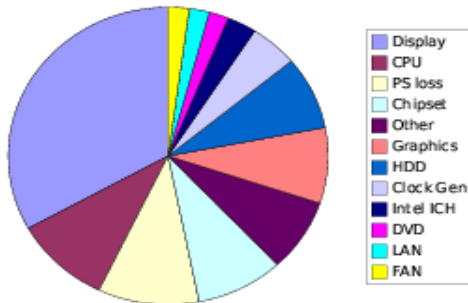
Пожалуйста, учтите, что данное руководство описывает настройку Управления Питанием только для *ноутбуков*. Хотя некоторые разделы и могут быть также применимы к серверам, другие - не только не применимы, но могут даже нанести урон. Не стоит делать что-либо без понимания того, что вы делаете.

Так как это руководство стало достаточно длинным, ниже приведён краткий обзор, который поможет вам не запутаться среди стольких букв.

В главе [Подготовка](#) говорится о некоторых общих действиях, которые необходимо выполнить перед настройкой отдельных устройств. Это включает настройку BIOS, конфигурацию ядра и некоторые изменения пользовательских настроек. Следующие три главы сфокусированы на устройствах, обычно потребляющих больше всего энергии - процессоре, дисплее и жёстком диске. Каждое из них может быть настроено независимо. Глава [Управление питанием ЦПУ](#) показывает, как изменять частоту процессора для сохранения максимального количества энергии, не теряя слишком много в производительности. Несколько способов отучить ваш жёсткий диск работать напрасно (тем самым заодно уменьшив шум) можно найти в [Управление питанием диска](#). Некоторые заметки о графических картах, беспроводных сетях и USB завершают раздел об устройствах главой

Управление питанием других устройств. Отдельная глава отведена (довольно не стабильным) Состояния спячки (sleep states). Последний раздел, Устранение неполадок, содержит список самых распространённых проблем.

Покомпонентное распределение энергии



Почти каждый элемент может находиться в различных положениях - выключен, спит, ждёт, активен - потребляя различное количество энергии. Наибольшие части отъедаются экраном, процессором, чипсетом и жёсткими дисками. Часто устройство может быть переведено в ОС-независимый режим управления питанием через BIOS, но интеллектуальные средства операционной системы, реагирующие на ситуацию, могут принести намного больше.

2. Подготовка

Перед детальным обсуждением настройки Управления Питанием для отдельных устройств, стоит убедиться, что следующие требования выполнены. После проверки установок BIOS, стоит включить некоторые опции в ядре (если коротко, то это - **ACPI**, режимы спячки и регулирование частоты ЦП). Так как сохранение энергии в большинстве случаев влечёт потери производительности, то его стоит применять только при работе от батарей. Для этого разграничения очень полезен отдельный уровень запуска (runlevel) *battery*.

BIOS

Для начала стоит взглянуть на настройки управления питанием в BIOS. Наилучшее решение - совмещать возможности BIOS и операционной системы. Но чтобы быть уверенными, что BIOS не будет конфликтовать с операционной системой, лучше пока отключить большую часть его настроек по управлению питанием. Не забудьте перепроверить настройки BIOS после конфигурации чего-либо ещё.

Настройка USE-флагов

Убедитесь, что *acpi* USE-флаг установлен в вашем */etc/make.conf*. Так же в вашей системе могут быть уместны флаги *apm*, *lm_sensors*, *nforce2*, *nvidia*, *pmu*. Подробно об этих флагах смотрите в */usr/portage/profiles/use*.desc*. Если вы забыли выставить один из этих флагов, вы можете перекомпилировать необходимые пакеты командой *emerge --deep --update --newuse world*, подробности в *man emerge*.

Конфигурация ядра

Работа над поддержкой **ACPI** (Advanced Configuration and Power Interface) в ядре всё ещё не закончена. Используя последнее ядро, вы получаете самую свежую её версию.

В портэжах (portage) содержится несколько версий ядра. Я бы рекомендовал использовать *gentoo-sources* или *suspend2-sources*. Последнее содержит патчи для программной остановки 2 (Software Suspend 2), смотрите главу о [режимах спячки](#) для более подробной информации. Во время конфигурации ядра включите хотя бы эти опции:

Листинг 2.1: Минимальные настройки ядра для Управления Питанием (приведено для ядра 2.6.23)

```
Power Management Options --->
[*] Power Management Support
[ ] Software Suspend

ACPI( Advanced Configuration and Power Interface ) Support --->
--- ACPI Support (Advanced Configuration and Power Interface) Support
    [*] Deprecated /proc/acpi files
    [*] Deprecated /proc/acpi/event support
    [ ] Sleep States
    [ ] /proc/acpi/sleep (deprecated)
    [*] AC Adapter
    [*] Battery
    <M> Button
    <M> Video
    <M> Fan
    < > Dock
    <M> Processor
    <M> Thermal Zone
    < > ASUS/Medion Laptop Extras
    < > IBM ThinkPad Laptop Extras
    < > Toshiba Laptop Extras
    (0) Disable ACPI for systems before Jan 1st this year
    [ ] Debug Statements
    [*] Power Management Timer Support
    < > ACPI0004,PNP0A05 and PNP0A06 Container Driver (EXPERIMENTAL)
    < > Smart Battery System (EXPERIMENTAL)

CPU Frequency Scaling --->
    [*] CPU Frequency scaling
    [ ] Enable CPUfreq debugging
    < > CPU frequency translation statistics
    [ ] CPU frequency translation statistics details
    Default CPUFreq governor (userspace)
    <*> 'performance' governor
    <*> 'powersave' governor
    <*> 'ondemand' cpufreq policy governor
    <*> 'conservative' cpufreq governor
    <*> CPU frequency table helpers
    <M> ACPI Processor P-States driver
    <*> CPUFreq driver for your processor
```

Примечание переводчика: поддержка `'/proc/acpi'` и `'/proc/acpi/event'` в ядре 2.6.23 была помечена не рекомендуемой (deprecated), в связи с изменением acpi-интерфейса (подробнее см. на kernel.org), но она необходима для acpid, который пока "не приспособился" к /sys интерфейсу.

Решите для себя, хотите ли вы включать программную остановку и состояния спячки (см. ниже). Если у вас ноутбук ASUS, Medion, IBM Thinkpad или Toshiba, но активируйте соответствующий пункт.

Ядро должно знать, как включить изменение частоты процессора. Так как разные типы ЦП имеют разные интерфейсы, стоит выбрать правильный драйвер для процессора. Будьте осторожны - включение, например, *Intel Pentium 4 clock modulation* на Pentium M приведёт странному поведению системы. Обратитесь к документации по ядру, если вы не знаете, что выбрать.

Скомпилируйте полученное поддерживающее ACPI ядро; убедитесь, что необходимые модули подгружаются при старте системы. Далее запустите **emerge sys-power/acpid**, чтобы получить acpi-демона. Он сообщает вам о событиях (event) таких, как переключение с питания от сети на батарею или закрытие крышки ноутбука. Убедившись, что модули запущены (если вы не скомпилировали их как часть ядра), запустите acpid командой **/etc/init.d/acpid start**. Выполните **rc-update add acpid default**, чтобы демон автоматически запускался при старте системы. Вы скоро узнаете, как использовать его.

Листинг 2.2: Установка acpid

```
# emerge sys-power/acpid # /etc/init.d/acpid start # rc-update add acpid default
```

Создание уровня запуска (runlevel) "battery"

Политики (policy) по умолчанию будут включать Управление Питанием, только когда это необходимо, т.е. при работе от батареи. Чтобы сделать переключение между питанием от сети и питанием от батарей более удобным, создайте уровень запуска *battery*, который будет содержать все скрипты запускающие и останавливающие управление питанием.

Примечание: Вы можете пропустить эту секцию, если вы не хотите создавать ещё один уровень запуска. В последующих секции предполагается, что уровень запуска *battery* существует.

Листинг 2.3: Создание уровня запуска battery

```
# cd /etc/runlevels # cp -a default battery
```

Вот и всё! Новоиспечённый уровень запуска *battery* содержит всё, что есть в *defaults*, но автоматического переключения между ними нет. Пора изменить это.

Реагирование на ACPI-события

Типичные ACPI-события - это закрытие крышки, изменение источника питания и нажатие клавиши засыпания. Для нас важно события изменения источника питания - именно из-за него будут переключаться уровни запуска. Маленький скрипт позаботится об этом.

Для начала, вам нужен скрипт, который меняет уровень запуска на *defaults* или *battery* в зависимости от источника энергии. Скрипт использует команду *on_ac_power* из *sys-power/powermgmt-base* - убедитесь, что он установлен на

вашей системе.

Листинг 2.4: Установка sys-power/powermgmt-base

emerge powermgmt-base

Теперь вы можете определить источник питания, запустив `on_ac_power && echo AC available || echo Running on batteries` в консоли. Скрипт ниже отвечает за переключение уровней. Сохраните его как `/etc/acpi/actions/pmg_switch_runlevel.sh`.

Листинг 2.5: `/etc/acpi/actions/pmg_switch_runlevel.sh`

```
#!/bin/bash

# BEGIN configuration
RUNLEVEL_AC="default"
RUNLEVEL_BATTERY="battery"
# END configuration

if [ ! -d "/etc/runlevels/${RUNLEVEL_AC}" ]
then
    logger "${0}: Runlevel ${RUNLEVEL_AC} does not exist. Aborting."
    exit 1
fi

if [ ! -d "/etc/runlevels/${RUNLEVEL_BATTERY}" ]
then
    logger "${0}: Runlevel ${RUNLEVEL_BATTERY} does not exist. Aborting."
    exit 1
fi

if on_ac_power
then
    if [ "$(cat /var/lib/init.d/softlevel)" != "${RUNLEVEL_AC}" ]
    then
        logger "Switching to ${RUNLEVEL_AC} runlevel"
        /sbin/rc ${RUNLEVEL_AC}
    fi
elif [ "$(cat /var/lib/init.d/softlevel)" != "${RUNLEVEL_BATTERY}" ]
then
    logger "Switching to ${RUNLEVEL_BATTERY} runlevel"
    /sbin/rc ${RUNLEVEL_BATTERY}
fi
```

Не забудьте сделать скрипт исполняемым, запустив **`chmod +x /etc/acpi/actions/pmg_switch_runlevel.sh`**. Последнее, что осталось сделать, - это вызов скрипта всякий раз, когда изменяется источник питания. Это можно сделать, перехватывая ACPI-события посредством *acpid*. Главное - знать, какое событие генерируется при изменении источника питания. События называются *ac_adapter* и *battery* на большинстве ноутбуков, но могут быть другими на вашем.

Листинг 2.6: Определение ACPI-событий для изменения источника питания

tail -f /var/log/messages | grep "received event"

Запустите команду выше и отключите ноутбук от питания. Затем снова

подключите его. Вы должны увидеть что-то такое:

Листинг 2.7: Определение ACPI-событий для изменение источника питания

```
[Tue Sep 20 17:39:06 2015] received event "ac_adapter AC 00000080 00000000" [Tue Sep 20 17:39:06 2015] received event "battery BAT0 00000080 00000001"
```

Интересующая часть - строки, заключенные в кавычки, после *received event*. Они должны совпадать с теми, что используются в файле ниже. Не беспокойтесь, если ваша система выдает много событий или всегда одни и те же. Пока хоть какое-нибудь событие генерируется, переключение уровней будет работать.

Листинг 2.8: `/etc/acpi/events/pmg_ac_adapter`

```
# Замените "ac_adapter" ниже на реальное событие вашем ноутбуке
# Например, ac_adapter.* будет соответствовать ac_adapter AC 00000080 00000000
event=ac_adapter.* action=/etc/acpi/actions/pmg_switch_runlevel.sh %e
```

Листинг 2.9: `/etc/acpi/events/pmg_battery`

```
# Замените "battery" ниже на реальное событие вашем ноутбуке
# Например, battery.* будет соответствовать battery AC 00000080 00000001 event=battery.*
action=/etc/acpi/actions/pmg_switch_runlevel.sh %e
```

Наконец, `acpid` должен быть перезапущен, чтобы изменения вступили в силу.

Листинг 2.10: Последний штрих

```
# /etc/init.d/acpid restart
```

Вы можете проверить работоспособность скриптов, отсоединив кабель питания и вставив его вновь, - в `syslog` должны появляться сообщения "Switching to AC mode" или "Switching to battery mode". Обратитесь к [устранению неполадок](#), если скрипт некорректно распознаёт источник питания.

Из-за природы механизма событий, ваш ноутбук будет загружаться в уровень запуска *default* не зависимо от источника питания. Это просто замечательно, если мы подключены к сети, но если нет, то нам хотелось бы загружаться в уровень запуска *battery*. Одно из решений - это добавить ещё один пункт в менеджер загрузки, с параметром *softlevel=battery*, но не сложно просто забыть его выбрать. Лучший путь - поставить ACPI-событие в конец процесса загрузки и позволить скрипту `pmg_switch_runlevel.sh` решать, какое изменение уровня необходимо. Откройте `/etc/conf.d/local.start` в *вашем любимом редакторе* и добавьте следующие строки:

Листинг 2.11: Изменение уровня запуска во время загрузки

```
# Эмулируем аспри событие, чтобы переключить уровень запуска, если работаем от батареи
/etc/acpi/actions/pmg_switch_runlevel.sh "battery/battery"
```

Подготовив систему таким образом, вы можете активировать политики управления питанием для отдельных устройств.

3. Управление питанием ЦПУ

Мобильные процессоры могут работать на нескольких частотах, а некоторые позволяют даже менять напряжение. Максимальная производительность нужна далеко не всегда, и меняя её можно сохранить немало энергии, причём зачастую без какого-либо понижения производительности

Техническая часть

Частота процессора связана с некоторыми техническими терминами, который могут быть вам незнакомы, поэтому ниже будет дано краткое введение.

Для начала, ядро должно "уметь" менять частоту процессора. *Драйвер процессора CPUfreq* (CPUfreq processor driver) знает как сделать это на вашем ЦП, поэтому важно указать его правильно в вашем ядре. Скорее всего, вы это уже сделали выше. Также ядро должно знать, какую частоту устанавливать. Это достигается посредством *политик* (policy), которые состоят из *политики CPUfreq* (CPUfreq policy) и *властелина* (governor). Первое - это пара из двух чисел, обозначающих максимальную и минимальную допустимую частоту ЦП, а второе отвечает за то, какую конкретно частоту выбрать. Так, властелин сохранения энергии (powersave governor) всегда выбирает минимальную допустимую частоту, а властелин максимальной производительностью (performance governor), наоборот, максимальную; властелин пользователя (userspace governor) использует необходимую пользователю (или его приложениям) частоту, читая её из /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed.

Всё это не очень похоже на динамическое изменение частоты (и действительно не является таковым). Однако динамика может быть получена несколькими способами. Например, "требовательный" властелин (ondemand governor) принимает решение, основываясь на текущей загрузке ЦП. Это же может быть реализовано посредством различных пользовательских инструментов таких, как *cpudn*, *cpufreqd*, *powertowd* и многих других. События ACPI можно использовать для включения или выключения динамического изменения частоты в зависимости от источника питания.

Установка частоты вручную

Уменьшение скорости и напряжение на ЦП даёт двойной выигрыш: с одной стороны, потребляется меньше энергии, а с другой - ваша система греется не так сильно, как на полной мощности. Главный недостаток - очевидная потеря производительности. Уменьшение скорости процессора - обмен производительности на сохранение энергии.

Примечание: Не все ноутбуки поддерживают изменение частоты. Если вы не уверены, посмотрите список поддерживаемых процессоров в секции [Устранение неполадок](#).

Самое время проверить, работает ли изменение частоты процессора. Давайте установим ещё один очень полезный при отладке инструмент: *sys-power/cpufrequtils*

Листинг 3.1: Проверка частоты ЦП

```
# emerge cpufrequtils # cpufreq-info
```

Вывод будет примерно таким:

Листинг 3.2: Пример вывода cpufreq-info

```
cpufrequtils 0.3: cpufreq-info (C) Dominik Brodowski 2004
Report errors and bugs to linux@brodo.de, please.
analyzing CPU 0:
  driver: centrino
  CPUs which need to switch frequency at the same time: 0
  hardware limits: 600 MHz - 1.40 GHz
  available frequency steps: 600 MHz, 800 MHz, 1000 MHz, 1.20 GHz, 1.40 GHz
```

Листинг 3.2: Пример вывода `cpufreq-info`

```

available cpufreq governors: conservative, ondemand, powersave, userspace,
performance
current policy: frequency should be within 924 MHz and 1.40 GHz.
The governor "performance" may decide which speed to use
within this range.

current CPU frequency is 1.40 GHz. governor "conservative" may decide which
speed to use within this range.
current CPU frequency is 1.60 GHz.

```

Теперь, поиграйтесь с `cpufreq-set`, чтобы убедиться в работоспособности переключение частоты. Например, выполните `cpufreq-set -g ondemand`, чтобы активировать "требовательного" властелина, и проверьте изменение с помощью `cpufreq-info`. Если не работает как надо, вы можете попробовать найти причину в [Устранении неполадок](#) в конце этого руководства.

Автоматическая настройка частоты

Сказанное, конечно, прекрасно, но не очень применимо в реальной жизни. Лучше поручить вашей системе автоматически подбирать подходящую частоту. Есть много способов сделать это. Следующая таблица даёт общее представление о них и поможет вам выбрать необходимый. Их можно не совсем точно разделить на три типа: **ядро** - для методов нуждающихся только в поддержке ядра, **демон** - для запускающихся в фоне программ и **графический** - для программ, которые предоставляют графический интерфейс для простой настройки.

Имя	Тип	Причины для переключения (Switch decision)	Используемые властелины ядра (Kernel governors)	Предоставляемые режимы (Further governors)	Комментарии
требовательный властелин ('ondemand' governor)	Ядро	Загрузка ЦП	-	-	Выставляет максимальную частоту при загрузке ЦП и медленно шагает вниз во время простоя ЦП. Дальнейшая настройка через /sys/devices/system/cpu/cpu0/cpufreq/ondemand/. По-прежнему требует пользовательских программ (скриптов), если переключение властелинов или подобное необходимо.
консервативный властелин ('conservative' governor)	Ядро	Загрузка ЦП	-	-	В отличие от требовательного властелина, консервативный не прыгает до максимальной частоты во время высокой загрузки ЦП, а увеличивает частоту постепенно. Дальнейшая настройка через /sys/devices/system/cpu/cpu0/cpufreq/conservative/. По-прежнему требует пользовательских программ (скриптов), если

Имя	Тип	Причины для переключения (Switch decision)	Используемые властелины ядра(Kernel governors)	Предоставляемые режимы (Further governors)	Комментарии
					переключение властелинов или подобное необходимо.
cpudyn	Демон	Загрузка ЦП	Производительность, сохранение энергии	Динамические	Также поддерживает disk standby. Однако стоит заметить, что laptop-mode в большинстве случаев работает лучше.
cpufreqd	Демон	Состояние батареи, загрузка ЦП, температура, запущенные программы и др.	Все возможные	Нет	Тонкая (и местами сложная) установка. Расширяем посредством плагинов, таких как отслеживание показаний сенсоров (sensor monitoring) (lm_sensors). Может управлять памятью и ядром некоторых построенных на NVidia графических карт. Cpuufreqd не поддерживает SMP, и при желании им можно управлять вручную во время выполнения.
powernowd	Демон	Загрузка процессора	Нет	Пассивный, синусоидальный (sine), агрессивный	Поддерживает SMP.
ncpufreqd	Демон	Температура	Нет	Производительность, сохранение энергии	Переключает властелинов производительности и сохранения энергии. Очень полезен на ноутбуках, страдающих перегревом.
speedfreq	Демон	Загрузка ЦП	Нет	Динамический, сохранения энергия, производительность, фиксированной скорости	Легко настраивается с помощью приятного клиент-серверного интерфейса. Требуется ядро 2.6. Не поддерживается, сломан и поэтому удален из портежей. Пожалуйста, перейдите на cpubfreqd, если вы всё ещё используете эту программу.
gtk-cpuspeedy	Графический	Нет	Нет	Нет	Gnome-приложение, графический инструмент для ручной установки частоты процессора. Не предоставляет никакой автоматизации.
klaptopdaemon	Графический	Состояние батареи	Все возможные	Нет	Только для КДЕ, властелин требований для динамического изменения частоты.

Хотя изменение частоты в зависимости от нагрузки выглядит просто на первый взгляд, это не такая уж и тривиальная задача. Плохой алгоритм может привести к постоянному переключению между двумя частотами или бессмысленному

перерасходу энергии из-за неоправданно большой частоты.

Что выбрать? Если вы не можете решить, попробуйте *cpufreqd*.

Листинг 3.3: Установка *cpufreqd*

emerge cpufreqd

cpufreqd настраивается редактированием */etc/cpufreqd.conf*. Конфиг, предоставляемый по умолчанию, может вас несколько смутить. Рекомендуется заменить его на конфигурационный файл, который создал бывший разработчик Gentoo - Henrik Brix Andersen. Учтите, что вам необходим *cpufreqd-2.0.0* или более новая версия. Ранние версии используют другой синтаксис конфигурационного файла.

Листинг 3.4: */etc/cpufreqd.conf* (*cpufreqd-2.0.0* или более новый)

```
[General]
pidfile=/var/run/cpufreqd.pid
poll_interval=3
enable_plugins=acpi_ac, acpi_battery
enable_remote=1
remote_group=wheel
verbosity=5
[/General]

[Profile]
name=ondemand
minfreq=0%
maxfreq=100%
policy=ondemand
[/Profile]

[Profile]
name=conservative
minfreq=0%
maxfreq=100%
policy=conservative
[/Profile]

[Profile]
name=powersave
minfreq=0%
maxfreq=100%
policy=powersave
[/Profile]

[Profile]
name=performance
minfreq=0%
maxfreq=100%
policy=performance
[/Profile]

[Rule]
name=battery
ac=off
profile=conservative
[/Rule]

[Rule]
```

Листинг 3.4: /etc/cpufreqd.conf (cpufreqd-2.0.0 или более новый)

```
name=battery_low
ac=off
battery_interval=0-10
profile=powersave
[/Rule]

[Rule]
name=ac
ac=on
profile=ondemand
[/Rule]
```

Теперь вы можете запустить демон cpufreqd. Добавьте его в уровни запуска *default* и *battery*.

Листинг 3.5: Запуск cpufreqd

```
# rc-update add cpufreqd default battery # rc
```

Иногда желательно установить политику отличную от той, что выбрал демон. Например, если заряд батареи низок, но вы знаете, что скоро появится питание от сети. В этом случае вы можете перевести cpufreqd в ручной режим с помощью команды *cpufreqd-set manual* и далее выбрать одну из сконфигурированных политик (они перечислены в *cpufreqd-get*). Выйти из ручного режима вы можете командой *cpufreqd-set dynamic*

Предупреждение: Не запускайте одновременно более одной программы, перечисленной выше. Это может привести к конфликтам таких, как постоянное переключение между двумя частотами.

Проверка результата

Осталось только проверить, что ваши новые политики работают правильно. Простой способ сделать это - следить за скоростью ЦП во время работы:

Листинг 3.6: Отслеживание скорости ЦП

```
# watch grep \"cpu MHz\" /proc/cpuinfo
```

Если /proc/cpuinfo не обновляется (см. [Устранение неполадок](#)), попробуйте *sys-apps/x86info*:

Листинг 3.7: Альтернативное отслеживание скорости ЦП

```
# watch x86info -mhz
```

В зависимости от ваших настроек, частота должна возрасти при нагрузке, падать в спокойном состоянии, или оставаться на том же уровне. Если вы используете *cpufreqd* и уровень подробности (verbosity) равен 5 или больше в *cpufreqd.conf*, то вы можете найти дополнительную информацию о происходящем в *syslog*.

4. LCD Power Management

Как вы видели в графике [распределения энергии](#), LCD экран потребляет наибольшую часть энергии (по крайней мере для мобильных ПК). Таким образом, важно не только выключать экран, когда он не нужен, но и уменьшать его яркость, если это возможно. Большинство ноутбуков предоставляют возможность контролировать яркость экрана

Настройки засыпания (standby)

В первую очередь необходимо проверить настройки засыпания(standby), остановки(suspend), выключения дисплея. Так как это сильно зависит от вашего оконного менеджера, то вам предстоит делать это самим. Только два замечания: погасить терминал можно командой *setterm -blank <количество минут>*, *setterm -powersave on* и *setterm -powerdown <количество минут>*. Для X.org, необходимо исправить в /etc/X11/xorg.conf следующее (или в /etc/X11/XF86Config для XFree86):

Листинг 4.1: Настройка засыпания дисплея в X.org и XFree86

```
Section "ServerLayout"
    Identifier [...]
    [...]
    Option "BlankTime" "5" # Blank the screen after 5 minutes (Fake)
    Option "StandbyTime" "10" # Turn off screen after 10 minutes (DPMS)
    Option "SuspendTime" "20" # Full suspend after 20 minutes
    Option "OffTime" "30" # Turn off after half an hour
    [...]
EndSection

[...]

Section "Monitor"
    Identifier [...]
    Option "DPMS" "true"
    [...]
EndSection
```

Уменьшение подсветки

Если вы можете управлять яркостью с помощью программы, напишите маленький скрипт, который уменьшает подсветку и поместите его в ваш уровень запуска *battery*. Следующий скрипт должен работать на большинстве ноутбуков IBM Thinkpad и Toshiba. Для ноутбуков IBM необходимо включить соответствующий пункт в ядре, а владельцы Toshiba могут поставить *sys-power/acpitool* и пропустить настройку *ibm_acpi*, описанную ниже.

Предупреждение: Поддержка изменения яркости является экспериментальной для *ibm-acpi*, так как работает с оборудованием напрямую и может серьезно поредить его. Обратитесь к [сайту ibm-acpi](#) за подробностями.

Для поддержки изменения уровня яркости, модуль *ibm_acpi* должен быть запущен с параметром *experimental*.

Листинг 4.2: Настройка засыпания дисплея в X.org и XFree86

```
(Прочтите предупреждение выше, прежде чем делать это)
# echo "options ibm_acpi experimental=1" >> /etc/modules.d/ibm_acpi
# /sbin/update-modules
# echo ibm_acpi >> /etc/modules.autoload.d/kernel-2.6
# modprobe ibm_acpi
```

Это должно выполняться без ошибок, а также должен появиться файл /proc/acpi/ibm/brightness. Скрипт инициализации будет принимать решение об уровне подсветки в зависимости от источника питания.

Листинг 4.3: /etc/conf.d/lcd-brightness

```
# См. допустимые значения в /proc/acpi/ibm/brightness
# Пожалуйста, прочтите /usr/src/linux/Documentation/ibm-acpi.txt
# Уровень яркости при питании от сети. По умолчанию - 7. BRIGHTNESS_AC=7
# Уровень яркости при питании от батареи. По умолчанию - 4. BRIGHTNESS_BATTERY=4
```

Листинг 4.4: /etc/init.d/lcd-brightness

```
#!/sbin/runscript

set_brightness() {
    if on_ac_power
    then
        LEVEL=${BRIGHTNESS_AC:-7}
    else
        LEVEL=${BRIGHTNESS_BATTERY:-4}
    fi

    if [ -f /proc/acpi/ibm/brightness ]
    then
        ebegin "Setting LCD brightness"
        echo "level ${LEVEL}" > /proc/acpi/ibm/brightness
        eend $?
    elif [[ -e /usr/bin/acpitool && -n $(acpitool -T | grep "LCD brightness")
    ]]
    then
        ebegin "Setting LCD brightness"
        acpitool -l $LEVEL >/dev/null || ewarn "Unable to set lcd brightness"
        eend $?
    else
        ewarn "Setting LCD brightness is not supported."
        ewarn "For IBM Thinkpads, check that ibm_acpi is loaded into the
kernel"
        ewarn "For Toshiba laptops, you've got to install sys-power/acpitool"
    fi
}

start() {
    set_brightness
}

stop () {
    set_brightness
}
```

Далее, убедитесь, что уровень яркости меняется автоматически, добавив скрипт в уровень запуска battery.

Листинг 4.5: Включаем автоматическое изменение яркости

```
# chmod +x /etc/init.d/lcd-brightness
# rc-update add lcd-brightness battery
# rc
```

Управление питанием диска

Жёсткий диск потребляет меньше энергии во время спящего режима (sleep mode). Поэтому разумно активировать возможности сохранения энергии, когда жёсткий диск не используется некоторое время. Есть два различных способа сделать это. Первый, (laptop-mode) будет сберегать больше всего энергии за счёт

предотвращения или, по крайней мере, задержки записи на диск. Обратной стороной медали является повышенный риск потери данных при сбое питания или аварии ядра; чтобы уменьшить масштабы возможной аварии, убедитесь, что не запущенно часто пишущих на диск процессов. В качестве альтернативного, второго способа вы можете включить сохранение энергии на диске с помощью *hdparm*.

Увеличение времени покоя (idle time) - laptop-mode

Последние 2.6-ядра включают одноимённый *laptop-mode*. Когда он активирован, "грязные" буферы (dirty buffers) пишутся на диск при вызовах чтения (read calls) или после 10 минут (вместо 30 секунд). Это минимизирует время, которое тратится на раскручивание диска.

Листинг 5.1: Автоматический старт laptop-mode

```
# emerge laptop-mode-tools
```

Настройки пакета *laptop-mode-tools* хранятся в файле */etc/laptop-mode/laptop-mode.conf*. Отредактируйте его на свой вкус - он отлично документирован. Выполните *rc-update add laptop_mode battery*, чтобы он запускался автоматически.

Последние версии (1.11 и позже) этого пакета включают полезный инструмент *lm-profiler*. Он следит за использованием диска и запущенными сетевыми сервисами и предлагает отключить ненужные. Вы можете отключить их как используя встроенную в *laptop-mode-tools* поддержку уровней запуска (которая будет переопределена */sbin/rc*), так и через созданные вами уровни запуска *default/battery* (рекомендуется).

Листинг 5.2: Пример вывода lm-profiler

```
# lm-profiler
Profiling session started.
Time remaining: 600 seconds
[4296896.602000] amarokapp
Time remaining: 599 seconds
[4296897.714000] sort
[4296897.970000] mv
Time remaining: 598 seconds
Time remaining: 597 seconds
[4296900.482000] reiserfs/0
```

После наблюдения за вашей системой в течении 10 минут, *lm-profiler* выведет список сервисов, которые могли вызвать обращения к диску за этот промежуток времени.

Листинг 5.3: *lm-profiler* предлагает отключить некоторые сервисы

```
Program:      "atd"
Reason:       standard recommendation (program may not be running)
Init script:  /etc/init.d/atd (GUESSED)
```

```
Do you want to disable this service in battery mode? [y/N]: n
```

Чтобы отключить *atd*, как предлагается в примере выше, выполните *rc-update del atd battery*. Будьте осторожны, не отключите жизненноважные сервисы - *lm-profiler* частенько выдает не совсем правильные данные. Не отключайте что-либо, если вы полностью не уверены в ненужности сервиса.

Ограничение доступа на запись

Если вы не хотите использовать `laptop-mode`, вы должны отключить часто пишущие на диск сервисы вручную - `syslogd` кандидат на выбывание, например. Возможно, вы не хотите выключать его полностью, тогда можно подправить конфиг так, чтобы писались только "необходимые вещи". `Cups` пишет на диск лишь время о времени, поэтому можно выключить его и включать вручную, когда необходимо.

Листинг 5.4: Выключение `cups` из уровня запуска `battery`

```
# rc-update del cupsd battery
```

Вы также можете использовать `lm-profiler` из `laptop-mode-tools` (см. выше), чтобы найти неудобные сервисы. Если вы истребили их всех, то можно переходить к настройке `hdparm`.

hdparm

Вторая возможность - использование `hdparm`. Пропустите этот шаг, если вы используете `laptop-mode`. Иначе, откройте `/etc/conf.d/hdparm` и добавьте следующие строки для ваших жёстких дисков. Этот пример предполагает, что ваш диск именуется `hda`:

Листинг 5.5: Использование `/etc/conf.d/hdparm` для засыпания диска

```
hda_args="-q -S12"
```

Это включит управление питанием для вашего жесткого диска. Если вы когда-нибудь решите отключить управление питанием, то просто измените значение в `/etc/conf.d/hdparm` на `-q -S0` или выполните `hdparm -q -S0 /dev/hda`.

Обратитесь к `man hdparm` за другими опциями. Хотя вы всегда можете запускать `hdparm` вручную командой `/etc/init.d/hdparm start`, намного проще автоматизировать её запуск и остановку. Чтобы сделать это, добавьте `hdparm` в уровень запуска `battery`.

Листинг 5.6: Автоматизация засыпания диска

```
#rc-update add hdparm battery
```

Внимание: Будьте осторожны с настройками засыпания/остановки (`sleep/spin down`) вашего жёсткого диска. Установление слишком низких значений может быстро "износить" ваш диск, что приведёт к потере гарантии. (Прим. пер.: вспомните [историю про Ubuntu](#))

Другие "типы и трюки"

Ещё один способ достижения светлой цели снижения энергопотребления состоит в отключении свопа (`swap`) в режиме питания от батареи. Перед тем, как писать переключатель `swapon/swaroff`, убедитесь, что у вас достаточно оперативной памяти и своп существенно не используется, иначе вы рискуете приобрести большие проблемы.

Если вы не хотите использовать `laptop-mode`, то можно уменьшить обращение к диску, монтируя некоторые директории как `tmpfs` - запись на такие разделы хранится не на диске, а в ОЗУ, и поэтому пропадает после отмонтирования. Зачастую удобно примонтировать `/tmp` таким образом, так как она в любом случае очищается при каждой перезагрузке. Правда, стоит убедиться, что у вас достаточно памяти и что нет программ (таких, как менеджеры зачек и

архиваторы), которым требуется экстремально много места в /tmp. Чтобы сделать это, включите поддержку tmpfs в ядре и добавьте подобную следующей строку в /etc/fstab:

Листинг 5.7: Редактирование /etc/fstab, чтобы сделать /tmp более отрешённым от диска (volatile)

```
none /tmp tmpfs size=32m 0 0
```

Предупреждение: Обратите внимание на параметр (32 m), и подстройте его под вашу систему. Если вы не уверены, не пробуйте это вообще - это может сильно уменьшить производительность. Если вы хотите подмонтировать /var/log таким же образом, убедитесь, что файлы сохраняются на диск перед размонтированием. Они необходимы. Не пытайтесь примонтировать /var/tmp таким образом - он используется Portage при сборке пакетов.

6. Управление питанием других устройств

Графические карты

Если у вас стоит карточка ATI с поддержкой PowerPlay (динамическое изменение частоты графического процессора), вы можете активировать эту возможность в X.org. Откройте /etc/X11/xorg.conf и добавьте (или включите) опцию *DynamicClocks* в секции Устройство (device). Учтите, что это может привести к падению на некоторых системах.

Листинг 6.1: Включение поддержки ATI PowerPlay в X.org

```
Section "Device" [...] Option "DynamicClocks" "on" EndSection
```

Управление питанием беспроводных соединений

Карточка беспроводного соединения потребляет немалую часть энергии. Переведите их в режим сохранения энергии так, как вы сделали это с жёсткими дисками.

Примечание: В скрипте предполагается, что ваш беспроводной интерфейс - wlan0. Если в вашей системе это не так, что просто замените его на своё.

Добавьте следующую строку в /etc/conf.d/net для автоматического включения управления питанием беспроводной карты:

Листинг 6.2: Автоматизированное управления питанием WLAN

```
iwconfig_wlan0="power on"
```

Обратитесь к *man iwconfig* за деталями и другими опциями такими, как интервалы между пробуждениями или настройки тайм аута. Если ваш драйвер и точка доступа поддерживает изменение частоты сигналов, то из этого можно извлечь ещё больше сохранённой энергии.

Управление питанием USB

Есть две проблемы с потреблением энергии USB-устройствами. Во-первых, устройства такие, как USB мыши, цифровые камеры или флэшки потребляют энергию всё время, пока подключены, вы не можете избежать этого (кроме как отключая их, когда в них нет необходимости). Во-вторых, когда USB устройство подключено, контроллер USB хоста время от времени опрашивает шину, что не даёт CPU уйти в спящий режим. Ядро предоставляет экспериментальную опцию для включения приостановления (suspend) для USB устройств посредством

вызова драйвера или одного из файлов power/state в /sys.

Листинг 6.3: Включение поддержки приостановления (suspend) USB в ядре

```
Device Drivers
USB support
[*]   Support for Host-side USB
[*]   USB suspend/resume (EXPERIMENTAL)
```

7. Состояния спячки: *sleep, standby, suspend to disk*

Примечание переводчика: данный раздел существенно устарел. Более свежую информацию можно найти на gentoo-wiki.com (EN). Также в русской есть короткий пересказ: gentoo-wiki.com.

ACPI определяет несколько состояний спячки. Основные три из них:

- S1 aka Standby
- S3 aka Suspend to RAM aka Sleep
- S4 aka Suspend to Disk aka Hibernate

Они могут вызываться в моменты, когда компьютер не используется, но выключение не желательно из-за долгой загрузки

Sleep (S3)

Поддержка ACPI для этих состояний спячки помечена экспериментальной, и для этого есть достаточно оснований. Возможно, состояния спячки в APM (Advanced Power Management) несколько более стабильны, но использовать APM и ACPI одновременно нельзя.

Листинг 7.1: Конфигурация ядра для включений состояний спячки

```
Power Management Options --->
[*]   Power Management support
      ACPI (Advanced Configuration and Power Interface) Support --->
      [*]   ACPI Support
      [*]   Sleep States
```

Когда ядро настроено, вы можете воспользоваться скриптом *hibernate-script*, чтобы активировать suspend или sleep. Для начала его надо установить.

Листинг 7.2: Установка hibernate-script

emerge hibernate-script

Необходимо некоторые настройки в /etc/hibernate. По умолчанию пакет предоставляет несколько конфигурационных файлов для каждого состояния спячки. Общие настройки помещены в common.conf; убедитесь, что этот файл правильно сконфигурирован для вашей системы.

Чтобы настроить sleep, отредактируйте sysfs-ram.conf в /etc/hibernate. Опция *UseSysfsPowerState mem* уже выставлена правильно, но если вы хотите произвести дальнейшие настройки в этом или любом другом состоянии спячки, то не забудьте добавить их в /etc/hibernate/hibernate.conf. Комментарии и названия опций помогут вам разобраться. Если вы используете nfs или сетевые ресурсы самбы (samba shares over the network), то необходимо завершать соответствующие скрипты инициализации, чтобы избежать тайм аутов (timeouts).

Примечание: Дополнительную информацию о состояниях спячки см. в *man hibernate.conf*

Готовы? У вас остался последний шанс сделать backup данных прежде, чем запустить следующую команду. Обратите внимание, что вам возможно потребуется нажать специальную клавишу такую, как *Fn*, чтобы вернуться из спячки.

Листинг 7.3: Засыпание

hibernate-ram

Если вы всё ещё читаете, то, видимо, всё работает. Вы можете таким же образом настроить *standby* (S1), отредактировав *sysfs-ram.conf* и изменив *UseSysfsPowerState mem* на *UseSysfsPowerState standby*. Режимы S3 и S4 более интересны, поскольку сохраняют больше энергии.

Hibernate (S4)

Эта секция посвящена hibernation. В этом режиме снимок (snapshot) работающей системы записывается на диск, перед выключением. При включении, снимок загружается и вы можете продолжать работы с момент выключения.

Предупреждение: Не заменяйте устройства не поддерживающие горячую замены во время остановки. Не пытайтесь загрузить снимок на другой машине. Отключите все сетевые ФС и клиенты/серверы самбы перед засыпанием.

Существуют две различных реализации S4. Оригинальная - *swsusp*, и более новая *tuxonice* с более приятным интерфейсом (с поддержкой *fb splash*). [Сравнение их возможностей](#) вы можете найти на [домашней странице tuxonice](#). *There used to be Suspend-to-Disk (pmdisk), a fork of swsusp, but it has been merged back.*

TuxOnIce пока не включен в основную ветку ядра (mainline kernel), поэтому вам либо надо пропатчить ядро (патчи можно найти на [tuxonice.net](#)), либо использовать *sys-kernel/tuxonice-sources*.

Настройки ядра и для *swsusp* и для TuxOnIce следующие:

Листинг 7.4: Настройки ядра для hibernate

```
Power Management Options --->
(hibernate with swsusp)
[*] Software Suspend
    (replace /dev/SWAP with your swap partition)
    (/dev/SWAP)          Default resume partition

(hibernate with TuxOnIce)
Enhanced Hibernation (TuxOnIce)
--- Image Storage (you need at least one writer)
[*]      File Writer
[*]      Swap Writer
--- General Options
[*]      LZF image compression
(replace /dev/SWAP with your swap partition)
(swaps:/dev/SWAP)    Default resume device name
[ ]      Allow Keep Image Mode
```

Настройка *swsusp* много проще. Если вы не вписали имя своп-раздела в настройках ядра, вы должны передать его как параметр в директиве *resume=/dev/*

SWAP. Если загрузка невозможна из-за сломанного образа, воспользуйтесь параметром *noresume*. Скрипт инициализации *hibernate-cleanup* чистит (invalidates) образы swsusp во время загрузки.

Листинг 7.5: Чистка образов swsusp во время загрузки

```
# rc-update add hibernate-cleanup boot
```

Для активации swsusp, используйте *hibernate script*, установив *UseSysfsPowerState disk* в */etc/hibernate/sysfs-disk*.

Предупреждение: Сделайте резервную копию данных перед остановкой. Запустите *sync*, чтобы записать на диск всю кэшированную информацию. Вначале, попробуйте сделать это без иксов, потом в иксах, но не логинясь.

Если вы наталкиваетесь на панику ядра (kernel panic) из-за uhci или подобного, попробуйте скомпилировать поддержку usb модулем и выгрузить модули из памяти перед остановкой. Соответствующая опция есть в *common.conf*.

Листинг 7.6: Засыпание посредством swsusp

```
# nano -w /etc/hibernate/common.conf (Час X - вы сделали резервную копию?)
```

```
# hibernate
```

В следующей секции обсуждается установка TuxOnIce, включая поддержку fbsplash для симпатичного графического индикатора прогресса во время засыпания и просыпания.

Первая часть конфигурации похожа на настройку swsusp. Если вы не вписали имя своп-раздела в настройках ядра, вы должны передать его как параметр в директиве *resume=swap:/dev/SWAP*. Если загрузка невозможна из-за сломанного образа, воспользуйтесь параметром *noresume*. Скрипт инициализации *hibernate-cleanup* чистит (invalidates) образы TuxOnIce во время загрузки.

Листинг 7.7: Чистка образов TuxOnIce во время загрузки

```
# rc-update add hibernate-cleanup boot
```

Теперь, отредактируйте */etc/hibernate/suspend2.conf*, включите опцию *TuxOnIce*. Не включайте *fbsplash* в *common.conf* пока.

Листинг 7.8: Засыпание посредством TuxOnIce

```
# nano -w /etc/hibernate/suspend2.conf (Час X - вы сделали резервную копию?)
```

```
# hibernate
```

Настройте *fbsplash*, если вы не сделали этого ранее. Чтобы задействовать поддержку fbsplash во время засыпания, необходимо поставить пакет *sys-apps/tuxonice-userui*. Также, вам необходимо установить USE-флаг *fbsplash*.

Листинг 7.9: Установка tuxonice-userui

```
# echo "sys-apps/tuxonice-userui fbsplash" >> /etc/portage/package.use
(It may be marked ~arch, so first it must be keyworded)
# echo "sys-apps/tuxonice-userui" >> /etc/portage/package.keywords
# emerge tuxonice-userui
```

Ебилд попросит вас создать символическую ссылку на тему, которую вы хотите использовать. Например, чтобы использовать тему *livecd-2005.1*, выполните следующую команду:

Листинг 7.10: Использование темы livecd-2005.1 во время засыпания

```
# ln -sfn /etc/splash/livecd-2005.1 /etc/splash/suspend2
```

Если вы не хотите видеть чёрный экран в первой части процесса пробуждения, добавьте *tuxoniceui_fbsplash* в ваш образ initrd. Если ваш образ initrd называется *splash_geninitramfs* и он сохранён в */boot/fbsplash-emergence-1024x768*, то вам

необходимо сделать буквально следующее:

Листинг 7.11: Добавление `tuxoniceui_fbsplash` в образ `initrd`.

```
# mount /boot
# mkdir ~/initrd.d
# cp /boot/fbsplash-emergence-1024x768 ~/initrd.d/
# cd ~/initrd.d
# gunzip -c fbsplash-emergence-1024x768 | cpio -idm --quiet -H newc
# rm fbsplash-emergence-1024x768
# cp /usr/sbin/tuxoniceui_fbsplash sbin/
# find . | cpio --quiet --dereference -o -H newc | gzip -9 > /boot/fbsplash-tuxonice-
emergence-1024x768
```

Далее, отрехтуйте ваш `grub.conf` (или `lilo.conf`), чтоб при загрузке вашего ядра грузился образ `/boot/fbsplash-tuxonice-emergence-1024x768`. Вы теперь можете попробовать холостой прогон, чтоб убедиться, что всё работает правильно.

Листинг 7.12: Тестовый запуск `fbsplash hibernation`

```
# tuxoniceui_fbsplash -t
```

Наконец, откройте `/etc/hibernate/common.conf` и включите поддержку `fbsplash`. Запустите `hibernate` и наслаждайтесь!

8. Устранение неполадок

В: Я пытаюсь изменить частоту ЦПУ, но `/sys/devices/system/cpu/cpu0/cpufreq/scaling_governor` не существует.

О: Убедитесь, что ваш процессор поддерживает изменение частоты и выберите правильный `CPUFreq` драйвер для вашего процессора. Список процессоров, поддерживающих изменение частоты (ядро 2.6.7): ARM Integrator, ARM-SA1100, ARM-SA1110, AMD Elan - SC400, SC410, AMD mobile K6-2+, AMD mobile K6-3+, AMD mobile Duron, AMD mobile Athlon, AMD Opteron, AMD Athlon 64, Cyrix Media GXm, Intel mobile PIII и Intel mobile PIII-M на определённых чипсетах, Intel Pentium 4, Intel Xeon, Intel Pentium M (Centrino), National Semiconductors Geode GX, Transmeta Crusoe, VIA Cyrix 3 / C3, UltraSPARC-III, SuperH SH-3, SH-4, некоторые "PowerBook" и "iBook2" и различные процессоры на совместимых с ACPI 2.0 системах (только если "ACPI Processor Performance States" доступно через интерфейс ACPI/BIOS).

В: Мой ноутбук поддерживает изменение частоты, но директория `/sys/devices/system/cpu/cpu0/cpufreq/` пуста.

О: Поищите связанные с ACPI ошибки в `dmesg | grep ACPI`. Попробуйте обновить BIOS, особенно, если обнаружены проблемы в DSDT. Вы также можете попробовать исправить это самостоятельно (что выходит за рамки этого руководства).

В: Мой ноутбук поддерживает изменение частоты, но согласно `/proc/cpuinfo` частота никогда не изменяется.

О: Возможно, вы включили поддержку симметричной мультипроцессорной обработки (`symmetric multiprocessing support`) (`CONFIG_SMP`) в вашем ядре. Попробуйте отключить её, всё должно заработать. Также это может быть багом старых ядер. В таком случае выполните `emerge x86info`, обновите ваше ядро как требуется и проверьте текущую частоту через `x86info -mhz`.

В: Я могу изменять частоты, но выбор не так широк, как на другой ОС.

О: Вы можете совместить изменение частоты с "торможением" через ACPI (ACPI throttling), чтобы добиться более низких частот. Но стоит понимать, что это не даст большого прироста в сохранении энергии и создана, по большей части, для температурного регулирования (чтобы ноутбук был холодным и тихим). Вы можете посмотреть текущее состояние "торможения" командой `cat /proc/acpi/processor/CPU/throttling` и изменить его, выполнив `echo -n "0:x" > /proc/acpi/processor/CPU/limit`, где *x* - одно из возможных состояний, перечислены в `/proc/acpi/processor/CPU/throttling`.

В: При конфигурировании ядра пункты `powersave governor`, `performance governor` и `userspace governor` есть, а `ondemand governor` отсутствует. В чём проблема?

О: Пункт `ondemand governor` появился только в последних версиях ядра. Попробуйте обновить его.

В: Время жизни от батареи стало меньше, чем было.

О: Проверьте настройки BIOS. Возможно, вы забыли заново включить какие-то опции.

В: Моя батарея заряжена, но KDE сообщает, что осталось 0%, и тут же выключается.

О: Убедитесь, что поддержка батареи выключена в вашем ядре. Если она скомпилирована модулем, то не забудьте загрузить его.

В: Моя система журналирования (system logger) пишет что-то, типа "logger: ACPI group battery / action battery is not defined".

О: Это сообщение генерирует скрипт `/etc/acpi/default.sh`, которые прилагается к `acpid`. Вы можете просто игнорировать его. Если же вы всё-таки хотите его унять, прокомментируйте соответствующую строку в `/etc/acpi/default.sh` как показано ниже:

Листинг 8.1: Отключение предупреждения о событиях ACPI

*) `# logger "ACPI action $action is not defined"`

В: У меня Dell Inspiron 51XX, и я не получаю никаких ACPI-событий

О: Скорее всего, это баг ядра. Прочтите [это](#).

В: Я включил *DynamicClocks* в `xorg.conf` и теперь X.org рушится / экран остаётся чёрным / мой ноутбук не выключается корректно.

О: Такое бывает на некоторых системах. Вам придётся отключить *DynamicClocks*.

В: Я хочу использовать TuxOnIce, но он говорит, что моя своп - партиция слишком маленькая. Изменение её размеров - не вариант.

О: Если в вашей системе достаточно свободного места, то вы можете использовать запись в файл, вместо запись в своп. Такой вариант поддерживается *hibernate-script*. Дополнительную информацию вы можете найти в `/usr/src/linux/Documentation/power/tuxonice.txt`.

В: Я только купил новую фирменную батарею, но она кончается за несколько минут. Что я делаю не так?

О: Для начала, следуйте инструкция производителя, как зарядить батарею правильно.

В: Выше написанное не помогло, что делать теперь?

О: Некоторые батареи продаются как "новые", но на самом деле уже не новы. Попробуйте следующее:

Листинг 8.2: Запрос состояния батареи

```
$ grep capacity /proc/acpi/battery/BAT0/info
design capacity:      47520 mWh
last full capacity:  41830 mWh
```

Если "last full capacity" кардинально отличается от design capacity, ваша батарея, скорее всего, неисправна. Попробуйте потребовать гарантию.

В: Моя проблема не перечислена выше. Что мне делать теперь?

О: Не бойтесь связаться со мной, [Dennis Nienhüser](#), напрямую. Также вам, наверняка, помогут на [Форумax Gentoo](#). Если вы предпочитаете IRC, заходите в канал [#gentoo-laptop](#) на [irc.freenode.net](#)

Замечания

к

переводу

Не удалось найти хороший перевод слову governor. Хороший означает, что оно не слишком общо (как, например, драйвер), ну и подходит по смыслу ;) . Поэтому в тексте используется слово властелин, формально подходящее под условия, но слишком уж не формальное.

Gentoo и СУБД

Руководство по MySQL для начинающих

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/mysql-howto.xml>

С версии: 1.5

1. Начало работы с MySQL

Введение

MySQL — это популярный сервер баз данных, используемый в различных областях. Аббревиатура SQL означает Structured Query Language (язык структурированных запросов), именно его использует MySQL для общения с другими программами. Более того, в MySQL есть собственные расширенные функции SQL, дающие дополнительные возможности пользователям. В этом руководстве рассматривается процесс начальной установки MySQL, настройка баз данных и таблиц, заведение новых пользователей. Итак, начнем с установки.

Установка MySQL

Сначала нужно установить MySQL. Если вам нужны определенные возможности MySQL, установите соответствующие флаги USE, которые позволяют точнее регулировать процесс установки.

Листинг 1.1: Установка MySQL

```
(просмотр имеющихся флагов USE)
# emerge --pretend --verbose mysql
(установка MySQL)
# emerge mysql
```

По завершении установки вы увидите такое сообщение:

Листинг 1.2: Сообщение программы MySQL einfo

```
You might want to run:
"emerge --config =dev-db/mysql-[version]"
if this is a new install.
```

Рекомендуется запустить команду
"emerge --config =dev-db/mysql-[версия]"
после первоначальной установки MySQL.

Поскольку у нас первоначальная установка, запустим команду. Во время настройки базы данных MySQL вам потребуется по запросу нажать клавишу ENTER. В процессе настройки устанавливается главная база данных MySQL, содержащая служебные сведения о базах данных, таблицах, пользователях, правах доступа и т.д. При настройке будет рекомендовано как можно раньше изменить корневой пароль. Мы определенно это сделаем, чтобы кто-нибудь не смог улучшить момент и взломать наш сервер MySQL, настроенный по умолчанию.

Листинг 1.3: Настройка MySQL

```
# ebuild /var/db/pkg/dev-db/mysql-[version]/mysql-[version].ebuild config
* MySQL DATADIR is /var/lib/mysql
* Press ENTER to create the mysql database and set proper
* permissions on it, or Control-C to abort now...

Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables

To start mysqld at boot time you have to copy support-files/mysql.server
to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, issue the following commands to start the server
and change the applicable passwords:
(обратите внимание на следующие 3 строки)
/etc/init.d/mysql start
/usr/bin/mysqladmin -u root -h pegasos password 'new-password'
/usr/bin/mysqladmin -u root password 'new-password'
Depending on your configuration, a -p option may be needed
in the last command. See the manual for more details.

(различные сообщения, не относящиеся к установке, опущены для краткости)

* For security reasons you should set your MySQL root
* password as soon as possible.
```

Важно: Начиная с mysql-4.0.24-r2, пароли вводятся на этапе настройки, что повышает надежность ввода корневого пароля.

Сценарий установки уже отобразил команды, нужные для установки пароля. Теперь запустим их.

Листинг 1.4: Установка корневого пароля MySQL

```
# /etc/init.d/mysql start
* Re-caching dependency info (mtimes differ)...
* Starting mysqld (/etc/mysql/my.cnf) ... [ ok ]
(вместо 'новый-пароль' укажите ваш пароль)
# /usr/bin/mysqladmin -u root -h localhost password 'новый-пароль'
```

Теперь вы можете убедиться в работоспособности нового пароля, попытавшись войти на свой сервер MySQL:

Листинг 1.5: Вход на сервер MySQL программой mysql

```
$ mysql -u root -h localhost -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 4.0.25

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

```
mysql>
```

Параметр `-u` указывает пользователя, который будет входить в систему. Параметр `-h` указывает узел сети. Обычно это `localhost`, если вы настраиваете не удаленный сервер. Наконец, `-p` сообщает программе-клиенту `mysql`, что для доступа к базе данных будет вводиться пароль. Обратите внимание на приглашение `mysql>`. Именно здесь вы будете вводить все свои команды. Теперь, находясь в среде `mysql` в качестве корневого пользователя, мы можем начать настройку базы данных.

2. Настройка базы данных

Создание базы данных

Мы вошли, и на экране — приглашение `mysql`. Сначала взглянем на список уже имеющихся баз данных. Для этого введем команду `SHOW DATABASES`.

Листинг 2.1: Вывод списка баз данных MySQL

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| mysql    |
| test     |
+-----+
2 rows in set (0.09 sec)
```

Важно: Запомните, что команды MySQL следует заканчивать точкой с запятой — ; Несмотря на то, что база данных `test` уже создана, создадим свою собственную. Базы данных создаются командой `CREATE DATABASE`. Мы назовем свою `gentoo`.

Листинг 2.2: Создание базы данных gentoo

```
mysql> CREATE DATABASE gentoo;
Query OK, 1 row affected (0.08 sec)
```

Ответ дает понять, что команда выполнена без каких-либо ошибок. В данном случае, изменилась одна строка. Это относится к главной базе данных `mysql`, в которой содержится список всех баз данных. Но вам не нужно слишком беспокоиться о второстепенных подробностях. Последнее число означает время выполнения запроса. Убедиться, что база данных создана, мы можем, снова запустив команду `SHOW DATABASES`.

Листинг 2.3: Проверка наличия базы данных

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| gentoo   |
| mysql    |
| test     |
+-----+
3 rows in set (0.00 sec)
```

Разумеется, наша база данных создана. Чтобы заняться созданием таблиц в новой базе данных gentoo, нам потребуется установить ее как текущую. Для этого используем команду USE. Параметром этой команды указывается название базы данных, которую нужно сделать текущей. Еще текущую базу можно устанавливать в командной строке, указывая ее название после параметра -D. Давайте продолжим и переключимся на базу данных gentoo.

Листинг 2.4: Переключение базы данных

```
mysql> USE gentoo;
Database changed
```

Теперь текущей является только что созданная база данных gentoo. Теперь, используя ее, мы можем заняться созданием таблиц и наполнением их информацией.

3. Работа с таблицами в MySQL

Создание таблицы

В структуру MySQL входят базы данных, таблицы, записи и поля. В базах данных собраны таблицы, в таблицах собраны записи, в записях — поля, в которых, в свою очередь, хранится собственно информация. Такая структура позволяет пользователям выбирать, каким образом обращаться к своей информации. Только что мы работали с базами данных, теперь давайте поработаем с таблицами. Для начала, список таблиц можно вывести так же, как и список баз данных, используя команду SHOW TABLES. Сейчас в нашей базе данных gentoo еще нет таблиц, как показывает эта команда:

Листинг 3.1: Пустая база данных gentoo

```
mysql> SHOW TABLES;
Empty set (0.00 sec)
```

Значит, нужно создать несколько таблиц. Чтобы это сделать, используем команду CREATE TABLE. Однако, эта команда довольно сильно отличается от простой команды CREATE DATABASE. Ей передается список аргументов следующего вида:

Листинг 3.2: Синтаксис CREATE TABLE

```
CREATE TABLE [имя_таблицы] ([имя_поля] [тип_данных_поля] ([размер]));
```

имя_таблицы — имя создаваемой таблицы. В данном случае, давайте создадим таблицу с названием developers (разработчики). В таблице будут находиться имена разработчиков, адреса электронной почты и род занятий. **имя_поля** будет содержать имена полей. Нам нужны три имени: name (имя), email (почта) и job (род занятий). В параметре **тип_данных_поля** укажем тип заносимой информации. Перечень возможных форматов находится на [странице описания типов столбцов MySQL \(англ.\)](#). Здесь мы для всех полей укажем тип VARCHAR. VARCHAR — один из простейших типов данных, предназначенный для работы со строками. размер указывает, сколько данных можно сохранить в одном поле. Укажем 128. Это значит, что поле сможет содержать 128 байт данных типа VARCHAR. Сейчас можно упрощенно считать, что это 128 знаков текста, хотя на

вышеупомянутом сайте представлено более точное описание. Теперь, зная, какую мы собираемся создать таблицу, сделаем это.

Листинг 3.3: Создание таблицы

```
mysql> CREATE TABLE developers ( name VARCHAR(128), email VARCHAR(128), job
VARCHAR(128));
Query OK, 0 rows affected (0.11 sec)
```

Похоже, таблица создалась нормально. Давайте проверим это командой SHOW TABLES:

Листинг 3.4: Проверка таблицы

```
mysql> SHOW TABLES;
+-----+
| Tables_in_gentoo |
+-----+
| developers        |
+-----+
1 row in set (0.00 sec)
```

Да, это она! Но не показано никакой информации о введенных типах полей. Для этого используем команду DESCRIBE (или ее краткий вариант, DESC), параметром которой указывается имя таблицы. Посмотрим, что она выдаст для таблицы developers.

Листинг 3.5: Описание таблицы developers

```
mysql> DESCRIBE developers;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| name  | varchar(128)  | YES  |     | NULL    |       |
| email | varchar(128)  | YES  |     | NULL    |       |
| job   | varchar(128)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Показаны различные поля и их типы. Видно также несколько дополнительных атрибутов, выходящих за рамки этого руководства. За дополнительными сведениями можно обратиться к [Руководству по MySQL \(англ.\)](#). Теперь у нас есть рабочая таблица. Давайте двинемся дальше и наполним ее.

Заполнение базы данных MySQL

Таблица заполняется (данные добавляются) командой INSERT. Как и у команды CREATE TABLE, у нее есть определенный формат:

Листинг 3.6: Синтаксис INSERT

```
INSERT INTO имя_таблицы (столбец1, столбец2, ...)
VALUES ('знач1', 'знач2', ...);
```

Эту команду используют для вставки записей в таблицу. Сначала указывается имя таблицы, в которую нужно добавить информацию. Затем может идти список столбцов, в которые добавляются данные, а в VALUES указываются значения, которые добавляются в таблицу. Можно опустить список полей, если значения добавляются в каждое в том же порядке, в каком определены поля в таблице. Сейчас мы добавим данные в таблицу developers. Вставьте записи как в примере:

Листинг 3.7: Добавление информации в таблицу developers

```
mysql> INSERT INTO developers VALUES('Joe Smith', 'joesmith@gentoo.org',
'toolchain');
Query OK, 1 row affected (0.06 sec)
(Если вы не знаете порядок полей в таблице или хотите добавить
неполную запись)
mysql> INSERT INTO developers (job, name) VALUES('outsourced', 'Jane
Doe');
Query OK, 1 row affected (0.01 sec)
```

Согласно полученному ответу, запись, похоже, добавилась правильно. Но что если требуется ввести более одной записи? Именно здесь пригодится команда **LOAD DATA**. Она загружает записи из файла, разделенного знаками табуляции. Попробуем создать файл с записями в своем домашнем каталоге. Назовем его **records.txt**. Вот пример:

Листинг 3.8: ~/records.txt

John Doe	johndoe@gentoo.org	portage
Chris White	chriswhite@gentoo.org	documentation
Sam Smith	samsmith@gentoo.org	amd64

Важно: Всегда уточняйте, с какими данными работаете. Чрезвычайно небезопасно использовать **LOAD DATA**, если вы не знаете, что находится внутри файла!

Вообще, у команды **LOAD DATA** весьма пространное определение, но сейчас мы используем простейшую форму.

Листинг 3.9: Синтаксис LOAD DATA

```
LOAD DATA LOCAL INFILE '/путь/к/файлу' INTO TABLE имя_таблицы;
```

Здесь тоже все просто. Укажите путь к файлу и имя таблицы. В нашем случае — это файл **~/records.txt** и таблица **developers**.

Листинг 3.10: Загрузка данных

```
mysql> LOAD DATA LOCAL INFILE '~/records.txt' INTO TABLE developers;
Query OK, 3 rows affected (0.00 sec)
Records: 3 Deleted: 0 Skipped: 0 Warnings: 0
```

Важно: Если вы столкнетесь со странностями, убедитесь, что поля разделены табуляторами. Если вы вставляете информацию в файл из другого источника, табуляторы могут преобразоваться в пробелы.

Сработало. Но эта команда просто добавляет записи и не дает проконтролировать работу MySQL. Множество веб-приложений используют сценарии **sql** для быстрой и легкой настройки базы данных MySQL. Если вы хотите использовать сценарий **sql**, вам понадобится запустить **mysql** в пакетном режиме (в качестве ввода используется файл со сценарием). Вот пример запуска **mysql** в пакетном режиме:

Листинг 3.11: MySQL в пакетном режиме

```
$ mysql -u root -h localhost -p < sqlfile
```

Как и в случае с **LOAD DATA**, убедитесь, что знаете, что именно выполняет файл **sqlfile**. Не делая этого, вы можете подвергнуть вашу базу данных серьезному

риску утечки! Другой путь заключается в использовании команды `source`. Эта команда запускает команды `mysql` из файла, когда `mysql` находится в интерактивном режиме. Вот как взять команды `sql` из файла:

Листинг 3.12: Исполнение команд `sql` из файла

```
mysql> source sqlfile;
```

Если вы столкнетесь с веб-приложением, которому нужно, чтобы вы запустили файл с командами `sql`, можно использовать одну из двух команд, показанных выше. Итак, наша таблица сформирована, но как же работать с полями? Это делается поиском в таблице при помощи запросов.

Запросы к таблицам в MySQL

Одна из главных функций любой базы данных SQL — запросы. Они помогают обратить данные из таблиц во что-то полезное. Большинство запросов выполняется командой `SELECT`. Эта команда довольно сложна, поэтому в этом документе мы рассмотрим лишь три основных ее формы.

Листинг 3.13: Формы `SELECT`

```
(Выборка всех записей в таблице)
SELECT * FROM имя_таблицы;
(Выборка конкретных записей в таблице)
SELECT * FROM имя_таблицы WHERE поле=значение;
(Выборка конкретных полей)
SELECT поле1,поле2,поле3 FROM имя_таблицы [WHERE поле=значение];
```

Посмотрим на первую форму. Она относительно проста и позволяет взглянуть на всю таблицу. Попробуем: запустим эту команду, чтобы увидеть, что находится в нашей таблице.

Листинг 3.14: Содержимое таблицы `developers`

```
mysql> SELECT * FROM developers;
+-----+-----+-----+
| name      | email                      | job              |
+-----+-----+-----+
| Joe Smith  | joesmith@gentoo.org        | toolchain        |
| John Doe   | johndoe@gentoo.org         | portage          |
| Chris White| chriswhite@gentoo.org      | documentation     |
| Sam Smith  | samsmith@gentoo.org        | amd64            |
| Jane Doe   | NULL                       | Outsourced job   |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

Видно данные, не только вставленные путем `INSERT`, но и вставленные командой `LOAD DATA`. Теперь, к примеру, нужно посмотреть запись для `Chris White`. Это можно сделать, используя вторую форму выборки.

Листинг 3.15: Выборка конкретной записи, используя `SELECT`

```
mysql> SELECT * FROM developers WHERE name = 'Chris White';
+-----+-----+-----+
| name      | email                      | job              |
+-----+-----+-----+
| Chris White| chriswhite@gentoo.org      | documentation     |
+-----+-----+-----+
```



```
1 row in set (0.08 sec)
```

Как и ожидалось, отображена нужная искомая запись. Теперь, допустим, что нужно знать только род занятий и адрес электронной почты, но не имя. Это делается при помощи третьей формы SELECT, как показано ниже.

Листинг 3.16: Выборка нужной записи и полей с помощью SELECT

```
mysql> SELECT email,job FROM developers WHERE name = 'Chris White';
+-----+-----+
| email                | job                |
+-----+-----+
| chriswhite@gentoo.org | documentation      |
+-----+-----+
1 row in set (0.04 sec)
```

Этот способ выборки легче в управлении, особенно при больших объемах информации, как показано далее. Сейчас, войдя как суперпользователь, вы обладаете неограниченными правами на любые действия в MySQL. Пользователь с такими привилегиями в серверной среде может быть достаточно трудноуправляемым. Чтобы управлять тем, кто и что может делать с базами данных, надо устанавливать привилегии.

4. Привилегии в MySQL

Предоставление привилегий командой GRANT

Привилегиями определяются возможности доступа пользователей к базам данных, таблицам... почти ко всему. Сейчас только суперпользователь root из MySQL может обращаться к базе данных gentoo, согласно данным разрешениям. Давайте создадим двух самых обычных пользователей, например, guest (гостя) и admin (администратора), которые будут обращаться к базе данных gentoo и работать с информацией из нее. Пусть пользователь guest будет ограничен в правах, и все, что он сможет — это получать информацию из базы данных. admin получит те же права доступа, что и root, но только к базе данных gentoo (не к основным базам mysql). Но прежде, чем начать, давайте взглянем на несколько упрощенный формат команды GRANT.

Листинг 4.1: Синтаксис GRANT

```
GRANT [привилегии] ON база_данных.* TO '[пользователь] '@'[узел] '
IDENTIFIED BY '[пароль]';
```

Примечание: Команда GRANT считается способом создания пользователя. Поздние версии MySQL, однако, также содержат функцию CREATE_USER, хотя GRANT до сих пор предпочтительнее.

Теперь нужны привилегии, которые можно присваивать. Используя все вышесказанное, можно устанавливать следующие привилегии:

- ALL - дается полный доступ ко всей базе данных
- CREATE - пользователям позволяет создавать таблицы
- SELECT - пользователям позволяет делать запросы к таблицам
- INSERT - пользователям позволяет вставлять данные в таблицу
- SHOW DATABASES - разрешается выводить список баз данных

- USAGE - у пользователя нет привилегий
- GRANT OPTION - пользователям разрешается давать привилегии

Примечание: Если вы используете MySQL для обмена данными с веб-приложением, то разъясняемые здесь привилегии CREATE, SELECT, INSERT, а также привилегии DELETE и UPDATE (описанные в [руководстве по MySQL, раздел GRANT and REVOKE Syntax \(англ.\)](#)) — единственные, которые, вероятно, потребуются. Многие совершают ошибку, раздавая все привилегии, когда это, в действительности, не нужно. Сверьтесь с разработчиками приложений, действительно ли такие разрешения создадут проблемы в работе.

Для нашего пользователя admin подойдет ALL. А для пользователя guest привилегии SELECT будет достаточно для доступа на чтение. В качестве базы данных укажем gentoo, на нее будут установлены разрешения. * означает «все таблицы». Если потребуется, можно установить права доступа к отдельным таблицам. Дальше идет имя пользователя и имя узла, с которого будет работать пользователь. В большинстве случаев это будет узел localhost. Наконец, задается пароль пользователя. Исходя из этого, давайте создадим пользователей.

Листинг 4.2: Создание пользователей admin и guest

```
(admin)
mysql> GRANT ALL ON gentoo.* TO 'admin'@'localhost' IDENTIFIED BY 'пароль';
(guest)
mysql> GRANT SELECT ON gentoo.* TO 'guest'@'localhost' IDENTIFIED BY
'пароль';
```

Итак, пользователи созданы; теперь протестируем их. Сначала выйдем из mysql, написав quit в строке приглашения:

Листинг 4.3: Выход из MySQL

```
mysql> quit
```

Теперь мы снова в консоли. Пользователи настроены, давайте посмотрим, что они могут делать.

Проверка прав пользователей

Попробуем зайти как пользователь guest. В данный момент у пользователя guest есть только право выборки (SELECT). В основном, это сводится к возможности поиска и ничему другому. Зайдем как пользователь guest.

Листинг 4.4: Вход опользователя guest

```
$ mysql -u guest -h localhost -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 4.0.25

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Теперь нам нужно проверить его ограничения. Переключимся на базу данных gentoo:

Листинг 4.5: Переключение на базу gentoo

```
mysql> USE gentoo;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

Теперь попытаемся сделать что-нибудь недозволенное. Попробуем создать таблицу.

Листинг 4.6: Попытка создать таблицу пользователем guest

```
mysql> CREATE TABLE test (test VARCHAR(20), foobar VARCHAR(2));
ERROR 1044: Access denied for user: 'guest@localhost' to database 'gentoo'
```

Как видно, это не удалось, из-за того, что у пользователя недостаточно прав. Но ему дано право на использование SELECT. Давайте проверим:

Листинг 4.7: Попытка выборки командой SELECT

```
mysql> SELECT * FROM developers;
+-----+-----+-----+
| name      | email                      | job              |
+-----+-----+-----+
| Joe Smith  | joesmith@gentoo.org        | toolchain        |
| John Doe   | johndoe@gentoo.org         | portage          |
| Chris White| chriswhite@gentoo.org      | documentation    |
| Sam Smith  | samsmith@gentoo.org        | amd64            |
| Jane Doe   | NULL                       | Outsourced job   |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

Команда успешно отработала, а получили представление о том, на что способны права пользователей. Но мы ведь еще создали пользователя admin. Он создавался для демонстрации того, что даже у пользователей, которым даны все права, могут быть ограничения. Выйдите из MySQL и зайдите как пользователь admin.

Листинг 4.8: Вход пользователя admin

```
mysql> quit
Bye
$ mysql -u admin -h localhost -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.0.25

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Для начала, попробуем создать новую базу данных как пользователь admin. У пользователя admin схожие права с учетной записью root в MySQL, и он может вносить любые изменения, выбрав базу данных gentoo. Мы проверим права его доступа к главной базе данных MySQL. Вспомним, что ранее мы дали ему права только для доступа к конкретной базе данных.

Листинг 4.9: Попытка создания новой базы данных

```
mysql> CREATE DATABASE gentoo2;
```

```
ERROR 1044: Access denied for user: 'admin@localhost' to database 'gentoo2'
```

Разумеется, пользователь admin не может создавать базы данных в основной базе MySQL, несмотря на все права на базу данных gentoo. Но admin все еще может изменять базу данных gentoo, как показывает вставка данных в следующем примере.

Листинг 4.10: Права admin в базе данных gentoo

```
mysql> USE gentoo;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> INSERT INTO developers VALUES('Bob Simmons', 'bobsimmons@gentoo.org',
'python');
Query OK, 1 row affected (0.08 sec)
```

Пользователь admin может обращаться к базе gentoo, как захочет. Но иногда требуется лишить пользователя прав. Это может быть что угодно, начиная с проблематичного пользователя, и заканчивая ушедшим сотрудником. Давайте разберемся, как отзывать разрешения с помощью команды REVOKE.

Удаление прав пользователя командой REVOKE

Команда REVOKE позволяет запретить доступ пользователю. Можно либо запретить любой доступ, либо только определенный. В самом деле, формат очень похож на GRANT.

Листинг 4.11: Синтаксис REVOKE

```
REVOKE [привилегии] ON база_данных.* FROM '[пользователь]@[узел]';
```

Параметры объясняются в разделе команды GRANT. А сейчас мы запретим пользователю любой вид доступа. Скажем, мы выяснили, что учетная запись guest вызывает проблемы с безопасностью. Мы решаем отозвать все права. Заходим как root делаем необходимое.

Листинг 4.12: Отзыв разрешений для пользователя user

```
mysql> REVOKE ALL ON gentoo.* FROM 'guest'@'localhost';
Query OK, 0 rows affected (0.00 sec)
```

Примечание: В данном случае, доступ пользователя прост, поэтому отмена прав на одну базу данных — не проблема. Но обычно вам, скорее всего, потребуется использовать *.* вместо gentoo.*, чтобы заодно отменить доступ пользователя ко всем остальным базам данных.

Давайте выйдем и попробуем зайти как пользователь guest.

Листинг 4.13: Попытка зайти как пользователь guest

```
$ mysql -u guest -h localhost -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 9 to server version: 4.0.25

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

```
mysql>
```

Хотя нам удалось войти, доступ к базе gentoo уже пропал.

Листинг 4.14: Пользователю guest запрещен доступ

```
mysql> USE gentoo;
ERROR 1044: Access denied for user: 'guest@localhost' to database 'gentoo'
```

И наш проблематичный пользователь больше не может обращаться к базе gentoo. Заметьте, что пользователь сохранил возможность входа. Это потому, что он остается в основной базе данных MySQL. Теперь взглянем, как полностью удалить учетную запись командой DELETE, и посмотрим на таблицу пользователей MySQL.

Удаление учетных записей командой DELETE

Таблица пользователей MySQL — это список пользователей и информации о них. Убедитесь, что вы зашли как root. Используйте основную базу данных MySQL.

Листинг 4.15: Использование основной базы mysql

```
mysql> USE mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

Посмотрим, какие в базе mysql существуют таблицы:

Листинг 4.16: Список таблиц БД mysql

```
mysql> SHOW TABLES;
+-----+
| Tables_in_mysql |
+-----+
| columns_priv    |
| db              |
| func            |
| host            |
| tables_priv     |
| user            |
+-----+
6 rows in set (0.00 sec)
```

Таблица user — та, что нужна. В ней 30 различных полей, и ее сложно читать. Для облегчения чтения используем третью форму команды SELECT. Искомые поля — Host (узел) и User (пользователь).

Листинг 4.17: Нахождение пользователя guest в таблице user

```
mysql> SELECT Host,User FROM user WHERE User = 'guest';
+-----+-----+
| Host      | User  |
+-----+-----+
| localhost | guest |
+-----+-----+
1 row in set (0.00 sec)
```

Теперь, получив информацию, мы можем избавиться от пользователя guest. Это делается командой DELETE; вот ее синтаксис:

Листинг 4.18: Синтаксис DELETE

```
DELETE FROM имя_таблицы WHERE поле='значение';
```

Вы могли заметить, что формат DELETE чем-то схож с форматом SELECT. Укажем поле User и значение guest. Это удалит запись из таблицы user, где пользователь — guest, удаляя нашу гостевую учетную запись. Сделаем так:

Листинг 4.19: Удаление пользователя guest

```
mysql> DELETE FROM user WHERE User='guest';
Query OK, 1 row affected (0.07 sec)
(Команда FLUSH PRIVILEGES нужна, чтобы обновить разрешения)
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

Похоже, сработало. Проверим: выйдите и попробуйте зайти как пользователь guest.

Листинг 4.20: Попытка входа пользователя guest

```
mysql> quit
Bye
$ mysql -u guest -h localhost -p
Enter password:
ERROR 1045: Access denied for user: 'guest@localhost' (Using password: YES)
```

Все, пользователь успешно удален!

Заключение

В этом руководстве в основном рассматривалась настройка MySQL из командной строки. Существует несколько альтернатив с графическим интерфейсом:

- [phpMyAdmin](#) — популярное средство администрирования MySQL, написанное на php.
- [mysqlnavigator](#) — оболочка MySQL для Qt.
- [gmyclient](#) — клиент MySQL для GNOME.
- [knoda](#) — клиент MySQL для KDE.

На этом заканчивается введение в MySQL. Надеюсь, что оно помогло вам лучше разобраться в основах MySQL и настройке базы данных. Пожалуйста, присылайте свои соображения мне по адресу Chris White.

PostgreSQL Guide

Ссылка на оригинал: <http://www.gentoo.org/doc/en/postgres-howto.xml>

С версии: 1.5

1. Introduction

PostgreSQL introduction

When talking to most developers about the different database solutions to use, two major databases will usually form the answer. One would be MySQL, and the other is what this document will refer to, PostgreSQL. The advantages of one over the other is a somewhat long winded debate, however it is just to say that PostgreSQL has had a more firm grasp on true relational database structure than MySQL. Most of the standard features such as **FOREIGN KEY** was only just added in MySQL 5. However, whatever the case may be, this document assumes that you have selected PostgreSQL as the database to use. The first place to start is the emerge process. In the next section, the installation process through emerge will be described, as well as the basic configuration.

PostgreSQL installation

To begin, we must first emerge the PostgreSQL package. To do so, run the following code to first ensure that the options for it are properly set:

Code Listing 1.1: Checking the PostgreSQL build options

```
# emerge -pv postgresql

These are the packages that I would merge, in order:

Calculating dependencies ...done!
[ebuild N    ] dev-db/postgresql-8.0.4  -doc -kerberos +nls +pam +perl -pg-
intdatetime +python +readline (-selinux) +ssl -tcl +xml +zlib 0 kB
```

Here's a list of what the different build options indicate:

USE Flag Meaning

doc	This USE flag enables or disables the installation of documentation outside of the standard man pages. The one good time to disable this option is if you are low on space, or you have alternate methods of getting a hold of the documentation (online, etc.)
kerberos	When connecting to the database, with this option enabled, the admin has the option of using kerberos to authenticate their users/services to the database.
nls	If this option is enabled, PostgreSQL can utilize translated strings for non-English speaking users.
pam	If this option is enabled, and the admin configures the PostgreSQL

	configuration file properly, users/services will be able to login to a PostgreSQL database using PAM (Pluggable Authentication Module).
perl	If this option is enabled, perl bindings for PostgreSQL will be built.
pgsql_intdatetime	If this option is enabled, PostgreSQL will support 64 bit integer date types.
python	If this option is enabled, PostgreSQL will be built with python bindings.
readline	If this option is enabled, PostgreSQL will support readline style command line editing. This includes command history and isearch.
selinux	If this option is enabled, an selinux policy for PostgreSQL will be installed.
ssl	If this option is enabled, PostgreSQL will utilize the OpenSSL library to encrypt traffic between PostgreSQL clients and servers.
tcl	If this option is enabled, PostgreSQL will build tcl bindings.
xml	If this option is enabled, XPATH style xml support will be built. More information on using xml support with PostgreSQL can be found on: PostgreSQL and XML .
zlib	This isn't really used by PostgreSQL itself, but by pg_dump to compress the dumps it produces.

Once you've customized PostgreSQL to meet your specific needs, go ahead and start the emerge:

Code Listing 1.2: Emerge-ing PostgreSQL

```
# emerge postgresql
(Output shortened)
>>> /usr/lib/libecpg.so.5 -> libecpg.so.5.0
>>> /usr/bin/postmaster -> postgres
* Make sure the postgres user in /etc/passwd has an account setup with /bin/
bash as the shell
*
* Execute the following command
* emerge --config =postgresql-8.0.4
* to setup the initial database environment.
*
>>> Regenerating /etc/ld.so.cache...
>>> dev-db/postgresql-8.0.4 merged.
```

As shown by the einfo output, there is some post setup that must be done. The next chapter will look at the actual configuration of PostgreSQL.

2. PostgreSQL configuration

Setting up the initial database environment

As noted in the earlier emerge output, the initial database environment must be setup. However, before this is done, one thing needs to be considered. Unlike, say MySQL, PostgreSQL's "root" password is the password of the actual user. However, only the user is created by the ebuild not the password. So before we can begin, the password must be set for the postgres user:

Code Listing 2.1: Setting the password

```
# passwd postgres
New UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Now that this is setup, the creation of the initial database environment can occur:

Code Listing 2.2: Configuring the database environment with emerge --config

```
# emerge --config =postgresql-8.0.4

Configuring pkg...

* Creating the data directory ...
* Initializing the database ...
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale C.

fixing permissions on existing directory /var/lib/postgresql/data ... ok
creating directory /var/lib/postgresql/data/global ... ok
creating directory /var/lib/postgresql/data/pg_xlog ... ok
creating directory /var/lib/postgresql/data/pg_xlog/archive_status ... ok
creating directory /var/lib/postgresql/data/pg_clog ... ok
creating directory /var/lib/postgresql/data/pg_subtrans ... ok
creating directory /var/lib/postgresql/data/base ... ok
creating directory /var/lib/postgresql/data/base/1 ... ok
creating directory /var/lib/postgresql/data/pg_tblspc ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 1000
creating configuration files ... ok
creating template1 database in /var/lib/postgresql/data/base/1 ... ok
initializing pg_shadow ... ok
enabling unlimited row size for system tables ... ok
initializing pg_depend ... ok
creating system views ... ok
loading pg_description ... ok
creating conversions ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
vacuuming database template1 ... ok
copying template1 to template0 ... ok

WARNING: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the -A option the
next time you run initdb.

Success. You can now start the database server using:

    /usr/bin/postmaster -D /var/lib/postgresql/data
or
    /usr/bin/pg_ctl -D /var/lib/postgresql/data -l logfile start

*
* You can use /etc/init.d/postgresql script to run PostgreSQL instead of
pg_ctl.
```

```
*
```

Now the initial database environment is setup. The next section will look at verifying the install and setting up users to access the database.

PostgreSQL database setup

Now that PostgreSQL is setup, it's a good idea at this point to verify the installation. First, make sure the service starts up ok:

Code Listing 2.3: Starting up the PostgreSQL service

```
# /etc/init.d/postgresql start
* Starting PostgreSQL ... [ ok ]
```

Once this is verified working, it's also a good idea to add it to the default runlevel so it starts at boot:

Code Listing 2.4: Adding to the default runlevel

```
# rc-update add postgresql default
* postgresql added to runlevel default
```

Now that the service has started, it's time to try setting up a test database. To start out, let's create a test database by using the `createdb` command. We'll also pass along the `-U` option to set the user (it defaults to the current user name if you don't), and the `-W` option to request the password we created earlier. Finally we give it the name of the database we want to create:

Code Listing 2.5: Creating a database with `createdb`

```
$ createdb -U postgres -W test
Password:
CREATE DATABASE
```

The database was successfully created, and we can confirm that the database can run basic tasks. We'll go ahead and drop this database (remove it) with the `dropdb` command:

Code Listing 2.6: Dropping a database with `dropdb`

```
$ dropdb -U postgres -W test
Password:
DROP DATABASE
```

Right now, only the `postgres` user can run commands. Obviously this is not the sort of setup one would like in a multi-user environment. The next section will look at working with user accounts.

Setting up database user accounts

As mentioned earlier, having to login as the `postgres` user is somewhat undesirable in a multi-user environment. In most cases there will be various users and services accessing the server, and each have different permission requirements. So, to handle this, the `createuser` command can be used. This command is an alternative to running a few SQL queries, and is a lot more flexible from an admin standpoint. We'll go ahead and create two users, a 'superuser' that can add other users and administer the db, and a standard user:

Code Listing 2.7: Setting up the superuser

```
(replace chris with the username you'd like to use)
$ createuser -a -d -P -E -U postgres -W chris
Enter password for new user:
Enter it again:
Password:
CREATE USER
```

There, we've created the superuser. The command line option `-a` specifies that this user can add other users. `-d` means that this user can create databases. `-P` let's you enter a password for the user and `-E` will encrypt it for security purposes. Now then, we'll test this new user's permissions out by setting up our standard user:

Code Listing 2.8: Setting up the standard user

```
(replace chris with the username you've just created)
$ createuser -A -D -P -E -U chris -W testuser
Enter password for new user:
Enter it again:
Password:
CREATE USER
```

Success! Our new user was created using the previously created superuser. The `-A` and `-D` options do the opposite of `-a` and `-d`, and instead deny the user the ability to create other users and databases. Now that there are users to work with, the next chapter will look at using the new database.

3. Using PostgreSQL

Setting up permissions

Now there is a user that can create databases and add other users, and the main postgres user that can do anything. The user created earlier can currently login to the server, and that's about it. In general, users need to be able to insert data and retrieve data, and sometimes any other number of tasks. So, for this new user to be able to do anything, they must be setup with the proper permissions. This can easily be done by passing the `-O` parameter to `createdb`. We'll start by making a new database, **MyDB** with our superuser that will be owned by the previous testuser:

Code Listing 3.1: Creating the MyDB database

```
$ createdb -O testuser -U chris -W MyDB
Password:
CREATE DATABASE
```

Alright, now we have a new MyDB database, and a testuser that can access it. To test this out, we'll login as the testuser to the new MyDB database. We'll do this with the `psql` program. This program is what's used to connect to the PostgreSQL database from command line. So connect to the new database like so:

Code Listing 3.2: Logging into the MyDB database as the testuser

```
$ psql -U testuser -W MyDB
Password:
Welcome to psql 8.0.4, the PostgreSQL interactive terminal.
```

```
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit
```

```
MyDB=>
```

So, the testuser is now logged into the database, and can begin to initiate some commands. To get a feel for using PostgreSQL, the next section will take a look at some of the basic commands in navigating the psql client.

Basic PostgreSQL commands and creating a table

For those who are used to MySQL, this is somewhat of a definite read. This is where PostgreSQL may get somewhat unique with regards to running commands. To start, here is a list of some commands that will be discussed:

Command	Usage	MySQL Equivalent
<code>\c [connect]</code> <code>[DBNAME] - Connects to another database</code> <code>[USER]</code>		USE DATABASE
<code>\q</code>	Quit the psql client	quit
<code>\i FILE</code>	Run commands from FILE	source FILE
<code>\o [FILE]</code>	Send query results to FILE	INTO OUTFILE, but outputs everything (not just SELECTS)
<code>\d [NAME]</code>	Describe a database or table (as well as other items)	DESC(RIBE)
<code>\db [PATTERN]</code>	List available tables that match PATTERN (all if no pattern is given)	SHOW TABLES

With the exception of `\c[connect]`, all the commands shown will be used later on in the section. So right now the database is empty. That said, we need to insert some data. The first step to inserting data, however, is to put it in a table. Right now there are no tables in the database, so we need to create one. This is done with the CREATE TABLE command. We'll make a table of items. They will contain a Product ID, Description, and price:

Code Listing 3.3: Creating the products table

```
MyDB=> CREATE TABLE products (
MyDB(>   product_id SERIAL,
MyDB(>   description TEXT,
MyDB(>   price DECIMAL
MyDB(> );
NOTICE: CREATE TABLE will create implicit sequence "products_product_id_seq"
for serial column "products.product_id"
CREATE TABLE
```

You can ignore the NOTICE, it's perfectly harmless. Looking at the last line of the function, CREATE TABLE seems to indicate that the command has succeeded. However, let's go ahead and verify that the table was indeed successfully created with the `\d` command:

Code Listing 3.4: Looking at the newly created table

```
MyDB=> \d products
```

Column	Type	Modifiers
product_id	integer	not null default nextval('public.products_product_id_seq'::text)
description	text	
price	numeric	

Indeed the table was successfully created. Now that the table is created, it needs to be populated with data. The next section will look at populating the database with data.

Inserting data into the database

This section will look at the two ways of populating the newly created table with data. First let's look at the most basic command, INSERT:

Code Listing 3.5: INSERT syntax

```
INSERT INTO [tablename] (column1,column2,column3) VALUES (value1,value2,value3)
```

tablename contains the name of the table to insert the data into. (column1,column2,column3) lets you specify the specific columns to insert the values into. VALUES(value1,value2,value3) is the listing of values. The values are inserted into the same order as the columns (column1 gets value1, column2 gets value2, column3 gets value3). These counts must be the same. So let's go ahead and insert an item into the table:

Important: From working with databases for a long time, I personally recommend specifying INSERT statements exactly as above. Developers often make the mistake of using INSERT INTO without specifying columns. This is unproductive, as if a new column gets added to the database, it will cause an error if the value to column count is not the same. You should always specify the columns unless you're 300% sure you'll never add a column.

Code Listing 3.6: Inserting data into the table

```
MyDB=> INSERT INTO products (description,price) VALUES('A test product',
12.00);
INSERT 17273 1
```

The last line needs a bit of explaining. The return of an insert command is an OID (Object Identifier) and the number of rows inserted. OID's are a bit beyond the scope of this guide, and the [PostgreSQL manual](#) has some good information on it. Now, for a situation where you have 20,000 products, these insert statements can be a little tedious. However, not all is lost. The COPY command can be used to insert data into a table from a file or stdin. In this example, let's assume that you have a csv (comma separated values) file, which contains the product id, description, and price. The file looks like this:

Code Listing 3.7: products.csv

```
2,meat,6.79
3,soup,0.69
```

```
4,soda,1.79
```

Now we'll use the COPY command to populate our data:

Important: The COPY FROM STDIN command is used because only the postgres user can insert data from a file (for obvious security reasons).

Code Listing 3.8: Using COPY to populate the products table

```
MyDB=> COPY products FROM STDIN WITH DELIMITER AS ',';
Enter data to be copied followed by a newline.
End with a backslash and a period on a line by itself.
>> 2,meat,6.79
>> 3,soup,0.69
>> 4,soda,1.79
>> \.
```

Unfortunately, this line doesn't return the same status information as the INSERT INTO statement. How do we know the data was inserted? The next section will look at running queries to check our data.

Using PostgreSQL queries

This section will look at using the SELECT statement to view data in our tables. The basic SELECT format looks like this:

Code Listing 3.9: SELECT syntax

```
SELECT (column1,column2|*) FROM (table) [WHERE (conditionals)]
```

There are two ways to select columns. The first is using * to select all columns, and the second is to specify a list of specific columns you wish to see. The second is quite handy when you want to find a specific column in a rather large list of them. Let's start out with using SELECT with * to specify all columns:

Code Listing 3.10: Viewing the products table

```
MyDB=> SELECT * FROM products;
 product_id | description | price
-----+-----+-----
          1 | A test product | 12.00
          2 | meat          |  6.79
          3 | soup          |  0.69
          4 | soda          |  1.79
(4 rows)
```

As shown here, all the data we inserted earlier is indeed in the table. Now let's say we only want to see the description and the price, and don't care about the product id. In this case we'll use the column specific SELECT form:

Code Listing 3.11: Viewing specific columns from the products table

```
MyDB=> SELECT description,price FROM products;
 description | price
-----+-----
A test product | 12.00
meat          |  6.79
soup          |  0.69
soda          |  1.79
```

```
(4 rows)
```

Now only the product and price is shown, letting us focus on only the important data. Now let's say that we want to see only the items that are greater than \$2.00. Here's where the WHERE clause comes in handy:

Code Listing 3.12: Viewing specific rows from the products table

```
MyDB=> SELECT description,price FROM products WHERE price > 2.00;
description  | price
-----+-----
A test product | 12.00
meat         |  6.79
(2 rows)
```

Now a listing of products over \$2.00 is displayed, focusing the data even more. These forms of querying for information are very powerful, and can help create extremely useful reports.

Conclusion

This concludes the PostgreSQL Guide. A big thanks goes to Masatomo Nakano, the previous Gentoo PostgreSQL maintainer for his help in answering my questions. Any suggestions on this guide should be sent to Chris White. For more extensive documentation, see the [PostgreSQL website](#).

Gentoo и системы контроля версий.

Gentoo и Subversion.

HOWTO Subversion сервер при помощи Apache2 и WebDav

Ссылка на оригинал: http://ru.gentoo-wiki.com/HOWTO_Subversion_сервер_при_помощи_Apache2_и_WebDav

С версии: 1.5

О чем эта статья

Данное HOWTO является вольным переводом оригинальной английской статьи. Также в статью добавлено некоторое количество отсебятины по поводу и без (надеюсь автор оригинала не знает русского и не увидит как я испохабил его несчастную статью).

Данная статья посвящена настройке **сервера** Subversion **под Apache** с применением **WebDav** и только ей. Информацию по клиенту и настройке автономного сервера можно найти по ссылке [en:HOWTO Subversion](#)

Предупреждение: Данная статья предусматривает, что читатель - разумный человек и не будет вешать репозиторий не на ssl хост, и вообще заботится о безопасности.

FIXME: Хорошо бы описать в чем плюсы WebDav подхода и в чем минусы

Настройка WebDav глобально, безотносительно в virtual-хостах

Apache. Компиляция без Worker MPM

Существует известная проблема при больших коммитах (commit, официальный перевод - "фиксация изменений", на мой взгляд длинновато) связанная с использованием апачевого (от Apache) "worker" MPM. Рекомендуется просто отключить этот MPM и скомпилировать Apache с "prefork" MPM. Для этого просто добавьте новую (или поменяйте существующую, если таковая уже есть) строку в package.use (в случае использования Gentoo, естественно):

Файл: /etc/portage/package.use

```
www-servers/apache -mpm-worker mpm-prefork
```

После чего перекомпилируйте Апач: **emerge -aDNtuv apache**

Более того, пожалуйста убедитесь, что subversion скомпилирована с (use flag) **apache2** и без **nowebdav** юзами (от use flags).

Включение SVN и WebDav модулей в Apache

Для использования WebDav и Subversion модулей их, как ни странно, необходимо включить. Делается это при помощи передачи соответствующих опций (далее - дефайнов, от Define) Апачу при загрузке. Также, для аутентификации, вам возможно понадобится SSL (если вам так не кажется, советую подумать еще раз).

Добавьте следующую строку после уже существующей APACHE2_OPTS строки. (-D SVN_AUTHZ необходимо для использования авторизации, см. ниже)

Файл: /etc/conf.d/apache2

```
APACHE2_OPTS="$APACHE2_OPTS -D SVN -D SVN_AUTHZ -D DAV -D SSL"
```

Создание репозиториев применительно к данной конфигурации

Необходимо, чтобы Апач имел права на запись на папку репозитория(иев), с которым(и) он должен работать. Это можно сделать двумя способами:

- Сменой пользователя/группы папки репозитория:
 1. Воспользуйтесь командой **chown apache:apache /путь/к/репозиторию/ -R**
- Добавлением пользователя apache (или того, под которым ваш Апач бежит) в группу svnusers:
 1. Для начала надо такую группу создать: **groupadd svnusers**
 2. Теперь надо добавить пользователя apache в новосозданную группу: **usermod -G svnusers -a apache**
 3. Ну а теперь надо поменять группу папки репозитория на новосозданную: **chgrp svnusers /var/svn/repos/test -R**
 4. Осталось всего ничего - сделать то, ради чего все это было нужно - дать доступ на запись группе svnusers: **chmod g+w /var/svn/repos/test -R**

Лично я, переводчик, рекомендую воспользоваться комбинацией этих способов, из соображений, что права на запись могут понадобится не только Апачу, а включать страждущего пользователя в группу apache тоже как-то нехорошо, т.к. это даст доступ не только к репозиториям. Для упрощения этой задачи мною был написан следующий скрипт:

Файл: create_repository.sh

```
#!/bin/sh

DIR_MODE=770
FILE_MODE=660
SVN_ROOT="/var/svn"

SVN_USER="apache"
SVN_GROUP="svn"

NAME=${1:? "Usage: create_repository.sh <repository name>"}

REPO_DIR="${SVN_ROOT}/${NAME}"

svnadmin create "${REPO_DIR}"
chown "${SVN_USER}:${SVN_GROUP}" -R "${REPO_DIR}"
find "${REPO_DIR}" -type d -exec chmod ${DIR_MODE} {} +
find "${REPO_DIR}" -type f -exec chmod ${FILE_MODE} {} +

# vim: nobackup
```

Базовая настройка

Вместе с subversion устанавливается файл `/etc/apache2/modules.d/47_mod_dav_svn.conf` (в случае включенного юза **apache2**), в котором и содержится практически работающий конфиг (от `config` - настройка).

Инструкция

```
<Location /svn>
    DAV svn
```

сообщает Апачу о том, что все запросы, чей путь начинается с `/svn` необходимо обрабатывать при помощи WebDav модуля (`Dav svn` директива).

- Если вам достаточно только одного репозитория то используйте директиву `SVNPath` /путь/к/репозиторию
- В противном случае, для многочисленных репозиториях, необходимо использовать следующую директиву:

```
SVNParentPath /путь/к/папке/с/репозиториями
```

В этом случае *все подпапки* папки, указанной в `SVNParentPath` директиве, *будут рассматриваться как репозитории Subversion* (Даже если они таковыми не являются).

Также существует возможность заставить Апач формировать список репозиториях/содержимого в них, в случае захода с обычного браузера. Для этого существует магическая опция `SVNListParentPath` (Правда данная опция доступна только с subversion версии 1.3, но для настоящего гентушника это все равно уже антиквариат), которую необходимо установить в *on*:

```
SVNListParentPath on
```

ВАЖНО: При использовании `SVNListParentPath` может так случиться, что вы получите **403 Forbidden**, а не список репозиториях (Более подробно данный природный феномен описан [здесь](#)). Как вариант избавления, вы можете добавить в описание блока *Location* заключительный слэш.

```
<Location /svn/>
```

В таком случае, вам наверняка также захочется, чтобы Апач также отлавливал и просто путь `/svn`. Для этого можно воспользоваться магической опцией `RedirectMatch` (добавив ее куда-нибудь ввне блока *Location*), как показано в нижеприведенном заклинании.

```
RedirectMatch ^(/svn)$ $1/
```

Дабы избежать проблем, вроде описанной тут: [Resource cannot be created at the destination...](#), будет полезно добавить в конфиг директиву:

```
SVNAutoVersioning On
```

Аутентификация

Стандартная

На данный момент мы, чисто теоретически, имеем работающий Апач с подключенным Сабвершеном (от Subversion). Однако любая подозрительная личность может сделать с нашим репозиторием что захочет, что естественно не входит в наши планы.

Добавление следующих строчек в `/etc/apache2/modules.d/47_mod_dav_svn.conf` включит базовую аутентификацию Апача

Файл: `/etc/apache2/modules.d/47_mod_dav_svn.conf`

```
AuthType Basic
AuthName "Subversion repository"
AuthUserFile /var/svn/conf/svnusers
Require valid-user
# Тип встроенной аутентификации "Basic" посылает пароли открытым
текстом, что есть небезопасно. Чтобы избежать ситуации кражи пароля
необходимо добавить следующую директиву, которая будет требовать подключения
с ssl
SSLRequireSSL
```

Примечание: Как несложно догадаться, для того чтобы можно было требовать ssl, его неплохо было бы включить. Как это сделать было описано выше.

Дополнение: Однако намного более завлекательным вариантом является принудительный редирект на ssl хост, т.к. не вводит среднестатистического пользователя в ступор при виде непонятной ошибки на иностранном языке

Начало общеприимского лирического отступления

Также положительным побочным действием предложенного решения является то, что если у вас стандартный ssl порт уже занят, то таким образом можно ликвидировать проблему с запоминанием номера порта, просто заходя на http, а Апач вас уже перешлет на правильный порт (при условии, что вы ему скажите это делать, естественно - болезненной самостоятельностью юниксовых сервера, по счастью, не обладающей (хотя периодически и кажется, что они проявляют признаки очень вредного интеллекта, подкидывая все новые и новые гадости)))

Конец лирическому отступлению

Файл: `/etc/apache2/modules.d/47_mod_dav_svn.conf`

```
<Location /svn>
[...]
```

```
RewriteEngine On
RewriteCond %{HTTPS} !on
RewriteRule ^/(.*) https://%{SERVER_NAME}%{REQUEST_URI} [R]
```

Возможность: Чтобы ограничить доступ только по ssl, недостаточно директивы `SSLRequireSSL` в `47_mod_dav_svn.conf`. Для этого необходимо перенести все директивы из `47_mod_dav_svn.conf` в файл с дефолтным (от default - по умолчанию) виртуальным хостом для ssl - `XX_mod_ssl_default-vhost.conf`. Также необходимо так поменять номер этого виртуального хоста так чтобы он грузился

после `mod_dav` (все файлы в папке `/etc/apache2/conf/modules.d` загружаются в алфавитном порядке). Впрочем это необходимо только тем у кого настолько древний Апач (или настолько новый, кто знает - постоянно туда-сюда эти хосты путешествуют. По моему мнению, их вообще выкинуть можно), что дефолтный хост лежит все еще в `modules.d` - их уже месяцев пять как переместили в `vhosts.d`. Остальным беспокоится не за чем - `vhosts.d` загружаются после `modules.d`

Небольшая оптимизация: Однако вместо того чтобы перемещать директивы из файла `47_mod_dav_svn.conf` внутрь виртуального ssl-хоста, можно попросту проинклудить (от `include` - включать (в себя)) туда целиком файл, предварительно поменяв ему расширение на что-нибудь отличное от `.conf`, либо вынеся вонне `{{filename|modules.d}}`, дабы самостоятельно Апач его не разыскал. Делается это при помощи магической директивы `Include` с указанием абсолютного пути до желаемого файла

Начало лирического отступления о культурных особенностях индейцев
Вообще-то можно указать и относительный, но стоит учитывать, что Апач - индеец и имеет несколько свои представления о том что такое "относительный путь". Для него это путь относительно истинного пути - путь `ServerRoot'a` - т.е. по умолчанию откуда-то от `/usr/lib/apache2`

Конец лирическому отступлению

Оптимизация побольше и по-глобальнее: По-умолчанию Апач грузит все конфиги в папке `{{filename|modules.d}}` по маске `{{filename|modules.d}/*.conf}`, так что можно немного переписать систему инклюдов, дабы разделить конфиги на обычные и продвинутые-ssl-ные. Для этого достаточно переименовать полюбившийся конфиг, в нашем случае - `47_mod_dav_svn.conf`, в что-нибудь с расширением отличным от `.conf` (либо, как уже было предложено выше, создав отдельную папку для подобных продвинутых конфигов), к примеру в `47_mod_dav_svn.ssl.conf`. Тогда достаточно будет указать в `httpd.conf` (или любом другом глобальном конфиге - к примеру в `modules.d/40_mod_ssl.conf`), внутри соответствующего блока `<IfModule ssl_module><IfDefine SSL>` следующие строки:

```
# Загрузить модули зависщие от ssl
                        Include /etc/apache2/modules.d/*.ssl-conf # либо
/etc/apache2/modules.ssl.d/*.conf
```

Анонимный доступ: Хорошо, мы добились того, что теперь при каждом запросе к `subversion` будет спрашиваться пароль, однако вам может захотеться, чтобы анонимные пользователи могли читать ваш труд, а то зачем все это надо? В таком случае директиву `Require valid-user` стоит поместить в блок `<LimitExcept ...>` указав в качестве параметра список операций, которые, по вашему мнению, всякие подозрительные личности без особой опасности могут производить на вашем репозитории:

```
[...]
AuthType Basic
[...]
# Для любых операций, кроме нижеперечисленных, спрашивать пароль.
<LimitExcept GET PROPFIND OPTIONS REPORT>
    Require valid-user
</LimitExcept>
```

В любом случае, для правильной работы `subversion` клиента с `ssl` в дефолтный

ssl-хост `XX_mod_ssl_default-vhost.conf`, либо в `40_mod_ssl.conf`, **необходимо** добавить следующую строчку (подробности - [тут](#))

```
BrowserMatch "SVN" redirect-carefully
```

Упомянутый выше файл `/var/svn/conf/svnusers` и содержит имена и зашифрованные пароли пользователей.

Чтобы создать этот файл/добавить в него новую запись воспользуйтесь командой `htpasswd2`.

Для создания файла с первым пользователем:

```
htpasswd2 -c /var/svn/conf/svnusers <имя первого пользователя>
```

и для добавления последующих:

```
htpasswd2 /var/svn/conf/svnusers <имя последующего пользователя>
```

Основанная на PAM

Если управление аж двумя различными файлами с паролями (системного и Апачевого) для вас слишком утомительно, то специально для вас был придуман модуль `mod_auth_pam`, который использует PAM для аутентификации пользователей. Если у вас нет еще этого модуля - **`emerge mod_auth_pam`**

Предупреждение: Да будет вам известно, что `mod_auth_pam` представляет из себя потенциальную уязвимость: системный пользователь 'apache' (или тот под которым запущен ваш сервер) получит доступ на чтение к `/etc/shadow`. Это означает, что в случае взлома Апача, злоумышленник получит доступ к файлу `/etc/shadow`, и сможет натравить на него программу по взлому паролей (дальнейшая судьба системы будет зависеть от того, заметите ли вы взлом и примите меры раньше, чем злоумышленник получит пароль рута).

После сборки `mod_auth_pam` необходимо внести несколько изменений (суперпользователем) в ваши конфигурационные файлы. Для начала надо выдать пользователю 'apache' доступ на чтение `/etc/shadow`:

1. **`groupadd shadow`**
2. **`gpasswd -a apache shadow`**
3. **`chgrp shadow /etc/shadow`**
4. **`chmod g+r /etc/shadow`**

Предупреждение: Если у вас есть веб-сервер на котором работают посторонние люди, и он работает под тем же пользователем 'apache', то эти самые посторонние люди без всякого взлома смогут прочесть ваш `/etc/shadow`, и это уже **действительно опасно!**.

После чего внесите следующие изменения в `/etc/apache2/modules.d/47_mod_dav_svn.conf`: Вовнутрь блока `<Location /svn>` впишите:

Файл: `/etc/apache2/modules.d/47_mod_dav_svn.conf`

```
<IfModule mod_auth_sys_group.c>
    AuthPAM_Enabled on
    AuthType Basic
    AuthName "Subversion Repository"
    SSLRequireSSL
    Require group subversion
</IfModule>
```

ВАЖНО: В отличие от стандартной ('basic') аутентификации, описанной выше, убедитесь что вы **не** указали директивы

```
AuthUserFile /var/svn/conf/svnusers
```

Впрочем существует возможность оставить стандартную аутентификацию как запасную на случай если PAM модуль по каким-то причинам не сработает. Для этого обратите внимание на файл [/etc/apache2/modules.d/10_mod_auth_pam.conf](#)

ВАЖНО: Как уже упоминалось выше базовый тип встроенной аутентификации (*AuthType Basic*) передает пароли открытым текстом и для обеспечения минимальной безопасности необходимо ограничить аутентификацию только по ssl, при помощи директивы *SSLRequireSSL*

Ну а для того, чтобы все, в пределах этого параграфа, проделанное имело смысл, необходимо включить *mod_auth_pam*, добавив *-D AUTH_PAM* в *APACHE2_OPTS* в [/etc/conf.d/apache2](#)

Active Directory/LDAP

Для того, чтобы настроить авторизацию в SVN через LDAP или Active Directory, следуйте следующим инструкциям.

В [/etc/conf.d/apache2](#) добавляем *-D LDAP* и *-D AUTH_LDAP*

```
APACHE2_OPTS="-D SSL -D PHP4 -D DAV -D DAV_FS -D SVN -D DAV_SVN -D LDAP -D AUTH_LDAP"
```

вот пример моей рабочей конфигурации Apache для авторизации через Active Directory: (нужно изменить путь к *ldap* на тот, что вы хотите использовать)

```
<IfDefine SVN>
<Location /svn/repo>
DAV svn
SVNPath /var/svn/repo
AuthType Basic
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthName "Authorize Me"
AuthLDAPURL
ldap://domain.com:389/OU=IT,OU=MainOffice,OU=Locations,OU=Corporate,DC=domain
name,DC=com?samAccountName?sub?(objectCategory=person)
AuthLDAPBindDN "CN=webuser,OU=Resources,OU=Corporate,OU=AOM,DC=domain,DC=com"
AuthLDAPBindPassword xxxxxxxxxxxx
Require valid-user
</Location>
</IfDefine>
```

Для того, чтобы разрешить доступ только ограниченному списку пользователей:

```
<Location "/useraccess">
AuthName "user permissions"
require user larry bill sam
</Location>
```

Для того, чтобы разрешить доступ только ограниченному списку групп:

```
<Location "/group">
AuthName "group permissions"
require group cn=Group,cn=Users,dc=domainname,dc=com
</Location>
```

В случае возникновения проблем с запросами к LDAP, запустите следующую команду на вашем Windows-сервере:

```
ldifde -f mydomain.ldif -s 127.0.0.1
```

Она сбросит все содержимое LDAP в указанный файл.

Авторизация

Если обычной авторизации недостаточно, можно расширить ее списками доступа. Имеется в виду что базовая аутентификация уже настроена и мы будем раздавать или запрещать доступ отдельным пользователям. Управлять списком можно через специальный файл, [SVNAccessFile](#):

Файл: SVNAccessFile

```
AuthzSVNAccessFile /var/svn/conf/svnpolicy
```

Директива, указанная выше включает возможность создания списков доступа (ACL) для работы с репозиториями subversion через apache.

Правила авторизации для определенного репозитория задаются примерно так:

Файл: SVNAccessFile

```
[test:/trunk]
testuser1 = rw
testuser2 =
* = r
```

Описанное выше правило разрешает производить пользователю **testuser1** запись (commit, сору и т.д.) в репозиторий **test** (папка /trunk). Пользователь же **testuser2** не имеет доступа к репозиторию. Остальные пользователи (маска *) имеют доступ только для чтения (могут производить checkout).

Для того, чтобы **apache** был в курсе наших опытов, нам нужно отредактировать [/etc/conf.d/apache2](#):

Файл: /etc/conf.d/apache2

```
APACHE2_OPTS="... -D SVN_AUTHZ"
```

И напоследок, несколько замечаний по работе с правилами доступа: будьте осторожны с пробелами - внутри скобок не должно быть ни одного; при использовании только одного репозитория, его название можно не указывать, достаточно указать только папку.

Наводим марафет

В последней ревизии репозитория можно полазать в любимом браузере. Удовольствия это много не доставит - выглядит все это дело не очень красиво.

Чтобы это исправить, добавьте директиву

```
SVNIndexXSLT /svnindex.xsl
```

внутри тега `<Location /svn>` и укажите свой xsl-файл. Если же с xml вы знакомы недостаточно хорошо, попробуйте xsl и css -файлы, входящие в стандартный пакет **subversion** в папке `/usr/share/doc/subversion-<версия_пакета>`.

Некоторые браузеры (например, Opera), вместо нормального отображения XSL-файла, могут показывать пустую страницу. Чтобы этого избежать, можно пропустить XSL через **xsltproc**, чтобы он преобразовал это дело в обычный html. Для этого необходимо установить пакет **xsltproc**:

```
emerge libxslt
```

И добавить следующее в файл настроек **apache**:

```
BrowserMatch "Opera" xsltfilter
```

```
ExtFilterDefine xslt mode=output enableenv=xsltfilter \  
    intype=text/xml outtype=text/html \  
    cmd="/usr/bin/xsltproc /var/www/svnindex.xsl -"
```

И добавить следующее внутри тега `<Location /svn>`:

```
SetOutputFilter xslt  
AddDefaultCharset utf-8
```

Полезные ссылки

- [Книга по Subversion](#)
- [Документация по mod_auth_ldap](#)

Руководство по udev в Gentoo

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/udev-guide.xml>

С версии: 1.0

1. Что такое udev?

Каталог /dev

Когда пользователи Linux разговаривают о начинке своих систем среди людей, полагающих, что Linux — это какой-то вид вируса или марка кофе, пользуясь оборотами вроде «слэш дев слэш что—нибудь» (/dev/foo), на них наверняка смотрят косо. Но для счастливицков (естественно, включая вас), /dev/hda1 — просто быстрый способ упоминания первого раздела ведущего диска IDE на первичном канале. Разве нет?

Все мы знаем, что такое файл устройства. Некоторые даже знают, почему рядом с ними видны специальные числа, если посмотреть на результат `ls -l`, запущенной в каталоге /dev. Но мы принимаем за данность, что первичный ведущий диск IDE называется /dev/hda. У вас может быть и не так, но это всего лишь пример.

Представим себе устройства «горячей замены» вроде USB, IEEE1394, заменяемой PCI и т.д. Какое из них является первым по счету? Надолго ли? Как изменятся названия других устройств, если первое отключить? Как это повлияет на текущие операции? Было бы забавно, если бы задание печати вдруг переключалось с вашего супер навороченного лазерного принтера на старый полудухлый матричный из-за того, что ваша мама решила выдернуть разъем лазерного принтера, а тот оказался первым в списке, не так ли?

Перейдем к udev. Цели проекта udev как занимательны, так и актуальны:

- запуск драйверов в режиме пользователя
- динамическое создание и удаление файлов устройств
- согласованное назначение имен устройств
- предоставление интерфейса программирования (API) для режима пользователя

Разработка udev состоит из трех отдельных проектов: `namedev`, `libsfs` и, конечно, `udev`.

namedev

`Namedev` позволяет отделить именование устройств от программы `udev`. Это дает возможность гибкого использования правил и схем именования, разработанных отдельно. Подсистема именования предоставляет стандартный интерфейс, который может использоваться `udev`.

Сейчас, через `namedev` представлена только одна схема именования, предоставляемая LANANA. Она используется на большинстве Linux-систем, и

поэтому хорошо подходит основной массе пользователей Linux.

В `namedev` для определения имени конкретного устройства используется последовательность из пяти шагов. Если на одном из шагов имя установлено, оно и используется. Вот эти шаги:

1. по метке или серийному номеру
2. по номеру устройства шины
3. по топологии шины
4. статическое имя
5. имя, назначенное в ядре

По метке или серийному номеру — проверяется, есть ли у устройства уникальный идентификатор. Например, у устройств USB есть уникальный серийный номер USB, а у устройств SCSI — уникальный UUID. Если `namedev` находит соответствие уникального номера и определенного конфигурационного файла, используется имя, указанное в конфигурационном файле.

По номеру устройства шины — проверяется номер устройства, подключаемого к шине. В среде, где нет возможности «горячей замены», этого шага достаточно для идентификации устройства (например, номер на шине PCI редко меняется за время существования системы). Опять же, если `namedev` находит соответствие номера устройства шины и определенного конфигурационного файла, используется имя, указанное в конфигурационном файле.

Подобным образом, по топологии шины — это, скорее, статический способ идентификации устройств, подходящий до тех пор, пока пользователь не начинает их переключать. Когда позиция устройства совпадает со значением, заданным пользователем, используется указанное имя.

Четвёртый шаг, статическое имя — это простая подстановка строки. Замененное имя присваивается, когда имя, назначенное в ядре (по умолчанию) совпадает с заданной строкой замены.

Последний шаг (имя, назначенное в ядре) покрывает все остальные случаи: при этом берётся имя по умолчанию, присвоенное в ядре. В большинстве случаев этого достаточно, поскольку имя соответствует принятому в современных Linux-системах.

libsysfs

`udev` взаимодействует с ядром через псевдофайловую систему `sysfs`. Проект `libsysfs` предлагает обобщенный интерфейс программирования для доступа к сведениям, предоставляемым файловой системой `sysfs`. Это даёт возможность опроса разнородных устройств без необходимости знания их типа.

udev

Каждый раз, когда ядро обнаруживает обновление структуры устройств, оно вызывает программу `/sbin/hotplug`. `Hotplug`, в свою очередь, запускает приложения, указанные в каталоге `/etc/hotplug.d/default`, где также находится символическая

ссылка на приложение udev. Hotplug направляет сведения, данные ядром, в программу udev, которая производит необходимые действия над структурой /dev (создавая или удаляя файлы устройств).

2. Как использовать udev в Gentoo

Требования к системе

Файловая система udev должна использоваться вместе с ядром версии 2.6 (пакеты `vanilla-sources` или `gentoo-sources`, и профилем 2005.0). Если вы используете такое ядро, убедитесь, что у вас есть самая последняя версия `sys-apps/baselayout`. Вот и всё, что потребуется.

Листинг 2.1: Установка udev
emerge udev

udev установит пакет `hotplug-base`, как одну из зависимостей. Не нужно устанавливать пакет `hotplug` до тех пор, пока вы хотите, чтобы модули автоматически загружались при подключении устройств. `hotplug` также управляет автоматическим включением сетевых устройств и загрузкой встроенного ПО.

Листинг 2.2: Установка дополнительных сценариев hotplug
emerge hotplug

Если хотите, чтобы загружались модули для устройств, подключаемых до загрузки, используйте пакет `coldplug`:

Листинг 2.3: Установка пакета coldplug
emerge coldplug

И не забудьте добавить `coldplug` на загрузочный уровень запуска (boot):

Листинг 2.4: Добавление coldplug на загрузочный уровень запуска
rc-update add coldplug boot

Что же касается ядра, не забудьте выбрать следующие параметры:

Листинг 2.5: Требуемые параметры ядра
 General setup --->
 [*] Support for hot-pluggable devices
 File systems --->
 Pseudo filesystems --->
 [*] /proc file system support
 [*] Virtual memory file system support (former shm fs)

По вашему усмотрению, можете оставить поддержку файловой системы /dev file system support (OBSOLETE) активной, но обязательно убедитесь в том, что параметр «Automatically mount at boot» выключен:

Листинг 2.6: Автоматически не монтировать devfsd

File systems --->

Pseudo Filesystems --->

[*] /dev file system support (OBSOLETE)

[] Automatically mount at boot

Если вы используете genkernel, не забудьте запустить её с параметром --udev, для включения всех необходимых параметров конфигурации ядра. Конфигурация по умолчанию, даваемая этим обращением к genkernel, достаточна.

Конфигурация

Если вы хотите использовать улучшения udev, добавленные Gentoo для более удобного использования, не читайте дальше. Gentoo будет использовать udev, сохраняя неизменной структуру /dev, так что вы никогда не потеряете специальные файлы устройств. Сценарии инициализации Gentoo не будут запускать демон devfsd и деактивируют devfs при загрузке.

Но если вы «крепкий орешек» и хотите запустить любыми средствами систему только с udev и без улучшений, к чему и стремится разработка udev (включая сложности в отсутствии узлов устройств, потому, что udev их ещё не поддерживает), тогда читайте дальше :)

Мы деактивируем правила, которые сохраняют специальные файлы устройств: отредактируйте переменную RC_DEVICE_TARBALL в /etc/conf.d/rc и установите её значение на no:

Листинг 2.7: /etc/conf.d/rc

RC_DEVICE_TARBALL="no"

Если вы включили поддержку devfs в вашем ядре, вы можете деактивировать её в конфигурации загрузчика: добавьте gentoo=nodevfs как параметр ядра. Если вы хотите использовать devfs и деактивировать udev, добавьте gentoo=noudev.

3. Известные проблемы

Отсутствие специальных файлов устройств при загрузке

Если ваша машина не может успешно загрузиться и выдаётся ошибка о том, что путь /dev/null не найден или отсутствует первоначальная консоль, то проблема в том, что отсутствуют некоторые файлы устройств, которые должны быть доступны до того, как /dev смонтирован и обработан udev. Это часто встречается на машинах, где Gentoo установлен со старых носителей.

Если вы используете sys-apps/baselayout-1.8.12 или более позднюю версию, эта проблема облегчается, так как процесс загрузки должен управлять до полного завершения. Однако, чтобы избавиться от этих раздражающих предупреждений, вам следует создать отсутствующие специальные файлы, как описано далее.

Выполните следующие команды для того, чтобы увидеть, какие специальные файлы присутствуют до обработки udev в /dev:

Листинг 3.1: Вывод списка специальных файлов устройств, доступных при загрузке

```
# mkdir test
# mount --bind / test
# cd test/dev
# ls
```

Устройства /dev/null и /dev/console необходимы для успешной загрузки. Если они не были показаны в предыдущем тесте, вы должны создать их самостоятельно. Выполните следующие команды в каталоге test/dev/:

Листинг 3.2: Создание необходимых специальных файлов устройств

```
# mknod -m 660 console c 5 1
# mknod -m 660 null c 1 3
```

Когда закончите, не забудьте отмонтировать каталог test/:

Листинг 3.3: Отмонтирование каталога test/

```
# cd ../../
# umount test
# rmdir test
```

udev и nvidia

Если вы используете коммерческий драйвер от nVidia и сервер X не запускается только в системе с udev, убедитесь что у вас:

название модуля nvidia указано в файле /etc/modules.autoload.d/kernel-2.6
версия nvidia-kernel больше или равна media-video/nvidia-kernel-1.0.5336-r2
версия baselayout как минимум sys-apps/baselayout-1.8.12

Если xorg-x11 отказывается запускаться, возможно, отсутствует файл устройства /dev/nvidia. В этом случае запустите /sbin/NVmakedevices.sh, чтобы его создать или пересоздать.

Не появляются имена LVM2

Если вы используете вместе udev и Logical Volume Manager 2 (LVM—менеджер логических разделов), вы можете заметить, что созданные вами группы разделов и логические разделы исчезли. Что ж, на самом деле нет, но они, к несчастью, названы /dev/dm-#, где #—это 0,1, ... и т.д.

Чтобы это исправить, отредактируйте файл `/etc/udev/rules.d/50-udev.rules` и уберите комментарий со следующей строки:

Листинг 3.4: Как убрать комментарий со строки из файла `/etc/udev/rules.d/50-udev.rules`
`KERNEL="dm-[0-9]*", PROGRAM="/sbin/devmap_name %M %m", NAME="%k", SYMLINK="%c"`

Далее, установите пакет `sys-fs/multipath-tools`, который содержит программу `devmap_name`.

Листинг 3.5: Установка пакета `multipath-tools`
(На момент написания, пакет `multipath-tools` был доступен только в тестовой ветке дерева Portage)
`# echo "=sys-fs/multipath-tools-0.4.2 ~x86" >> /etc/portage/package.keywords`
`# emerge multipath-tools`

Отличающиеся названия в DevFS и udev

Даже если наша цель—иметь совместимую схему именований между этими динамическими системами управления устройствами, иногда случаются различия в именованиях.

Сообщалось о коллизии в работе контроллера HP Smart Array 5i (более точно - модуль ядра `cciss`). При использовании `udev`, устройства именовались как `/dev/cciss/cXdYpZ`, где X, Y и Z - числа. При использовании `devfs`, устройства именовались `/dev/hostX/targetY/partZ` или являлись символическими ссылками на `/dev/cciss/cXdY`.

В этом случае, не забудьте обновить файл `/etc/fstab` и конфигурационные файлы загрузчика соответственно.

То же самое происходит со всеми символическими ссылками, которые существуют в каталоге `/dev`, такими как `/dev/mouse`, которые `udev` больше не создаёт. Обязательно проверьте настройку графического сервера X и посмотрите секцию `Device`, указывает ли там строка настройки на существующий файл устройства.

Другая проблема заключается в различном именовании терминалов между `devfs` и `udev`. Если в `devfs` терминалы называются `tty`, то в `udev` они называются `vc`. Это может привести к проблемам если возможность входа `root` с консоли ограничивается с помощью `/etc/securetty`. Чтобы обеспечить возможность входа `root` с консоли, нужно обязательно изменить `tty1` на `vc/1` в `/etc/securetty`.

Другие проблемы

Если файлы устройств не создаются, когда соответствующий модуль загружается из `/etc/modules.autoload.d/kernel-2.6`, но появляются когда вы загружаете этот модуль вручную при помощи `modprobe`, тогда вам стоит произвести обновление

до sys-apps/baselayout-1.8.12 или более поздней версии.

Поддержка устройств кадрового буфера (/dev/fb/*) включена в ядро с версии 2.6.6-rc2.

В ядрах старше, чем 2.6.4, вам следует явно включить поддержку для файловой системы /dev/pts.

Листинг 3.6: Включение поддержки файловой системы /dev/pts

```
File systems --->
```

```
Pseudo filesystems --->
```

```
[*] /dev/pts file system for Unix98 PTYs
```

4. Ссылки на ресурсы и благодарности

Разговор об udev начался на симпозиуме по Linux в Оттаве в 2003 году (Linux Symposium, Ottawa, Ontario Canada - 2003) Грегом Кроа-Хартманом (Greg Kroah-Hartman) из корпорации IBM, который дал ясное понимание для применения udev.

Decibel's UDEV Primer - документ о udev и Gentoo.

Написание правил для udev члена сообщества разработки Gentoo Дэниела Дрэйка (Daniel Drake) - отличный документ для того, чтобы научиться управлять установкой udev.

Gentoo LDAP-DNS Guide (Англ.)

Ссылка на оригинал: <http://www.gentoo.org/doc/en/ldapdns-guide.xml>

С версии: 1.0

1. Introduction

LDAPDNS is a small server that provides DNS services to your network. With DNS services, you can manage your hostnames in a central manner deprecating the tedious /etc/hosts updates every time a system is added/removed or has changed its host name.

The LDAPDNS package uses an LDAP service to store all DNS records (which is, simply explained, a line that contains the mapping between a hostname and an IP address). LDAP is a standard protocol to obtain information from a hierarchically represented knowledge base (directories). The most well-known LDAP service for Linux is OpenLDAP, a free LDAP implementation.

With this small guide at hand, you should be able to set up DNS services on your network with as little effort as possible.

2. Configuring LDAP-DNS

First, install net-dns/ldapdns.

Code Listing 2.1: Installing ldapdns

```
# emerge net-dns/ldapdns
```

Next, configure ldapdns to host the DNS records for your network. We use ldapdns-conf which uses the following syntax:

Code Listing 2.2: Syntax for ldapdns-conf

```
ldapdns-conf acct logacct /path yourip ldaphost dn [suffix]
```

Keyword	Explanation	Example

acct	Username as which ldapdns will run	ldapdns

logacct	Username as which the ldapdns logging will run	dnslog

/path	Chrooted home directory for ldapdns	/var/lib/ldapdns

yourip	IP address to listen to	127.0.0.1

ldaphost	Address of the LDAP service	ldapi://%2fvar%2frun%2fopenldap%2fslapd.sock

dn	LDAP login credentials	cn=Manager,dc=yourdomain

suffix	Optional default ldap path to add on all queries	ou=Machines,dc=yourdomain

You should substitute the 127.0.0.1 IP address with an IP address that all hosts can reach and don't forget to use the domain name you want instead of yourdomain.

Code Listing 2.3: Example ldapdns-conf run


```
# ldapdns-conf ldapdns dnslog /var/lib/ldapdns 127.0.0.1 ldapi://%2fvar%2frun%2fopenldap
%2fsldapd.sock cn=Manager,dc=yourdomain ou=Machines,dc=yourdomain
```

Now set the LDAP login password:

Code Listing 2.4: Setting the LDAP login password

```
# echo YourSecretPassword > /var/lib/ldapdns/root/password
# chmod 0400 /var/lib/ldapdns/root/password
```

Now configure ldapdns to use the simple authentication method and the cosine LDAP schemas:

Code Listing 2.5: Configuring ldapdns

```
# echo simple > /var/lib/ldapdns/env/LDAP_AUTH
# echo cosine > /var/lib/ldapdns/env/SCHEMA
```

You can optionally set an e-mail address for the service:

Code Listing 2.6: Setting a host master e-mail address

```
# echo YourMail@address > /var/lib/ldapdns/env/HOSTMASTER
```

Add the service to the supervise scan and run the service supervisor if you haven't started it already:

Code Listing 2.7: Managing supervise services

```
# ln -s /var/lib/ldapdns /service
# /etc/init.d/svscan start
# rc-update add svscan default
```

3. Configuring OpenLDAP

Now we need to configure OpenLDAP with the DNS schema. Open up /etc/openldap/sldap.conf with your favorite editor and make sure the following three lines are listed:

Code Listing 3.1: Editing /etc/openldap/sldap.conf

```
include /etc/openldap/schema/cosine.schema
include /etc/openldap/schema/inetorgperson.schema
include /etc/openldap/schema/nis.schema
```

Bootstrap LDAP with the base dn you defined previously with ldapdns-conf. To accomplish this, we first create a file called bootstrap.ldif (it is just a name) which we fill up with DNS information.

As an example, we provide a bootstrap.ldif file for the fictitious domain

cherchetoujours.org.

Now bootstrap your LDAP with this information:

Code Listing 3.2: Bootstrapping LDAP

```
# ldapadd -x -D "cn=Manager,dc=yourdomain" -W -f bootstrap.ldif
```

4. Testing the Installation

Now you are all set. Test your setup using nslookup (part of net-dns/bind-tools):

Code Listing 4.1: Testing the ldapdns configuration

```
# nslookup ns1.yourdomain
```

Note: nslookup is deprecated and may be removed from future releases.

Consider using the `dig` or `host` programs instead. Run nslookup with the `-sil[ent]` option to prevent this message from appearing.

```
Server:      127.0.0.1
```

```
Address:     127.0.0.1#53
```

```
Name: ns1.example.com
```

```
Address: (Your server IP address)
```

Руководство по настройке Samba в режиме PDC с использованием LDAP

Ссылка на оригинал: http://ru.gentoo-wiki.com/Samba_PDC_LDAP
С версии: 1.2

смотрите также:

[Настройка и использование централизованного управления сервисами с использованием сервера LDAP.](#)

Устанавливаемые пакеты

Установим нужные пакеты:

```
# USE="ldap acl ldapsam pam gdbm samba ssl tcpd winbind" emerge samba openldap  
acl nss_ldap pam_ldap
```

ВАЖНО: Версия nss_ldap должна быть не ниже 250-r1 (>=sys-auth/nss_ldap-250-r1)

OpenLDAP

Конфигурирование OpenLDAP

Для примера используется домен amber.global.com который является дочерним доменом домена global.com, который работает под управлением Win2003 Server.

Примечание: Не решенной задачей осталось создание доверительных отношений между доменами и включение домена amber в лес global.com

Файл: /etc/openldap/slapd.conf

```
include /etc/openldap/schema/core.schema  
include /etc/openldap/schema/cosine.schema  
include /etc/openldap/schema/inetorgperson.schema  
include /etc/openldap/schema/misc.schema  
include /etc/openldap/schema/nis.schema  
include /etc/openldap/schema/openldap.schema  
include /etc/openldap/schema/samba.schema  
pidfile /var/run/openldap/slapd.pid  
argsfile /var/run/openldap/slapd.args  
  
access to dn.base=""  
by self write  
by * auth  
access to attr=userPassword  
by self write  
by * auth
```

```
access to attr=shadowLastChange
        by self write
        by * read
access to *
        by * read
        by anonymous auth

#loglevel 1
database      ldbm
suffix        "dc=amber,dc=global,dc=com"
rootdn        "cn=Manager,dc=amber,dc=global,dc=com"
# Пароль rootpw лучше всего указывать в зашифрованном виде.
# Для генерации зашифрованного пароля используйте утилиту slappasswd
# Например: slappasswd -h {MD5}
# rootpw = secret
rootpw        {MD5}Xr4il0zQ4PCOq3aQ0qbuaQ==
directory     /var/lib/openldap-ldbm

index objectClass      eq
index cn                eq,subinitial
index sn                eq,subinitial
index uid               eq,subinitial
index displayName      eq,subinitial
index uidNumber         eq
index gidNumber         eq
index memberUID         eq
index sambaSID          eq
index sambaPrimaryGroupSID eq
index sambaDomainName   eq
```

Файл: /etc/openldap/ldap.conf

```
HOST 127.0.0.1
BASE dc=sanaa,dc=global,dc=com
```

Проверяем разрешения на каталоги **/var/lib/openldap-***
`ls -la /var/lib/`

ВАЖНО:

drwx-----	2	ldap	ldap	104	Июн	8 18:31	openldap-data
drwx-----	2	ldap	ldap	72	Июн	8 18:31	openldap-ldbm
drwx-----	2	ldap	ldap	72	Июн	8 18:31	openldap-slurp

Запуск сервера OpenLDAP

Пока работаем без шифрования трафика, так как сервер LDAP и SAMBA работают на одном сервере, подправляем конфиг чтобы LDAP слушал 389 порт только на localhost.

Файл: /etc/conf.d/slaped

```
# conf.d file for the openldap-2.1 series
#
# To enable both the standard unciphared server and the ssl encrypted
# one uncomment this line or set any other server starting options
# you may desire.
```

```
#  
# OPTS="-h 'ldaps:// ldap:// ldapi://%2fvar%2frun%2fopenldap%2fslapd.sock'"  
OPTS="-h 'ldap://127.0.0.1'"
```

и запускаем OpenLDAP

```
/etc/init.d/slaped start
```

SAMBA

Миграция OpenLDAP

Существует 2 пути создания учетных записей пользователей: домен уже существует и создание нового домена. В первом случае самбу нужно настроить в режиме BDC, перенести все учетные записи, и потом изолировав PDC, перезапустить самбу в режиме PDC. Во втором случае все еще проще, самбу сразу запускаем в режиме PDC и создаем стандартные учетные записи при помощи замечательного пакета smbldap-tools.

emerge	smbldap-tools
/etc/init.d/samba	start

Code: **configure.pl**

```
=====
smbldap-tools script configuration
=====
Before starting, check
. if your samba controller is up and running.
. if the domain SID is defined (you can get it with the 'net getlocalsid')

. you can leave the configuration using the Ctrl-c key combination
. empty value can be set with the "." character
=====
Looking for configuration files...

Samba Configuration File Path [/etc/samba/smb.conf] >
The default directory in which the smbldap configuration files are stored is
shown.
If you need to change this, enter the full directory path, then press enter
to continue.
Smbldap-tools Configuration Directory Path [/etc/smbldap-tools/] >
=====
Let's start configuring the smbldap-tools scripts ...

. workgroup name: name of the domain Samba act as a PDC
workgroup name [amber] >
. netbios name: netbios name of the samba controler
netbios name [neptun] >
. logon drive: local path to which the home directory will be connected
(for NT Workstations). Ex: 'H:'
logon drive [U:] >
. logon home: home directory location (for Win95/98 or NT Workstation).
(use %U as username) Ex: '\\neptun\%U'
logon home (press the "." character if you don't want homeDirectory) [\\%L\
users\%U] >
. logon path: directory where roaming profiles are stored.
Ex: '\\neptun\profiles\%U'
logon path (press the "." character if you don't
want roaming profile) [\\%L\Profiles\%a\%U] >
```

```

. home directory prefix (use %U as username) [/home/%U] >
. default users' homeDirectory mode [700] >
. default user netlogon script (use %U as username) [] >
  default password validation time (time in days) [45] > 900
. ldap suffix [dc=amber,dc=global,dc=com] >
. ldap group suffix [ou=Groups] >
. ldap user suffix [ou=Users] >
. ldap machine suffix [ou=Users] >
. Idmap suffix [ou=Idmap] >
. sambaUnixIdPool: object where you want to store the next uidNumber
  and gidNumber available for new users and groups
  sambaUnixIdPool object (relative to ${suffix}) [sambaDomainName=amber] >
. ldap master server: IP adress or DNS name of the master (writable) ldap
server
  ldap master server [127.0.0.1] >
. ldap master port [389] >
. ldap master bind dn [cn=Manager,dc=amber,dc=global,dc=com] >
. ldap master bind password [] >
. ldap slave server: IP adress or DNS name of the slave ldap server: can also
  be the master one
  ldap slave server [127.0.0.1] >
. ldap slave port [389] >
. ldap slave bind dn [cn=Manager,dc=amber,dc=global,dc=com] >
. ldap slave bind password [] >
. ldap tls support (1/0) [0] >
. SID for domain amber: SID of the domain (can be obtained with 'net
getlocalsid neptun')
  SID for domain amber [S-1-5-21-1918777035-593721947-2697221154] >
. unix password encryption: encryption used for unix passwords
  unix password encryption (CRYPT, MD5, SMD5, SSHA, SHA) [SSHA] > MD5
. default user gidNumber [513] >
. default computer gidNumber [515] >
. default login shell [/bin/bash] >
. default skeleton directory [/etc/skel] >
. default domain name to append to mail adress [] >
=====
backup old configuration files:
  /etc/smbldap-tools/smbldap.conf->/etc/smbldap-tools/smbldap.conf.old
  /etc/smbldap-tools/smbldap_bind.conf->/etc/smbldap-
tools/smbldap_bind.conf.old
writing new configuration file:
  /etc/smbldap-tools/smbldap.conf done.
  /etc/smbldap-tools/smbldap_bind.conf done.

```

Если мы не хотим, что бы у всех пользователей профили были перемещаемыми, то в файле `/etc/smbldap-tools/smbldap.conf` установим следующее значение:

Файл: `/etc/smbldap-tools/smbldap.conf`

```

...
userProfile=""
...

```

Инициализируем каталоги самбы в LDAP:

```
smbldap-populate -a Administrator -k 0 -m 0
```

Конфигурация SAMBA

Файл: /etc/samba/smb.conf

```
[global]
    workgroup = amber
    netbios name = neptun
    realm = amber.global.com
    nt acl support = yes
    acl compatibility = win2k
    map acl inherit = yes
    server string = Samba Server %v
    interfaces = eth0
    bind interfaces only = yes
    hosts allow = 192.168.7. 127.
    log file = /var/log/samba/log.%m
    debug level = 9
    max log size = 500
    socket options = TCP_NODELAY SO_SNDBUF=8192 SO_RCVBUF=8192
    security = user
    os level = 250
    passdb backend = ldapsam:"ldap://127.0.0.1/"
    enable privileges = yes

    passwd program = /usr/sbin/smbldap-passwd "%u"
    passwd chat = *new*password* %n\n *new*password* %n\n *successfully*
    passdb expand explicit = no
    unix password sync = no
    ldap passwd sync = no

    ldap suffix = dc=amber,dc=global,dc=com
    ldap admin dn = cn=Manager,dc=amber,dc=global,dc=com
    ldap user suffix = ou=Users
    ldap group suffix = ou=Groups
# Т.к. для самбы компьютеры и пользователи - одно и то же,
# и искать она в дальнейшем записи компьютеров будет в пользователях,
# то для избежания дальнейших проблем при добавлении рабочих станций
# к домену мы вместо следующей строки
#
#     ldap machine suffix = ou=Computers
#
# напишем другую:
    ldap machine suffix = ou=Users
    ldap idmap suffix = ou=Idmap
    idmap backend = ldapsam:ldap://127.0.0.1/
    idmap uid = 10000-20000
    idmap gid = 10000-20000

    ldap delete dn = Yes
    ldap ssl = no

    add user script = /usr/sbin/smbldap-useradd -n -a "%u"
    delete user script = /usr/sbin/smbldap-userdel "%u"
```



```

add group script = /usr/sbin/smbldap-groupadd -p "%g"
delete group script = /usr/sbin/smbldap-userdel "%g"

add user to group script = /usr/sbin/smbldap-groupmod -m "%u" "%g"
delete user from group script = /usr/sbin/smbldap-groupmod -x "%u"
"%g"

set primary group script = /usr/sbin/smbldap-usermod -g "%g" "%u"

add machine script = /usr/sbin/smbldap-useradd -w "%u"

#PDC
domain master = yes
preferred master = yes
#BDC
# domain master = no
# preferred master = no
domain logons = Yes

logon script =
# Если хотите, что бы профили всех пользователей были перемещаемыми
# и хранились на сервере (со всеми гигабайтами фильмов и личных фотографий)
# то укажите такое значение следующего параметра:
#
# logon path = \\%L\Profiles\%a\%U
#
# Если вы не хотите гонять профили по сети, оставьте значение пустым,
# (но ни в коем случае не комментируйте эту строку, она просто получит
# значение по умолчанию), вот так:
logon path =
logon drive = U:
logon home = \\%L\users\%U

#===== Share Definitions =====
[netlogon]
comment = Network Logon Service
path = /var/lib/samba/netlogon
browseable = yes
guest ok = yes
writable = no
share modes = no

[Profiles]
admin users = admin
create mode = 600
directory mode = 700
path = /var/lib/samba/profiles
browseable = yes
guest ok = yes
writable = yes

[homes]
comment = Home Directories
browseable = no
read only = no

[public]

```

```
path = /pub
guest ok = yes
read only = no

[users]
path = /home/users
writable = yes
printable = no
```

Добавим запуск winbind с самбой (если нужно):

Файл: /etc/conf.d/samba

```
...
daemon_list="smbd nmbd winbind"
...
```

Введём пароль рабочей станции:

```
smbpasswd -w secret
```

Введём контроллер домена, собственно в домен

```
net rpc join -S neptun -U Administrator
```

Настройка системы на авторизацию в LDAP

--ladserg 14:05, 28 июля 2006 (UTC) У меня честно говоря не получилось сделать авторизацию пользователей samba через LDAP без настройки поддержки авторизации системных пользователей в LDAP, пришлось настраивать и это.

Сначала поправим файл /etc/ldap.conf, приведя его примерно к следующему виду:

Файл: /etc/ldap.conf

```
host 127.0.0.1
base dc=amber,dc=global,dc=com
ldap_version 3
rootbinddn cn=Manager,dc=amber,dc=global,dc=com
bind_timelimit 10
bind_policy soft
pam_filter objectClass=posixAccount
pam_password exop
nss_base_passwd ou=Users,dc=tty,dc=perm,dc=ru?one
nss_base_shadow ou=Users,dc=tty,dc=perm,dc=ru?one
nss_base_group ou=Groups,dc=tty,dc=perm,dc=ru?one
nss_base_hosts ou=Hosts,dc=tty,dc=perm,dc=ru?one
nss_base_services ou=Services,dc=tty,dc=perm,dc=ru?one
nss_base_networks ou=Networks,dc=tty,dc=perm,dc=ru?one
nss_base_protocols ou=Protocols,dc=tty,dc=perm,dc=ru?one
nss_base_rpc ou=Rpc,dc=tty,dc=perm,dc=ru?one
nss_base_ethers ou=Ethers,dc=tty,dc=perm,dc=ru?one
nss_base_netmasks ou=Networks,dc=tty,dc=perm,dc=ru?one
nss_base_bootparams ou=Ethers,dc=tty,dc=perm,dc=ru?one
nss_base_aliases ou=Aliases,dc=tty,dc=perm,dc=ru?one
nss_base_netgroup ou=Netgroup,dc=tty,dc=perm,dc=ru?one
```

```
ssl off
nss_reconnect_tries 4
nss_reconnect_sleeptime 1
nss_reconnect_maxsleeptime 16
nss_reconnect_maxconntries 2
```

Сим мы скажем `nss_ldap` где и как искать записи пользователей и групп.

Теперь создадим файл `/etc/ldap.secret` и при помощи любого текстового редактора в `plain/text` виде занесём туда пароль пользователя, который выше у нас указан в опции `rootbinddn`, например пароль `secret`:

Файл: `/etc/ldap.secret`

`secret`

Затем непременно установим на него нужные права:

Code: Установка прав на файл `/etc/ldap.secret`

```
#chmod 600 /etc/ldap.secret
#chown root:wheel /etc/ldap.secret
```

Далее приведём файл `/etc/pam.d/system-auth` к следующему виду:

Файл: `/etc/pam.d/system-auth`

```
auth      required      pam_env.so
auth      sufficient    pam_unix.so likeauth nullok
auth      sufficient    pam_ldap.so use_first_pass
auth      required      pam_deny.so

account   sufficient    pam_ldap.so
account   required      pam_unix.so

password  required      pam_cracklib.so difok=2 minlen=8 dcredit=2 ocredit=2
retry=3
password  sufficient    pam_unix.so nullok md5 shadow use_authok
password  sufficient    pam_ldap.so use_authok
password  required      pam_deny.so

session   required      pam_limits.so
session   required      pam_unix.so
session   required      pam_mkhomedir.so skel=/etc/skel/ umask=077
session   optional      pam_ldap.so
```

Обратите внимание на строку:

```
session required pam_mkhomedir.so skel=/etc/skel/ umask=077
```

Она заставляет систему создавать домашние каталоги для тех пользователей у которых они ещё не созданы, при этом в новый каталог помещается содержимое директории `/etc/skel/` и задаётся маска каталога `0x700`

После чего правим файл `/etc/nsswitch.conf`, приводя его к следующему виду:

Файл: `/etc/nsswitch.conf`

```
passwd:      files ldap
shadow:      files ldap
group:       files ldap
```

```

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files
rpc:        ldap [NOTFOUND=return] db files
ethers:     ldap [NOTFOUND=return] db files
netmasks:   files
netgroup:   ldap [NOTFOUND=return] files
bootparams: files

automount:  files
aliases:    files

```

Предупреждение: Ни в коем случае не добавляйте значение `ldap` к следующим базам: `hosts`, `networks`, `protocols`, `services`. Иначе вы рискуете не дожидаться следующей загрузки системы.

Всё, теперь мы указали системе брать пользователей как из системных файлов, так и из LDAP.

Перезагрузим наш компьютер, дабы убедиться что система грузится нормально. Если система останавливается на загрузке `udev`, то смотрите ошибки в файле `/etc/nsswitch.conf`, может вы указали использовать `ldap` не в той базе.

Управление пользователями

Ранее мы установили пакет `smbldap-tools`, теперь рассмотрим возможность управления пользователями с его помощью.

Создание пользователя

```

smbldap-useradd [-o] [-a] [-b] [-w] [-i] [-u uid] [-g gid ] [-G groups,,,]
[-n] [-d home] [-s shell] [-c gecos] [-m [-k]] [-t] [-P] [-A 0|1] [-B 0|1]
[-C sambaHomePath] [-D sambaHomeDrive] [-E sambaLogonScript] [-F
sambaProfilePath]
[-H sambaAcctFlags] [-N surname] [-S family name] [-M local mailAddress,,,]
[-T mailToAddress] [-?] user

```

Где:

```

user - системное имя создаваемого пользователя
-o   - add the user in the organizational unit (relative to the user suffix)
-a   - is a Windows User (otherwise, Posix stuff only)
-b   - is a AIX User
-w   - is a Windows Workstation (otherwise, Posix stuff only)
-i   - is a trust account (Windows Workstation)
-u   - uid
-g   - gid
-G   - список групп пользователя, разделённых запятой.
-n   - do not create a group
-d   - домашний каталог пользователя (по умолчанию /home/имя_пользователя)
-s   - оболочка пользователя (по умолчанию /bin/false)
-c   - отображаемое в Windows имя пользователя
-m   - создать домашний каталог и скопировать в него файлы из /etc/skel
-k   - указать иной каталог, из которого будут копироваться файлы при
      создании домашнего каталога пользователя (используется с ключём -m)
-t   - time. Wait 'time' seconds before exiting (when adding Windows

```

Workstation)

```
-P - ends by invoking smbldap-passwd
-A - возможность менять пароль пользователем, значение 0 если нет, 1 если
да
-B - пользователь должен поменять пароль, значение 0 если нет, 1 если да
-C - домашний каталог samba (например '\\PDC-SRV\homes')
-D - буква диска для монтирования домашнего каталога samba (например 'H:')
-E - скрипт, выполняемый при входе в систему
-F - каталог профиля пользователя (например '\\PDC-SRV\profiles\foo')
-H - sambaAcctFlags (samba account control bits like '[NDHTUMWSLKI]')
-N - настоящее имя пользователя (для русских ещё и отчество)
-S - фамилия пользователя
-M - local mailAddress (comma seperated)
-T - mailToAddress (forward address) (comma seperated)
-? - отобразить помощь
```

Например создание пользователя ladserg:

```
smbldap-useradd -a -c 'Serg Alex Lad' -N 'Serg Alex' -S 'Lad' -s /bin/bash ladserg
```

К сожалению подружить smbldap-tools с русским мне не удалось, даже при использовании кодировки UTF-8.

Итак, в приведённом выше примере будет создан пользователь с системным именем ladserg, фамилией Lad, именем Serg Alex, оболочкой /bin/bash, домашним каталогом /home/ladserg. Флаг -a укажет, что пользователь также будет являться пользователем домена.

Изменение пароля

```
smbldap-passwd [-s] [-u] [-h] username
```

Где:

```
username      - имя пользователя
-h, -?, --help - показать помощь
-s            - обновить только samba пароль
-u           - обновить только UNIX пароль
```

Например:

```
smbldap-passwd ladserg
```

После чего дважды будет запрошен пароль.

Теперь можно попробовать зайти в систему под учётной записью только что созданного пользователя.

Модификация пользователя

```
smbldap-usermod [-a] [-c comment] [-d home_dir] [-e expiration_date]
[-g initial_group] [-r new_login_name] [-p passwd] [-s shell] [-u uid [-o]]
[-x]
[-A canchange] [-B mustchange] [-C smbhome] [-D homedrive] [-E scriptpath]
[-F profilepath] [-G group[,...]] [-H acct-flags] [-N canonical_name]
[-S surname] [-P] login
```

Где:

- c - Полное имя
- d - Домашний каталог
- r - новое имя пользователя (cn, sn и dn будут обновлены)
- u - uid
- o - uid может быть не уникальным
- g - gid
- G - список групп пользователя, разделённых запятой.
- s - оболочка
- N - настоящее имя пользователя (для русских ещё и отчество)
- S - фамилия пользователя
- P - ends by invoking smbldap-passwd

For samba users:

- a - add sambaSAMAccount objectclass
- e - expire date ("YYYY-MM-DD HH:MM:SS")
- A - возможность менять пароль пользователем, значение 0 если нет, 1 если да
- B - пользователь должен поменять пароль, значение 0 если нет, 1 если да
- C - домашний каталог samba (например '\\PDC-SRV\homes')
- D - буква диска для монтирования домашнего каталога samba (например 'H:')
- E - скрипт, выполняемый при входе в систему
- F - каталог профиля пользователя (например '\\PDC-SRV\profiles\foo')
- H - sambaAcctFlags (samba account control bits like '[NDHTUMWSLKI]')
- I - disable an user. Can't be used with -H or -J
- J - enable an user. Can't be used with -H or -I
- M - mailAddresses (comma seperated)
- T - mailToAddress (forward address) (comma seperated)
- ?|-h - отобразить помощь

Например команда:

```
smbldap-usermod -A 1 ladserg
```

Позволит пользователю ladserg менять пароль. А команда:

```
smbldap-usermod -a slad-adm
```

Добавит к учётной записи пользователя slad-adm класс sambaSAMAccount, что сделает его пользователем samba.

Удаление пользователя

```
smbldap-userdel [-r|-R|-?] username
```

Где:

- r - удалить домашний каталог
- R - удалить домашний каталог с запросом на удаление каждого файла
- ? - отобразить помощь

Например команда:

```
smbldap-userdel -r slad-adm
```

удалит пользователя slad-adm, и его домашний каталог.

Управление пользователями в оффтопике

Если вы вводите в домен компьютеры под управлением Windows, то вам пригодятся пара утилит, архив которых можно скачать отсюда:

<ftp://ftp.microsoft.com/Softlib/MSLFILES/SRVTOOLS.EXE>

Источники

- <http://us4.samba.org/samba/docs/man/Samba-Guide/happy.html>
- http://www.opennet.ru/base/net/ldap_spama_pdc.txt.html

Создание виртуальной почтовой системы

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/virt-mail-howto.xml>

С версии: 1.0

1. Введение

Для большинства пользователей gentoo простого почтового клиента и fetchmail (для сбора почты) достаточно. Однако, если вы размещаете на своей системе домен, вам необходим полноценный MTA (Mail Transfer Agent). И если вы размещаете на своей системе несколько доменов, тогда вам точно необходимо что-то мощное для обработки всей почты ваших пользователей. Эта система была разработана для элегантного решения этой проблемы.

Виртуальная почтовая система должна быть способна обработать почту от многочисленных доменов с множеством пользователей через разнообразные интерфейсы. С этим связано несколько проблем, которые надо решить. Для примера, что если вы имеете двух пользователей в разных доменах, которые хотят иметь одинаковые имена? Если вы предоставляете сервисы imap и smtp-авторизации, как вы комбинируете различные демоны авторизации в одной системе? Как обеспечиваете безопасность многочисленных компонентов которая содержит система? Как вы управляете этим всем?

Это howto покажет вам, как настроить гибкую систему обработки почты из стольких доменов, сколько ваша поддерживает система, как создавать виртуальных пользователей не требующих наличия аккаунта в системе, иметь доменные имена пользователей, как авторизовать пользователей через веб, imap, smtp и pop3 снова таки из одной базы данных, использовать ssl-транспорт для безопасности, как обрабатывать списки рассылки для любого домена на машине, и контролировать все хорошей и простой базой данных mysql.

Конечно существует множество путей настройки виртуальной почтовой системы. Другой способ возможно окажется более подходящим вашим нуждам. Дополнительную информацию вы можете найти на <http://www.qmail.org/> и <http://www.exim.org/>

Мы будем использовать следующие программы: apache, courier-imap, pam_mysql, postfix, mod_php, phpmyadmin, squirrelmail, cyrus-sasl, mysql, php, и mailman.

Убедитесь что вы добавили в файле /etc/make.conf в переменную USE следующие пакеты: USE="mysql imap libwww maildir sasl ssl". Иначе, возможно, вам придется перекомпилировать пакеты, для поддержки всех необходимых протоколов. Затем, хорошей идеей будет отключить любые другие почтовые и сетевые программы которые вам не нужны, вроде IPv6.

Важно: Это howto написано для postfix-2.0.x. Если вы используете postfix версии < 2 некоторые переменные используемые в этом документе могут отличаться. В таком случае рекомендуется обновить postfix. Некоторые пакеты используемые в

этом документе зависят от версии. Вы поступите мудро, если прочтете документацию поставляемую с пакетами, при возникновении вопросов.

Важно: В нашем документе мы будем использовать apache-1.3.x. В портежах Apache-2 был помечен stable, но как бы то ни было существуют еще некоторые проблемы интеграции с php. Пока php-поддержка в apache-2.0.x не будет отмечена stable, в этом документе будем продолжать использовать apache-1.3.x.

Важно: Вам необходимо доменное имя для работы общедоступного почтового сервера, или по крайней мере иметь MX-записи для домена. В идеале вы должны иметь контроль над двумя доменами, для извлечения пользы из функциональности новой виртуальной системы.

Важно: Убедитесь что в /etc/hostname указано правильное имя сервера, также убедитесь что в /etc/hosts нет конфликтных значений.

Примечание: Рекомендуется прочесть документ полностью и ознакомиться со всеми шагами настройки, перед тем как начинать инсталляцию. Если вы столкнетесь с проблемами при любом из шагов, проверьте troubleshooting guide в конце этого документа. Также не все упомянутые пакеты необходимы, просто настройка, описанная здесь, достаточно гибка. Для примера, если вам не требуется веб-интерфейса, вы свободны пропустить соответствующий раздел squirrelmail.

2. Начальная настройка postfix

Листинг 2.1: Инсталляция postfix
emerge postfix

Предупреждение: Проверьте чтобы у вас не были проинсталлированы любые другие MTA, например ssmtp, exim или qmail, иначе у вас могут быть БОЛЬШИЕ проблемы.

После инсталляции postfix, время его настроить. Измените следующие настройки в /etc/postfix/main.cf:

Листинг 2.2: /etc/postfix/main.cf

```
myhostname = $host.domain.name
mydomain = $domain.name
inet_interfaces = all
mydestination = $myhostname, localhost.$mydomain $mydomain
mynetworks = my.ip.net.work/24, 127.0.0.0/8
home_mailbox = .maildir/
local_destination_concurrency_limit = 2
default_destination_concurrency_limit = 10
```

Следующие изменения следуют для /etc/postfix/master.cf. Они включают режим подробного протоколирования для отладки:

Листинг 2.3: /etc/postfix/master.cf

```
# service type private unpriv chroot wakeup maxproc command + args
#          (yes) (yes) (yes) (never) (50)
#
=====
smtp      inet  n       -       n       -       -       smtpd -v

(просто добавьте -v после smtpd)
```

Дальше правим /etc/mail/aliases для добавления локальных псевдонимов. По крайне туда должен быть включен псевдоним для root типа: root: your@email.address.

Листинг 2.4: Запуск postfix'a в первый раз

```
# /usr/bin/newaliases
(это создаст новые псевдонимы. Это требуется делать)
(когда вы создаете или обновляете файл псевдонимов.)
# /etc/init.d/postfix start
```

Теперь postfix запущен, запустите вашего любимого консольного почтового клиента и пошлите самому себе письмо. Я использую mutt для всей переписки из консоли.

Примечание: Настоятельно рекомендуется проверить, чтобы postfix доставлял почту локальным пользователям, перед тем как перейти к следующему шагу.

3. Courier-imap

Листинг 3.1: Инсталляция courier-imap

```
# emerge courier-imap
```

Листинг 3.2: Настройка courier-imap

```
# cd /etc/courier-imap
(если вы хотите использовать возможности ssl в courier-imap или pop3, )
(вам необходимо создать сертификаты)
(если вы не хотите использовать ssl, просто пропустите следующий шаг )

# nano -w pop3d.cnf
# nano -w imapd.cnf
(измените значения C, ST, L, CN и адреса e-mail)

# mkpop3dcert
# mkimapdcert
```

Листинг 3.3: Запуск нужных вам сервисов courier

```
# /etc/init.d/courier-imapd start
# /etc/init.d/courier-imapd-ssl start
# /etc/init.d/courier-pop3d start
# /etc/init.d/courier-pop3d-ssl start
```

Снова запустите ваш любимый почтовый клиент и проверьте что сервисы начали принимать и посылать почту. Теперь когда основная часть заработала, перейдем к связыванию всех компонентов, для получения спокойно работающей системы. Снова, проверьте, что все что мы сделали работает, перед тем как переходить к следующему шагу.

4. Cyrus-sasl

Следующим шагом будет инсталляция cyrus-sasl. Sasl на самом деле играет роль передатчика авторизационных переменных к pam (Pluggable Authentication Modules), который передаст эту информацию mysql, для авторизации smtp пользователей. Мы не будем проверять работоспособность sasl, пока не настроим mysql, и не создадим тестовых пользователей. В конечном результате, он будет авторизовать пользователей в mysql.

Примечание: По некоторым причинам, sasl плохо работает с pam используя shadow. Для меня долгое время это была большая проблема. Если кто-нибудь знает почему sasl не авторизует пользователей из /etc/shadow в текущем релизе gentoo, пожалуйста напишите мне об этом, я буду очень рад услышать решение этой проблемы. E-mail.

Листинг 4.1: Инсталляция и настройка cyrus-sasl

```
# USE='-ldap -mysql' emerge cyrus-sasl
(мы не используем ldap и мы не используем возможности sasl-mysql,
поэтому мы отключаем их )
```

Дальше правим /usr/lib/sasl2/smtp.conf.

Листинг 4.2: Запуск sasl

```
# nano -w /usr/lib/sasl2/smtp.conf
pwcheck_method: saslauthd
mech_list: LOGIN PLAIN
(важно выключить метод авторизации который мы не используем.
Они могут привести к граблям на некоторых почтовых клиентах)
# /etc/init.d/saslauthd start
```

5. SSL-сертификаты для Postfix и Apache

Дальше делаем ssl-сертификаты для postfix и apache.

Листинг 5.1:

```
# cd /etc/ssl/  
# nano -w openssl.cnf  
(измените следующие значения для вашего домена:)  
countryName_default  
stateOrProvinceName_default  
localityName_default  
0.organizationName_default  
commonName_default  
emailAddress_default.
```

(если каких-то переменных нет, просто добавьте их в любое удобное место)

```
# cd misc  
# nano -w CA.pl  
(нам необходимо добавить -nodes в код строк "# create a certificate" и  
"# create a certificate request" таким образом чтобы наши новые ssl-сертификаты  
загрузились без пароля. Иначе после перезагрузки системы ssl-сертификаты будут  
не доступны )  
(найдите эти строки в файле и измените их:)
```

```
# create a certificate  
system ("$REQ -new -nodes -x509 -keyout newreq.pem -out newreq.pem $DAYS");
```

```
# create a certificate request  
system ("$REQ -new -nodes -keyout newreq.pem -out newreq.pem $DAYS");
```

(далее создаем сертификат для postfix)

```
# ./CA.pl -newca  
# ./CA.pl -newreq  
# ./CA.pl -sign  
# cp newcert.pem /etc/postfix  
# cp newreq.pem /etc/postfix  
# cp demoCA/cacert.pem /etc/postfix  
(а теперь такой же для apache)
```

```
# openssl req -new > new.cert.csr  
# openssl rsa -in privkey.pem -out new.cert.key  
# openssl x509 -in new.cert.csr -out new.cert.cert -req -signkey new.cert.key -days 365  
(оставьте получившиеся сертификаты, мы их проинсталлируем когда будем настраивать apache )
```

6. Добавим SSL и SASL поддержку в Postfix

Теперь поправим конфигурацию postfix чтобы включить совместимость с sasl и ssl. Добавьте следующие параметры в конец файла, где их легко потом можно будет найти.

Листинг 6.1: /etc/postfix/main.cf
nano -w /etc/postfix/main.cf

```
smtpd_sasl_auth_enable = yes  
smtpd_sasl2_auth_enable = yes  
smtpd_sasl_security_options = noanonymous  
broken_sasl_auth_clients = yes  
smtpd_sasl_local_domain =
```

(опция `broken_sasl_auth_clients` и метод авторизации используется только для outlook и outlook express, они не документированы. Опция `smtpd_sasl_local_domain` добавляет имя домена для клиентов использующих smtp-авторизацию. Убедитесь что postfix отвергает пустое или просто имя пользователя, и они не могут авторизоваться)

```
smtpd_recipient_restrictions =
    permit_sasl_authenticated,
    permit_mynetworks,
    reject_unauth_destination
```

```
smtpd_use_tls = yes
#smtpd_tls_auth_only = yes
smtpd_tls_key_file = /etc/postfix/newreq.pem
smtpd_tls_cert_file = /etc/postfix/newcert.pem
smtpd_tls_CAfile = /etc/postfix/cacert.pem
smtpd_tls_loglevel = 3
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
```

(опция `smtpd_tls_auth_only` закомментирована для простоты тестирования, вы можете включить ее позже, если захотите)

postfix reload

Теперь мы попробуем проверить работоспособность postfix'a, и то что он прочел файлы конфигурации.

Листинг 6.2: Проверка поддержки sasl и tls
telnet localhost 25

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mail.domain.com ESMTP Postfix
EHLO domain.com
250-mail.domain.com
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250-XVERP
250 8BITMIME
^]
telnet> quit
```

Проверьте, что ответ postfix'a содержит линии AUTH и STARTTLS. Как уже замечалось выше, AUTH пока не будет работать, потому что sasl пытается авторизовать из sasldb, вместо shadow по неизвестным причинам. Итак мы просто пропускаем это и дальше настраиваем mysql для принятия нашей авторизации и

информации о виртуальных доменах.

7. MySQL

Для инсталляции mysql нам необходим dump-файл genericmailsql.sql

Листинг 7.1: Инсталляция и настройка MySQL

```
# emerge mysql
```

```
# /usr/bin/mysql_install_db
```

(далее следуйте указаниям на экране, для добавления пароля root в mysql, не mysqladmin, иначе ваша база будет открыта всем ветрам)

```
# /etc/init.d/mysql start
```

```
# mysqladmin -u root -p create mailsql
```

```
# mysql -u root -p mailsql < genericmailsql.sql
```

```
# mysql -u root -p mysql
```

```
mysql> GRANT SELECT,INSERT,UPDATE,DELETE
```

```
-> ON mailsql.*
```

```
-> TO mailsql@localhost
```

```
-> IDENTIFIED BY '$password';
```

```
-> quit
```

(проверьте, что новый mailsql пользователь, может подключаться к mysql серверу)

```
# mysql -u mailsql -p mailsql
```

Ваша новая база имеет значения по умолчанию и таблицы для двух доменов. Вложены следующие таблицы:

- **alias** — локальные e-mail и информация о псевдонимах для mailman
- **relocated** — информация о адреса перемещенных пользователей
- **transport** — информация о почтовых транспортах для всех доменов размещающихся у вас
- **users** — информация о всех пользователях
- **virtual** — информация о псевдонимах для виртуальных доменов

Листинг 7.2: пример таблицы псевдонимов

```
id alias destination
```

```
1 root foo@bar.com
```

```
2 postmaster foo@bar.com
```

Листинг 7.3: пример таблицы пользователей

(приведем для ясности строку)

```
id email clear name uid gid homedir \  
maildir quota postfix
```

```
10 foo@virt-bar.org $password realname virtid virtid /home/vmail \  
/home/vmail/virt-bar.org/foo/.maildir/ y
```

```
13 foo@bar.com $password realname localid localid /home/foo \  
/home/foo/.maildir/ y
```

Листинг 7.4: пример таблицы транспортов

```
id domain destination
1 bar.com local:
2 virt-bar.org virtual:
```

Листинг 7.5: пример таблицы псевдонимов для виртуальных доменов

```
id email destination
3 root@virt-bar.org other@email.address
```

8. Apache и phpMyAdmin

Итак, следующим шагом мы настроим apache и создадим интерфейс для еще более простого взаимодействия с базой данных.

Листинг 8.1: Настройка apache и phpmyadmin

```
# emerge apache mod_php phpmyadmin
```

Существует множество руководств по настройке apache с поддержкой php. Например, <http://www.linuxguruz.org/z.php?id=31>. Также многочисленные сообщения на <http://forums.gentoo.org> где рассматриваются решения проблем возникших в ходе инсталляции (поиск по "apache php"). Итак, я не стараюсь раскрыть эту тему здесь. Настройте apache и php, затем продолжим вместе настройку. Теперь слово для умных: .htaccess положите в директорию к phpmyadmin. Если вы не сделаете этого, поисковые системы проиндексируют страницы phpmyadmin и каждый сможет получить к нему доступ с помощью google, и изменить ваши базы, что не есть хорошо. Существует много howto как это сделать. <http://docs.csoft.net/micro/black-htaccess.html>.

Теперь мы переходим к инсталляции сертификатов для apache, созданных нами ранее. Директивы apache которые вам требуется изменить для этого:

```
SSLCertificateFile /path/to/certs/new.cert.cert
SSLCertificateKeyFile /path/to/certs/new.cert.key
```

Листинг 8.2: Инсталляция Apache SSL сертификатов

```
# cp /etc/ssl/misc/new.cert.cert /etc/apache/conf/ssl/
# cp /etc/ssl/misc/new.cert.key /etc/apache/conf/ssl/
# nano -w /etc/apache/conf/vhosts/ssl.default-vhost.conf
(измените следующие параметры)
```

```
ServerName host.domain.name
ServerAdmin your@email.address
SSLCertificateFile /etc/apache/conf/ssl/new.cert.cert
SSLCertificateKeyFile /etc/apache/conf/ssl/new.cert.key
```

```
# /etc/init.d/apache restart
```

Примечание: Если у вас apache уже проинсталлирован, вам вероятно придется сделать перезагрузку сервера. Проверьте логи системы на предмет корректного

запуска apache.

Дальше настраиваем phpMyAdmin.

Листинг 8.3: Настройка phpMyAdmin

```
# nano -w /home/httpd/htdocs/phpmyadmin/config.inc.php
```

(измените следующие параметры)

```
$cfg['Servers'][$i]['host'] = 'localhost';      // MySQL hostname
$cfg['Servers'][$i]['controluser'] = 'mysql';   // MySQL настройки системного аккаунта
                                                // (этот аккаунт должен иметь read-only
$cfg['Servers'][$i]['controlpass'] = '$password'; // доступ к таблицам "mysql/user"
                                                // и "mysql/db" tables)
$cfg['Servers'][$i]['user'] = 'mysql';         // MySQL пользователь
$cfg['Servers'][$i]['password'] = '$password'; // MySQL пароль
```

Теперь введите адрес `phpmyadmin` страницы и просмотрите ваши таблицы в базе. Вы можете добавить локальные псевдонимы, поправить таблицу пользователей и добавить тестового пользователя, изменить таблицу транспортов для добавления информации о ваших доменах. Значений по умолчанию которые устанавливаются вместе с `dump`-файлом должно быть достаточно для примера, чтобы помочь вам настроить систему. Убедитесь что ввели в базу корректную информацию. Для примера, убедитесь что директории локальных пользователей существуют и указаны корректные `uid/gid`. Почтовые директории пользователей, должны быть созданы `postfix`, при первой принятой почте для пользователя. Будет неплохо, если вы пошлете "Добро пожаловать!", чтобы убедиться, что `.maildir` создан.

9. Vmail-пользователь

Здесь вы можете быть удивлены, узнав, что для виртуальных акаунтов, используются пользователи и директории, и будете правы.

Листинг 9.1: Adding the vmail user

```
# adduser -d /home/vmail -s /bin/false vmail
# uid=`cat /etc/passwd | grep vmail | cut -f 3 -d :`
# groupadd -g $uid vmail
# mkdir /home/vmail
# chown vmail. /home/vmail
```

Теперь, когда вы настраиваете виртуальный акаунт, используйте `vmail uid, gid`, и его домашнюю папку. Когда вы создаете локальный акаунт, используйте `uid, gid` и домашнюю папку нового пользователя, а не `vmail`. Мы имеем ввиду, что если вы захотите создать `php`-страничку для администрирования пользователей, не забывайте, что `phpmyadmin` в целом справляется с этой работой очень неплохо.

10. Настройка MySQL авторизации и виртуальных доменов

Дальше мы перенастроим нашу авторизацию, на использование mailsql базы в courier-imap и postfix. Во всех следующих примерах, замените \$password паролем, который вы задали пользователю mailsql для mysql.

Листинг 10.1:

```
# emerge /usr/portage/sys-libs/pam_mysql/pam_mysql-$currentversion.ebuild
```

(этот пакет здесь задан маской, которую вы должны заменить на текущую версию пакета. версию пакета вы можете посмотреть в portage)

```
# nano -w /etc/pam.d/imap
```

(закомментируйте существующие строки настройки авторизации, и добавьте указанные ниже)

```
#auth    required    pam_nologin.so
#auth    required    pam_stack.so service=system-auth
#account required    pam_stack.so service=system-auth
#session required    pam_stack.so service=system-auth
```

```
auth    optional    pam_mysql.so host=localhost db=mailsql user=mailsql \
    passwd=$password table=users usercolumn=email passwdcolumn=clear crypt=0
account required    pam_mysql.so host=localhost db=mailsql user=mailsql \
    passwd=$password table=users usercolumn=email passwdcolumn=clear crypt=0
```

```
# nano -w /etc/pam.d/pop3
```

```
# nano -w /etc/pam.d/smtp
```

(сделайте такие же изменения в pop3 и smtp файлах)

Далее нам нужно поправить конфигурацию авторизации courier.

Листинг 10.2:

```
# nano -w /etc/courier-imap/authdaemonrc
```

```
authmodulelist="authmysql authpam"
```

```
# nano -w /etc/courier-imap/authdaemond.conf
```

```
AUTHDAEMOND="authdaemond.mysql"
```

```
# nano -w /etc/courier-imap/authmysqlrc
```

```
MYSQL_SERVER      localhost
MYSQL_USERNAME     mailsql
MYSQL_PASSWORD     $password
MYSQL_DATABASE     mailsql
MYSQL_USER_TABLE   users
#MYSQL_CRYPT_PWFIELD crypt (эта строка должна быть закомментирована)
MYSQL_CLEAR_PWFIELD clear
MYSQL_UID_FIELD    uid
MYSQL_GID_FIELD    gid
MYSQL_LOGIN_FIELD  email
MYSQL_HOME_FIELD   homedir
MYSQL_NAME_FIELD   name
MYSQL_MAILDIR_FIELD maildir
```

```
# /etc/init.d/authdaemond restart
```

```
# /etc/init.d/saslauthd restart
```

Мы уже почти подошли к тому что вам обещал. Далее мы настроим необходимые конфиги postfix'а для связки с базой данных для всех необходимых необходимых транспортов.

Листинг 10.3: /etc/postfix/mysql-aliases.cf
nano -w /etc/postfix/mysql-aliases.cf
mysql-aliases.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = alias
select_field = destination
where_field = alias
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.4: /etc/postfix/mysql-relocated.cf
nano -w /etc/postfix/mysql-relocated.cf
mysql-relocated.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = relocated
select_field = destination
where_field = email
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.5: /etc/postfix/mysql-transport.cf (необязательно)
nano -w /etc/postfix/mysql-transport.cf
mysql-transport.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = transport
select_field = destination
where_field = domain
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.6: /etc/postfix/mysql-virtual-gid.cf (необязательно)
nano -w /etc/postfix/mysql-virtual-gid.cf
#mysql-virtual-gid.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = users
select_field = gid
where_field = email
additional_conditions = and postfix = 'y'
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.7: /etc/postfix/mysql-virtual-maps.cf
nano -w /etc/postfix/mysql-virtual-maps.cf
#mysql-virtual-maps.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = users
select_field = maildir
where_field = email
additional_conditions = and postfix = 'y'
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.8: /etc/postfix/mysql-virtual-uid.cf (необязательно)
nano -w /etc/postfix/mysql-virtual-uid.cf
mysql-virtual-uid.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = users
select_field = uid
where_field = email
additional_conditions = and postfix = 'y'
hosts     = unix:/var/run/mysqld/mysqld.sock
```

Листинг 10.9: /etc/postfix/mysql-virtual.cf
nano -w /etc/postfix/mysql-virtual.cf
mysql-virtual.cf

```
user      = mailsql
password  = $password
dbname    = mailsql
table     = virtual
select_field = destination
where_field = email
hosts     = unix:/var/run/mysqld/mysqld.sock
```

И последнее, правим /etc/postfix/main.cf еще один раз.

Листинг 10.10: /etc/postfix/main.cf
nano -w /etc/postfix/main.cf
alias_maps = mysql:/etc/postfix/mysql-aliases.cf
relocated_maps = mysql:/etc/postfix/mysql-relocated.cf

```
local_transport = local
local_recipient_maps = $alias_maps $virtual_mailbox_maps unix:passwd.byname
```

```
virtual_transport = virtual
virtual_mailbox_domains =
    virt-bar.com,
    $other-virtual-domain.com
```

```
virtual_minimum_uid = 1000
virtual_gid_maps = static:$vmail-gid
virtual_mailbox_maps = mysql:/etc/postfix/mysql-virtual-maps.cf
virtual_alias_maps = mysql:/etc/postfix/mysql-virtual.cf
virtual_uid_maps = static:$vmail-uid
virtual_mailbox_base = /
#virtual_mailbox_limit =
```

Здесь видно чем значительно отличается postfix 2.0.x от 1.1.x. Сильнее всего заметно отсутствие необходимости в таблицах транспорта, virtual-gid и virtual-uid, хотя эти таблицы и вкладываются, чтобы вы могли их использовать при необходимости.

Примечание: Рекомендуем прочесть VIRTUAL_README идущий с postfix, для большего количества информации.

Листинг 10.11:
postfix reload

Теперь, если все прошло хорошо, вы должны иметь работающий почтовый сервер. Пользователи должны быть способны авторизоваться в sql базе, использовать свой полный почтовый адрес для pop3, imap, и smtp. Настоятельно рекомендую проверить , что всё это действительно работает. Если вы столкнулись с проблемами, проверьте раздел troubleshooting в конце документа.

11. Squirrelmail

Листинг 11.1:
emerge squirrelmail
(добавим ссылку на htdocs для более короткого пути)
ln -s /home/httpd/htdocs/squirrelmail/ /home/httpd/htdocs/mail
cd /home/httpd/htdocs/mail/conf
./conf.pl

(измените настройки Organization, Server, и Folder для squirrelmail)
(теперь вы имеете возможность залогиниться в squirrelmail, снова с вашим полным email адресом, и использовать ваш новый webmail setup)

12. Mailman

Последний шаг — mailman. Новая версия имеет великолепную поддержку виртуальных доменов, поэтому я использую его, не говоря уже о том что это просто великолепная программа. Настоятельно рекомендую прочесть документацию mailman, включающую README.POSTFIX.gz, для более полного понимания.

Одно замечание, текущая версия mailman устанавливается в /usr/local/mailman. Если вы хотите изменить директорию установки, вы можете изменить в ebuild файле переменную INSTALLDIR.

Листинг 12.1: /usr/portage/net-mail/mailman/mailman-\$ver.ebuild

```
# nano -w /usr/portage/net-mail/mailman/mailman-$ver.ebuild
```

```
MAILGID="280"
```

(задайте MAILGID для группы mailman взамен nobody)

(это необходимо для интеграции с postfix)

Листинг 12.2:

```
# emerge mailman
```

(имя этого пакета маскировано, поэтому вам надо указать emerge путь к ebuild. После установки, следуйте инструкциям в README.gentoo.gz *исключая* — не добавляйте псевдонимы в /etc/mail/aliases. Вместо этого, мы укажем postfix использовать псевдонимы из базы)

```
# zless /usr/share/doc/mailman-$ver/README.gentoo.gz
```

Листинг 12.3: Настройки по умолчанию: Mailman/Defaults.py

```
# nano -w /var/mailman/Mailman/Defaults.py
```

(измените значения приведенные ниже на ваши, дальше будем настраивать виртуальные домены)

```
DEFAULT_EMAIL_HOST = 'domain.com'
```

```
DEFAULT_URL_HOST = 'www.domain.com'
```

Листинг 12.4: настройка mailman: mm_cfg.py

```
# nano -w /var/mailman/Mailman/mm_cfg.py
```

```
MTA = "Postfix"
```

```
POSTFIX_STYLE_VIRTUAL_DOMAINS = ['virt-domain.com', 'virt.domain2.com']
```

```
add_virtualhost('www.virt.domain.com', 'virt.domain.com')
```

```
add_virtualhost('www.virt.domain2.com', 'virt.domain2.com')
```

(это нужно для работы mailman с виртуальными доменами)

Листинг 12.5:

(создадим свой первый список рассылки)

```
# su mailman
```

```
# cd ~
```

```
# bin/newlist test
```

Enter the email of the person running the list: your@email.address

Initial test password:

Hit enter to continue with test owner notification...

(листы виртуального домена могут быть указаны в виде list@domain.com)

```
# bin/genaliases
```

(теперь ваши псевдонимы сгенерированы, проверьте, что они добавлены правильно)

```
# nano -w data/aliases
```

```
# STANZA START: test
```

```
# CREATED:
```

```
test: "/var/mailman/mail/mailman post test"
```

```
test-admin: "/var/mailman/mail/mailman admin test"
```

```
test-bounces: "/var/mailman/mail/mailman bounces test"
```

```
test-confirm:    "/var/mailman/mail/mailman confirm test"
test-join:       "/var/mailman/mail/mailman join test"
test-leave:      "/var/mailman/mail/mailman leave test"
test-owner:      "/var/mailman/mail/mailman owner test"
test-request:    "/var/mailman/mail/mailman request test"
test-subscribe:  "/var/mailman/mail/mailman subscribe test"
test-unsubscribe: "/var/mailman/mail/mailman unsubscribe test"
# STANZA END: test
```

```
# /etc/init.d/mailman start
```

```
# rc-update add mailman default
```

(для запуска mailman при каждой загрузке)

Листинг 12.6: Добавление поддержки псевдонимов mailman в postfix

```
# nano -w /etc/postfix/main.cf
```

```
owner_request_special = no
```

```
recipient_delimiter = +
```

(прочтите README.POSTFIX.gz для более детальной информации)

```
alias_maps =
```

```
    hash:/var/mailman/data/aliases,
```

```
    mysql:/etc/postfix/mysql-aliases.cf
```

```
virtual_alias_maps =
```

```
    hash:/var/mailman/data/virtual-mailman,
```

```
    mysql:/etc/postfix/mysql-virtual.cf
```

(это добавит поддержку файлов псевдонимов mailman'a в postfix, конечно

вы можете использовать таблицу в mysql для этого, но я ненавижу делать это руками.

Также если вы не используете виртуальных доменов, добавление псевдонимов, может привести к проблемам)

Теперь вы можете создавать листы рассылок для каждого домена в вашей системе. Последнее замечание, убедитесь, что mailman запущен из под пользователя mailman (su mailman) иначе у вас будут проблемы с разрешениями. прочтите документацию mailman для более детальной информации по управлению листами рассылки.

13. Фильтрация содержимого и Anti-Virus

Скоро будет,... это уже существует, но я должен еще немного разобраться в perl'e и протестировать результат. Если вы хотите помочь с этим, свяжитесь со мной.

14. Окончание

Итак,. Вы все сделали, теперь поправьте /etc/postfix/master.cf и отключите режим verbose. Возможно вы захотите доавать сервис в автозагрузку. Убедитесь что вы добавили в автозагрузку все сервисы которые используете - apache, mysql, saslauthd, postfix, courier-imapd, courier-imapd-ssl, courier-pop3d, и courier-pop3d-ssl, все зависит от вашего решения какие сервисы предоставлять. Обычно я разрешаю все сервисы.

Листинг 14.1: Окончание
`# postfix reload`
`# rc-update add $service default`

Вобщем инджой и хэв фан!

15. Troubleshooting

Введение

Troubleshooting: Это небольшой список решений для проблем, наиболее часто возникающих в ходе установки. Это не исчерпывающая информация, но вы можете начать поиск решения граблей отсюда. Такая запутанная настройка и инсталляция как в этом документе, может привести к тому что некоторые компоненты будут работать неправильно. Вообще советую сделать несколько следующих шагов. Запустите систему в базовой конфигурации, добавляя компонент за компонентом, чтобы выяснить какой из них сбоит.

Шаг 1: Проверьте ваши файлы конфигурации.

Опечатки — главный враг, особенно когда идет речь о системе авторизации. Проверьте ваши конфигурационные файлы и базу данных на предмет опечаток. Если вы делаете изменения настроек для сервиса, убедитесь, что вы перезапустили сервиса, дабы изменения вступили в силу.

Листинг 15.1:
`# /etc/init.d/service restart`

Шаг 2: Все необходимые сервисы запущены и работают?

Если не запущены, запустите. Крайне сложно отлаживать программу если она не запущена. Иногда сервис запускается, но не функционирует. Иногда, когда используется неправильный файл конфигурации сервис может использовать другой порт предназначенный другому процессу. Иногда вы можете воспользоваться `netstat`. Или если возможно, перезагрузите ваш сервер, это очистит любой зависший сервис.

Листинг 15.2:
`# /etc/init.d/$service status`
`# netstat -a | grep $service (или $port)`

Шаг 3: Все сервисы используют правильные файлы конфигурации?

Если вы внесли изменения в конфигурационные файлы сервиса, убедитесь что он использует новые параметры. Некоторые программы, вроде postfix могут показывать конфигурацию их параметров.

Листинг 15.3:

```
# apachectl fullstatus (необходим lynx)
# apachectl configtest (проверка нормальной конфигурации)
# postconf -n (показывает текущие параметры используемые postfix)
# /etc/init.d/$service restart
```

Шаг 4: Проверьте логи.

На то логи и существуют, чтобы помочь нам выяснить причину, а иногда и подсказать решение проблемы. Просмотрите все логи, которые имеют отношение к программе, ее сбою.

Листинг 15.4:

```
# kill -USR1 `ps -C metalog -o pid=` (для выключения metalog buffering)
# nano -w /var/log/mail/current
# cat /var/log/mysql/mysql.log
# tail /var/log/apache/error_log
```

Вы также можете найти параметр `debug_peer` в `main.cf` который может быть полезен. Включение этого параметра, увеличит подробность выводимых сообщений в логи.

Листинг 15.5: добавление поддержки `debug_peer`

```
# nano -w /etc/postfix/main.cf
debug_peer_level = 5
debug_peer_list = $host.domain.name
(раскомментируйте одну из нужных строк)
```

Шаг 5: Попробуйте связаться с сервисом.

SMTP, IMAP, и POP3 отвечают на telnet сессию, как вы видели немного раньше при конфигурации postfix. Иногда помогает открыть telnet сессию с сервисом, для того чтобы выяснить что случилось.

Листинг 15.6:

```
# telnet localhost $port
(SMTP на 25-м, IMAP, на 143-м, POP3 на 110-м порту. по крайней мере вы должны увидеть слово OK. Так вы будете знать что сервис запущен, и готов к ответу.)
```

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
* OK Courier-IMAP ready. Copyright 1998-2002 Double Precision, Inc.
```

Шаг 6: Иногда только большая пушка может дать нам информацию: strace.

Вы должны иметь эту программу всегда под рукой. Это бесценный инструмент для отлаживания программ. Вы можете запустить из командной строки `strace` и

наблюдать все системные вызовы происходящие в программе. Часто дампы содержат огромное количество информации, так что вы можете либо наблюдать в реальном времени, за транзакцией приводящей к краху программы, или записать его в файл, для неторопливого просмотра позже.

Листинг 15.7:

```
# emerge strace
# strace $command
# strace -p `ps -C $service -o pid=`
```

Виртуальная почтовая система на основе qmail/vpopmail.

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/qmail-howto.xml>

С версии: 1.0

1. Введение

Не важно для чего вы хотите организовать систему электронной почты, только лишь для сервисов, как самостоятельный сервер, или для нескольких виртуальных доменов, qmail может быть легко настроен для любых ваших нужд. Это руководство поможет вам установить и настроить qmail, для любого из этих вариантов с уклоном на удаленный доступ, в том числе через зашифрованные каналы данных.

Это руководство поможет вам освоить qmail, courier-imap, vpopmail, и horde/imp. Но вместе с ними так же вступят в "игру" daemontools, ucspi-tcp, mysql, apache, и mod_php. qmail предоставляет функции MTA (Mail Transfer Agent), courier-imap - предоставляет удаленный сбор почты, (pop3/imap сервер. прим.пер.) vpopmail предоставляет функции управления виртуальными доменами, и horde/imp - это почтовый веб-клиент.

До установки (emerging) чего-либо, вы должны корректно установить переменную USE. Если у вас уже были установлены какие-нибудь из этих пакетов, то, возможно, вам придется их переустановить. USE="maildir ssl imap mysql". Также, если вы хотите использовать horde/imp как web-клиент, то так же надо добавить USE="nls" до установки mod_php.

Примечание: Это руководство написано "по-шагам" Если в какой-то момент вы почувствуете что все то, что нужно готово, то можете не продолжать.

Существует достаточно пакетов, с помощью которых можно построить систему электронной почты. Сейчас самое время осмыслить и решить для вас ли qmail. У нас так же есть прекрасное руководство основанное на Postfix, или вы можете взглянуть на exim. Выбирайте решение для себя, а мы покажем вам, как использовать qmail.

2. qmail ("Разговаривая с собой")

Листинг 2.1: Emerge qmail
emerge qmail

Важно: Эта часть построена вокруг qmail-1.03-r13 или старше. Будет ли нижеизложенное работать с ранними версиями? Может быть. Должны ли вы обновиться? Да, если вы хотите быть уверены, что это сработает.

Предупреждение: Если вы получили сообщение the virtual/mta package conflicts with another package тогда вам необходимо удалить из системы любой другой МТА. Чтобы узнать что это за пакет, просто запустите emerge qmail -p.

Установка qmail так же "потянет" за собой установку ucspi-tcp и daemontools. О них, если хотите, вы можете прочитать на ucspi-tcp и на daemontools. Обычно, daemontools отвечает за управление qmail'ом как сервисом, тогда как ucspi-tcp отвечает за управление входящими TCP соединениями для сервиса qmail.

Для начала проведем несколько пост-инсталяционных мероприятий. :)

Листинг 2.2: Конфигурирование qmail
(Измените под свою конфигурацию)

```
# nano /var/qmail/control/servercert.cnf
# ebuild /var/db/pkg/mail-mta/qmail-1.03-r*/qmail-1.03-r*.ebuild config
```

Qmail проектировался так, чтобы полностью соответствовать требованиям безопасности. Почта никогда не посылается пользователю 'root'. Поэтому вам надо выбрать пользователя на машине, который будет получать почту предназначенную 'root'. С этого момента я предполагаю, что этот пользователь 'vapier'.

Листинг 2.3: Установка не-root акаунта для e-mail.

```
# cd /var/qmail/alias
# echo vapier > .qmail-root
# echo vapier > .qmail-postmaster
# echo vapier > .qmail-mailer-daemon
```

Теперь, поднимем сервис доставки qmail.

Листинг 2.4: Запуск qmail delivery service

```
# rc-update add svscan default
# /etc/init.d/svscan start
# cd /service
# ln -s /var/qmail/supervise/qmail-send qmail-send
```

Надо убедиться, что qmail работает корректно, вот небольшой тест.

Листинг 2.5: Test delivery service

```
# ssh vapier@localhost
# maildirmake .maildir
# qmail-inject root << EOF
test root e-mail!
EOF
# qmail-inject postmaster << EOF
test postmaster e-mail!
EOF
# qmail-inject vapier << EOF
test vapier e-mail!
EOF
```

Виртуальная почтовая система на основе qmail/vpopmail.

mutt

(Теперь в вашем ящике должно быть 3 письма)

Вот и все! Теперь у вас стоит почтовая система способная удовлетворить, потребности локальной машины и системных демонов/пользователей.

Предупреждение: Если вы не получили никаких писем или видите странные ошибки в лог-файлах (проверьте /var/log/qmail/) включающие 'localhost.localhost', это означает что ваша domain/dns информация не корректна. По умолчанию, qmail использует информацию от hostname --fqdn. Если на вашей машине, эта команда выводит 'localhost', тогда проверьте ваши файлы /etc/hostname, /etc/hosts, и настройки dns чтобы убедиться что все правильно. Отредактируйте конфигурационные файлы в каталоге /var/qmail/control/. Используйте их примеры, (там же) если вам нужна помощь.

Листинг 2.6: Пример /var/qmail/control/ файлов для домена второго уровня

```
# hostname --fqdn
wh0rd.org
# cat me
wh0rd.org
# cat defaultdomain
wh0rd.org
# cat plusdomain
wh0rd.org
# cat locals
wh0rd.org
# cat rcpthosts
wh0rd.org
```

Листинг 2.7: Пример /var/qmail/control/ файлов для домена третьего уровня

```
# hostname --fqdn
mail.wh0rd.org
# cat me
mail.wh0rd.org
# cat defaultdomain
wh0rd.org
# cat plusdomain
wh0rd.org
# cat locals
mail.wh0rd.org
# cat rcpthosts
mail.wh0rd.org
```

3. vpopmail

Листинг 3.1: Emerge vpopmail

```
# emerge vpopmail
```

Важно: Эта часть построена вокруг vpopmail-5.4.6 или старше. Будет ли

нижеизложенное работать с ранними версиями? Может быть. Должны ли вы обновиться? Да, если вы хотите быть уверены что это сработает.

vpopmail требует немного больше старания при установке, чем предыдущие пакеты. Мы должны убедиться что он запущен и работает (до mysql). Далее мы можем установить базу данных vpopmail. До того как приступить к следующему шагу, необходимо проверить что mysql корректно установлен. Учтите, пароль для базы данных vpopmail будет 'vpoprw', конечно вы можете выбрать другой.

Листинг 3.2: Установка vpopmail в mysql

```
# rc-update add mysql default
```

Если вы установили mysql в первый раз, убедитесь что была запущена команда `ebuild <mysql.ebuild> config` до запуска mysql сервера.

```
# /etc/init.d/mysql start
```

```
# nano /etc/vpopmail.conf
```

(Изменим пароль с 'secret' на 'vpoprw')

```
# mysql -p << EOF
```

```
create database vpopmail;
```

```
use mysql;
```

```
grant select, insert, update, delete, create, drop on vpopmail.* to vpopmail@localhost identified by 'vpoprw';
```

```
flush privileges;
```

```
EOF
```

(Следующие шаги могут понадобиться, а могут и нет, но мы выполним их просто чтобы быть уверенными)

```
# chown root:vpopmail /etc/vpopmail.conf
```

```
# chmod 640 /etc/vpopmail.conf
```

```
# chown root:vpopmail /var/vpopmail/bin/vchkpw
```

```
# chmod 4711 /var/vpopmail/bin/vchkpw
```

Предупреждение: Если вы испытываете трудности с доступом в mysql/vpopmail, перезапустите mysql. Выполните `/etc/init.d/mysql restart`.

Теперь, vpopmail готов к использованию. В этой части, мы предоставим виртуальный хостинг для домена 'wh0rd.org'. Это означает что мы должны "сообщить" vpopmail об этом домене. Кстати, давайте добавим пользователя 'vapier'.

Листинг 3.3: Добавление домена

(это нужно сделать только в том случае если vadddomain выдает "command not found" (см. далее))

```
# source /etc/profile
```

(при отладке vpopmail, вам может понадобиться просмотр журналов)

```
# mysql -u vpopmail -p
```

```
mysql> select * from vpopmail.vlog;
```

```
# vadddomain wh0rd.org postpass
```

(теперь проверим, правильно ли установлен домен)

```
# printf "postmaster@wh0rd.org\0postpass\0blah\0" | vchkpw `which id` 3<&0
```

```
uid=89(vpopmail) gid=89(vpopmail) groups=0(root)
```

(если вы не видите что-либо подобное, значит права доступа где-то выставлены некорректно)

```
# vadduser vapier@wh0rd.org vappw
```

В каждом домене созданном vpopmail существует аккаунт 'postmaster'. Мы задаем

Виртуальная почтовая система на основе qmail/vpopmail.

пароль postmaster'a в 'postpass'. Чтобы vpopmail был действительно полезен нам, мы должны получать почту через courier и посылать с помощью qmail и SMTP.

4. Courier POP/IMAP

Листинг 4.1: Emerge courier-imap
emerge net-mail/courier-imap

Важно: Сразу после vpopmail вы должны установить courier-imap. Модуль authvchkpw будет встроен.(?)

Важно: Эта часть построена вокруг net-mail/courier-imap-3.0.7 или старше. Будет ли нижеизложенное работать с ранними версиями? Может быть. Должны ли вы обновиться? Да, если вы хотите быть уверены что это сработает.

И еще несколько пост-инсталляционных шагов:). Они необходимы только в том случае если вы будете использовать SSL зашифрованные коммуникации (Рекомендуется!). Иначе вы можете пропустить последние 2 шага, в следующих листингах, удаляя '-ssl' из имени инициализационного скрипта.

Листинг 4.2: Конфигурирование POP3/SSL
nano /etc/courier/authlib/authdaemonrc
(Оставьте в переменной authmodulelist только "authvchkpw")
cd /etc/courier-imap
nano pop3d.cnf
(Отредактируйте секцию [req_dn])
mkpop3dcert
rc-update add courier-pop3d-ssl default
/etc/init.d/courier-pop3d-ssl start

Листинг 4.3: Конфигурирование IMAP/SSL
cd /etc/courier-imap
nano imapd.cnf
(Отредактируйте секцию [req_dn])
mkimapdcert
rc-update add courier-imapd-ssl default
/etc/init.d/courier-imapd-ssl start

Ваш почтовый клиент теперь может соединиться с сервером (с запущенным на нем courier) и получать почту предназначенную для виртуального хоста. В моем случае, я могу соединиться с именем 'varier@wh0rd.org' и паролем 'varpw'.

5. qmail (разговаривая с миром)

Давайте заставим SMTP работать, не причинив ущерба людям, открыв очередную спам-дыру.

Листинг 5.1: Запуск qmail SMTP service

```
# cd /var/qmail/control/
# nano conf-smtpd
(Раскомментируйте SMTP-AUTH переменную и установите QMAIL_SMTP_CHECKPASSWORD
 в /var/vpopmail/bin/vchkpw)
# nano servercert.cnf
(Отредактируйте секцию [ req_dn ] )
# mkservercert
# cd /service
# ln -s /var/qmail/supervise/qmail-smtpd qmail-smtpd
# /etc/init.d/svscan restart
```

Предположим вы не меняли файлов qmail, тогда он теперь принимает почту для виртуального домена wh0rd.org и для пользователей локальной машины. Более того, qmail доверяет любому кто посылает письма через 127.0.0.1 и для любого кто может пройти аутентификацию через vpopmail. При настройке своего почтового клиента для отправки почты, убедитесь что вы включили опцию вроде 'Server requires authentication'. В моем случае, я поставил имя пользователя в 'vapier@wh0rd.org' и пароль - 'varpw'. И последняя деталь - убедитесь что настроили использование SSL/TLS для SMTP в своем клиенте. Qmail не позволит вам пройти аутентификацию если сессия не зашифрована.

6. Веб клиент Horde / IMP

Несмотря на то что существует множество веб-клиентов электронной почты (и вы свободны выбрать любой из них), я предпочитаю IMP Webmail Client который является частью "окружения Horde". Основная причина заключается в том что Horde предоставляет достаточно прозрачный web доступ к почте, и вы можете очень просто добавить другие компоненты, такие как Адресные Книги, Календари, Задачи, и др. И если он до сих пор вас не устраивает, тогда вам следует прочитать Horde.

On to the good stuff! Теперь установим IMP.

Листинг 6.1: Emerge IMP

```
# emerge horde-imp
```

Важно: Эта часть построена вокруг horde-2.2.x and horde-imp-3.2.x. CVS версии (и следующие релизы) скорее всего, будут очень различаться в настройках. Так же имейте ввиду php-5.x, скорее всего, нарушит работу horde-2.x.

Horde не просто настроить если вы точно не знаете что делать. К счастью для вас, я точно знаю что делать, значит и вы будете. :)

Листинг 6.2: Конфигурирование Horde

```
# cd /var/www/localhost/htdocs/horde/config/
# for f in *.dist ; do mv ${f} ${f}.dist ; done
# nano horde.php
```

Виртуальная почтовая система на основе qmail/vpopmail.

(Ниже 'Horde Authentication':)

```
$conf['auth']['driver'] = 'imap';
```

```
$conf['auth']['params']['dsn'] = '{localhost:993/imap/ssl/novalidate-cert}';
```

(Ниже 'Horde Logging':)

```
$conf['log']['name'] = '/var/log/apache2/horde.log';
```

(Ниже 'Problem Reporting':)

```
$conf['problems']['enabled'] = true;
```

```
$conf['problems']['email'] = 'webmaster@wh0rd.org';
```

nano registry.php

(Ниже 'Handlers':)

```
$this->registry['auth']['login'] = 'imp';
```

```
$this->registry['auth']['logout'] = 'imp';
```

(Ниже 'Application registry':) Установите элемент 'status', раздел applications['imp'], из 'inactive' в 'active'

```
# touch /var/log/apache2/horde.log
```

```
# chown apache:apache /var/log/apache2/horde.log
```

Примечание: Вы можете поменять аккаунт 'webmaster' на что-нибудь другое. Если вы хотите использовать этот e-mail адрес, тогда вы должны добавить такого пользователя в vpopmail. (или смотрите ниже о qmailadmin).

Теперь действительно быстро установим IMP.

Листинг 6.3: Конфигурирование IMP

```
# cd /var/www/localhost/htdocs/horde/imp/config/
```

```
# for f in *.dist ; do mv ${f} ${f}.dist ; done
```

```
# nano servers.php
```

(Отредактируйте массив \$servers['imap']

```
$servers['imap'] = array(
    'name' => 'wh0rd.org',
    'server' => 'localhost',
    'protocol' => 'imap/ssl/novalidate-cert',
    'port' => 993,
    'folders' => "",
    'namespace' => 'INBOX.',
    'maildomain' => 'wh0rd.org',
    'smtp host' => 'localhost',
    'realm' => "",
    'preferred' => ""
);
```

Наконец, поднимем apache, чтобы начать использовать webmail.

Листинг 6.4: Запуск apache

```
# nano /etc/conf.d/apache2
```

(Раскомментируйте APACHE2_OPTS="-D SSL -D PHP4")

```
# rc-update add apache2 default
```

```
# /etc/init.d/apache2 start
```

Предупреждение: Вы должны подумать о том, чтобы заставить работать Horde

через https. Это не простая задача, но приглядитесь к ней ради безопасности.

Чтобы проверить установку IMP, запустите браузер и зайдите <http://localhost/horde/> (измените localhost на ваш сервер) Вы должны увидеть страницу приглашения Horde где вы можете войти. Опять же, в моей установке я вхожу с помощью 'vapier@wh0rd.org' и 'vappw' соответственно как логин и пароль.

На данный момент, Horde и IMP установлены. Однако, вы можете пройти заново через все дерево конфигурационных директорий и настроить все что вашей душе угодно.

7. Дополнительные пакеты

qmailadmin

Первый пакет на который я обращаю ваше внимание это qmailadmin. Это веб интерфейс для управления виртуальными доменами. Просто emerge net-mail/qmailadmin а затем укажите вашему браузеру на <http://localhost/cgi-bin/qmailadmin> чтобы начать использовать его. qmailadmin делает жизнь намного проще.

qmHandle

Если вы вляпались в проблемы с очередями qmail и провели много времени над отладкой, то, возможно захотите взглянуть на qmHandle. Это простая перл программа которая позволяет вам видеть и редактировать очередь сообщений qmail. Опять таки все что нужно сделать это emerge net-mail/qmhandle.

Дополнения к horde

Я очень рекомендую посмотреть на множество приложений Horde. Turba, Kronolith, и Nag эти приложения очень хорошо дополняют IMP для разных случаев. Их настройка схожа с настройкой IMP, так что у вас не будет проблем с ними. Просто не забудьте отредактировать registry.php в директории конфигурации horde, и новые приложения появятся внизу странички Horde.

ucspi-tcp

qmail использует ucspi-tcp для фильтрации входящих соединений. Если вы хотите настроить правила фильтрации, тогда смотрите файлы в /etc/tcprules.d/ (в старых версиях qmail эти файлы находятся в /etc). Здесь вы найдете по два файла на каждый сервис, конфигурационный файл (tcp.qmail-smtp) и его откомпилированную версию, ту которую и использует ucspi-tcp (tcp.qmail-smtp.cdb). Каждый раз после обновления конфигурационного файла, надо пересобрать бинарную его версию. Запустите `tcprules tcp.qmail-smtp.cdb tcp.qmail-smtp.tmp < tcp.qmail-smtp`. Каждый раз при соединении с qmail, откомпилированные правила перечитываются с диска, то есть, нет нужды

перезапускать сервис.

qmail-scanner

Если вы хотите обеспечить фильтрацию контента на своем mail сервере (спам, вирусы), тогда вам необходимо использовать другую программу для организации очередей. Одна из хороших программ это qmail-scanner. Наберите emerge qmail-scanner и отредактируйте /etc/tcprules.d/tcp.qmail-smtp.

Важно: Процесс сборки qmail-scanner очень уродлив. Чтобы поддерживать пакеты, они уже должны быть установлены в системе. Это означает что вы должны установить SpamAssassin и/или Clam AntiVirus до qmail-scanner'a. Смотрите подходящие секции ниже.

Листинг 7.1: Смена программы организации очередей.

```
# cd /etc/tcprules.d/  
# nano tcp.qmail-smtp  
(добавьте QMAILQUEUE="/var/qmail/bin/qmail-scanner-queue" в правило catchall)  
# tcprules tcp.qmail-smtp.cdb tcp.qmail-smtp.tmp < tcp.qmail-smtp
```

Смотрите секции об установке спам и вирус фильтров. Вы можете отредактировать несколько опций в /var/qmail/bin/qmail-scanner-queue.pl.

SpamAssassin

Один из лучших спам фильтров это SpamAssassin. Чтобы установить emerge mail-filter/spamassassin. Пакет поставляется двояко, как версия для командной строки и как клиент/серверная. Для серверов через которые проходит небольшое количество почты, использование версии для командной строки себя оправдывает. Но любой у кого большая загрузка серверов должен использовать клиент/серверную версию.

Листинг 7.2: Конфигурирование SpamAssassin

```
# nano /etc/mail/spamassassin/local.cf  
(чтобы обеспечить самый минимум, добавьте следующие опции:)  
required_hits 6  
skip_rbl_checks 1  
  
# rc-update add spamd default  
# /etc/init.d/spamd start  
# nano /var/qmail/bin/qmail-scanner-queue.pl  
(Убедитесь что $spamc_binary variable выставлена в '/usr/bin/spamc'.)  
(Если она выставлена в "", тогда смотрите далее.)
```

Важно: Если у вас не был установлен SpamAssassin до установки qmail-scanner'a, Вы должны будете переустановить qmail-scanner. Процесс его сборки очень ужасен, включены будут только те функции, пакеты для которых уже были установлены.

На этом этапе входящая почта будет проходить через qmail-scanner который в

свою очередь пропустит ее через SpamAssassin.

Clam AntiVirus

Как и SpamAssassin, Clam AntiVirus поставляется в двойном исполнении. Я расскажу как быстро поставить клиент/серверную версию. Во-первых, emerge app-antivirus/clamav.

Листинг 7.3: Конфигурирование Clam AntiVirus

```
# nano /etc/conf.d/clamd
(Установите START_CLAMD=yes)
# nano /etc/clamav.conf
(Настройте все как хотите)
# rc-update add clamd default
# /etc/init.d/clamd start
# nano /var/qmail/bin/qmail-scanner-queue.pl
(Убедитесь что $clamscan_binary переменная установлена
 в '/usr/bin/clamscan'. Если нет, смотрите ниже.)
# nano /var/qmail/control/conf-common
(Если ClamAV сообщает о проблемах с памятью попробуйте увеличить
 программный лимит (softlimit))
```

Важно: Если у вас не был установлен Clam AntiVirus до установки qmail-scanner'a, Вы должны будете переустановить qmail-scanner. Процесс его сборки очень ужасен, включены будут только те функции, пакеты для которых уже были установлены.

На этом этапе входящая почта будет проходить через qmail-scanner который в свою очередь пропустит ее через Clam AntiVirus.

8. Заключительные примечания

У меня нет особых примечаний, но если у вас проблемы с этим руководством, пожалуйста, свяжитесь со мной или разместите сообщение в системе распределения запросов Gentoo. Если у вас есть интересная информация которая вы считаете, может расширить это руководство, то шлите ее мне для включения. Я люблю qmail и с удовольствием добавлю веши, которые могут расширить опыт работы пользователя с mta

Руководство по электронной почте с использованием Mutt

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/guide-to-mutt.xml>

С версии: 1.0

1. Введение в E-Mail

Если вы не любитель e-mail клиентов с замысловатым графическим интерфейсом, или если вы просто хотите поэкспериментировать с другими клиентами, чтобы выбрать подходящий для себя, самым простым способом в вашем случае будет использование связки консольных приложений:

fetchmail->procmail->mutt->smtp

Программы эти не просто мощны и гибко настраиваемы, они также легковесны и эффективны. Потому, однажды настроив и запустив всю эту e-mail систему, вы будете просто ошеломлены тем, что сможете делать с её помощью.

Поскольку это краткое руководство, мы не будем рассматривать службы отправки почты (Mail Transfer Agent), такие как sendmail, postfix или exim. Мы также не будем использовать 25 порт для почтовых служб.

Мы можем себе это позволить, так как fetchmail способен передавать принятую почту непосредственно службе доставки почты (MDA) вместо того чтобы перенаправлять её на 25 порт. И мы можем не использовать службу отправки почты (MTA) для простой передачи почты.

Для запуска своей e-mail системы вам понадобится установить следующее программное обеспечение.

Листинг 1.1: Установка нужных программ
emerge fetchmail procmail mutt nbsmtp

Теперь от полностью рабочей e-mail системы нас отделяют четыре маленьких шажка в направлении конфигурации файлов.

Важно: После каждого шага необходимо тестировать настройку на корректность. Это конечно же подразумевают, что всё необходимое программное обеспечение у вас уже установлено.

2. Fetchmail

Fetchmail забирает почту с удаленного сервера на вашу локальную машину. Для

этого вам необходимо настроить файл `.fetchmailrc` в вашей домашней директории, подобно примеру:

Листинг 2.1: Пример `.fetchmailrc`
`poll mail.myisp.net protocol pop3 user "myname" password "mypassword"`

Сразу после создания файла, необходимо выставить права доступа к нему (он должен быть читаем только его владельцем). Сделать это можно следующей командой:

Листинг 2.2: Изменение прав доступа
`# chmod 710 .fetchmailrc`

Чтобы увидеть процесс в действии, используйте ключ `-v`. Чтобы получить все сообщения, используйте `-a`. И ещё вы должны использовать ключ `-m` для того, чтобы передать принятую почту `procmail`.

Предупреждение: Будет также хорошей идеей использовать ключ `-k`, в этом случае если что-то пойдет не так, ваша почта не будет удалена с сервера и вы сможете её повторно забрать.

Пришло время проверить `fetchmail` в действии!

Листинг 2.3: Тест `Fetchmail` #1
`# fetchmail -akv -m "/usr/bin/procmail -d %T"`

Как только вы добьётесь работающей системы, вы можете добавить процесс в `cron` или какой-нибудь монитор вроде `gkrellm`. `Fetchmail` также может быть запущен как демон с указанием секундного интервала.

3. *Procmail*

`Procmail` - это программа фильтрующая почту получаемую от `fetchmail`. А далее, также как и `MDA`, она поставляет отфильтрованную почту в ваши почтовые ящики, откуда её уже можно прочитать программой `mutt` (это почтовый клиент, который мы будем использовать).

Для использования `procmail`, также необходимо создать файл `.procmairc` в своём домашнем каталоге. Для наших целей "быстрой настройки" мы будем использовать простой `.procmairc`, который фильтрует почту от трех списков рассылки `gentoo` в три почтовых ящика: `gentoo-dev`, `gentoo-user` и `gentoo-announce`. Примечание: Правила фильтра называются условиями. Я также добавил условия для того, чтобы отфильтровать некоторый спам.

Листинг 3.1: Пример .procmairc

```
MAILDIR=$HOME/MuttMail          ##проверьте правильность пути
5LOGFILE=$HOME/.procmaillog
LOGABSTRACT=no
#VERBOSE=on... используется только для отладки
VERBOSE=off
FORMAIL=/usr/bin/formail
NL=""
"
##условные строки начинаются с :0
##не записывайте комментарии в строки условия
##отредактируйте ненужные условия!
##строки условий начинаются с *, а регулярные выражения ваши лучшие друзья
##условия добавленные после * попадают прямо в egrep
##строка следующая за условиями, в следующем регистре является именем почтового ящика

#отлавливание копий, используя formail
:0 Whc: .msgid.lock
| $FORMAIL -D 16384 .msgid.cache

:0 а
$MAILDIR/duplicates

#люди которые всегда пишут с одного почтового адреса
:0
* ^From:.*(craig\@hotmail|renee\@local.com)
$MAILDIR/friends

#выборка некоторого спама
:0
* ^Subject:.*(credit|cash|money|debt|sex|sale|loan)
$MAILDIR/spam

#никаких html писем
:0
* ^Content-Type:.*html
$MAILDIR/junk

#складировать письма из списков рассылки в мои почтовые ящики
:0
* ^List-Id:.*gentoo-user
gentoo-user

:0
* ^List-Id:.*gentoo-dev
gentoo-dev

:0
* ^List-Id:.*gentoo-announce
gentoo-announce

#получать любую другую почту с gentoo
:0
* ^From:.*gentoo.org
gentoo

:0
```

```
* ^From:.*@freshmeat\.net
freshmeat

#####
# последнее условие: складывает остальную #
# почту в почтовый ящик по умолчанию #
#####
:0
* *
.
default

# конец файла
```

Примечание: В данном случае вам только потребуется создать почтовую директорию в \$HOME/MuttMail, после чего Procmail создаст все необходимые файлы почтового ящика в этом каталоге, используя названия из строк условий. Для дополнительной информации посетите <http://www.procmail.org/>

Для проверки нашего .procmailrc, повторно запустите fetchmail (который мы уже настроили). Помните также, что опция -k оставляет почту на удаленном сервере, потому её можно использовать для нашего теста.

Листинг 3.2: Тест Procmail #1
fetchmail -akv -m "/usr/bin/procmail -d %T"

Ну всё, теперь когда fetchmail и procmail работают, мы можем зайдя в \$HOME/MuttMail прочитать нашу почту программой less или вашим любимым файловым менеджером.

4. Почтовый клиент Mutt

Mutt используется для чтения и написания писем. Это очень мощное, сложно-настраиваемое, легковесное и эффективное приложение.

Mutt поддерживает чтение и запись в различных форматах почтового ящика: mbox, MMDF, MH и Maildir. Тип почтового ящика распознается автоматически. В нашем случае мы используем формат mbox, где все сообщения почтового ящика сохраняются в отдельных файлах.

Mutt также имеет способность работать с папками, расположенными на удаленном IMAP сервере. См. Поддержку IMAP в разделе 4.11 руководства по Mutt и сайт Mutt <http://www.mutt.org/>.

При установке mutt создаётся основной файл конфигурации /etc/mutt/Muttrc. Вам также необходимо создать файл .muttrc в своём домашнем каталоге.

Листинг 4.1: Пример .muttrc

(Конечно, неплохо бы прочитать документацию Mutt из /usr/share/doc/mutt*)

(Любые настройки здесь отменяют параметры общей конфигурации из /etc/mutt/Muttrc)

```
# cp /etc/mutt/Muttrc ~/.muttrc
# nano -w .muttrc
set pager_context=1
set pager_index_lines=6          #показывать оглавления в окне программы
set menu_scroll
set pgp_verify_sig=no           #не показывать pgr на странице
set status_on_top               #разместить статусную строку вверху
set sort=threads                #сортировать сообщения по заголовкам

set status_format="%r %b %f %n   Del %d   Msgs %m %l %> (%P)"
set pager_format="%-10i %[%a %b %d %R]"
set date_format="!%H:%M %a %d %b   "
set index_format="%4C %Z %[%b%d] %-15.15F %s"
set folder_format="%2C %t %8s %d %N %f"

#set sendmail="/usr/bin/nbsmtp -d isp.net -h smtp.isp.net -f yourname@isp.net"

#set from="default-mailaddress"   #задаёт ваш адрес в строке "from"
#set realname="myname"

set record="$HOME/MuttMail/sent"  #сохранять отправленную почту здесь
set delete=yes                   #удалить без подтверждения
set include=yes                  #выделять сообщение в ответе
set fast_reply=yes               #не подтверждать ответ
set beep=no                      #не пищать
set markers=no                  #не помечать + сложенные строки
set confirmappend=no             #не подтверждать сохранение в =keep
set to_chars=" +TCF"             #нет L для mail_list

set folder = $HOME/MuttMail
mailboxes =gentoo-user
mailboxes =gentoo-dev
mailboxes =gentoo-announce
mailboxes =gentoo
mailboxes =freshmeat
mailboxes =duplicates
mailboxes =default
mailboxes =sent
mailboxes =friends
mailboxes =junk
mailboxes =spam
mailboxes =keep

save-hook .* =keep               #mbox по умолчанию сохраняет (s) почту в =keep
subscribe gentoo-user gentoo-dev #подписанные списки

bind pager h display-toggle-weed #переключать заголовки кнопкой h

# симулировать старое url меню
macro index \cb |urlview\n 'call urlview to extract URLs out of a message'
macro pager \cb |urlview\n 'call urlview to extract URLs out of a message'

#запуск fetchmail нажатием кнопки G
macro index G "!fetchmail -a -m 'procmail -d %T'\r"
macro pager G "!fetchmail -a -m 'procmail -d %T'\r"

#редактирование .muttrc... не требует перезапуска
```



```

macro generic ,sm ":source $HOME/.muttrc\r"
macro generic \cj "!rxvt -bg wheat -e joe $HOME/.muttrc\r"

#по умолчанию список заголовков в полях удаляется перед показом почты
#игнорирует всё, кроме того, что вам нужно
ignore *
unignore Date To From: Subject X-Mailer Organization User-Agent
hdr_order Date From To Subject X-Mailer User-Agent Organization

##ваш Mutt должен поддерживать несколько цветов
##для обозначения четырех уровней выделенного текста
##данные настройки отменяют параметры общей конфигурации в /etc/mutt/Muttrc

#color quoted green default
color quoted1 magenta blue
#color quoted2 yellow default
#color quoted3 red default
#color signature cyan cyan

#эта цветовая схема взята из /etc/mutt/Muttrc.color
#закомментируйте её, если вам нужна цветовая схема по умолчанию из /etc/mutt/Muttrc
# Je vois la vie en rose :)
color hdrdefault brightcyan blue
color header brightwhite blue "^from:"
color header brightwhite blue "^subject:"

color quoted brightgreen blue
color signature brightwhite blue

color indicator blue green

color error red black
mono error bold
color status black cyan
mono status bold
color tree yellow blue

color tilde brightmagenta blue
color body brightwhite blue "[-a-z_0-9.]+"@[-a-z_0-9.]+"
mono body bold "[-a-z_0-9.]+"@[-a-z_0-9.]+"
color body brightyellow black "^Good signature"
mono body bold "^Good signature"
color body brightwhite red "^Bad signature from.*"
mono body bold "^Bad signature from.*"
color normal white blue
color message green black
color attachment brightgreen blue

# конец файла... но вы можете его дописывать и дописывать... :)

```

Это только маленький пример файла .muttrc. На самом же деле гораздо больше опций поддаются конфигурации, например, те же настройки gpg. Для примеров и помощи посмотрите <http://mutt.netliberte.org/>.

Теперь вы можете протестировать наш .muttrc

Листинг 4.2: Тест .muttrc
mutt -y

После чего должно появиться окно Mutt с почтовыми ящиками, которые мы создали, когда настраивали fetchmail.

Нажмите ? для получения помощи в навигации по почтовым ящикам в Mutt.

5. SMTP

Последний шаг, это настройка nbsmtp ('No-Brainer SMTP'), используемого для отправки почты на ваш сервер SMTP. Данная настройка самая простая и требует лишь добавления нескольких строк в конфигурационный файл .muttrc

domain: домен, сообщаемый nbsmtp. Будет почти всегда похож на окончание вашего адреса электронной почты.

from@addr: Это тот адрес, который будет сообщаться nbsmtp в строке "from". Обратите внимание, что данный адрес может отличаться от того, что записан в поле "From:" вашего почтового клиента (MUA).

host: Сервер smtp, куда, собственно, и будет отправляться почта.

Листинг 5.1: Добавление поддержки smtp
nano -w .muttrc
set sendmail="/usr/bin/nbsmtp -d isp.net -h smtp.isp.net -f urname@isp.net"

Теперь всё готово для создания письма. В окне Mutt нажмите m, для того чтобы написать текстовое сообщение на ваш же почтовый ящик. Mutt использует значение EDITOR или VISUAL, указываемое с помощью опций editor= в .muttrc. После того, как сообщение будет написано, нажмите u для его отправки. Если всё прошло удачно, мы увидим сообщение 'sending mail', следующее за 'New mail in =sent'.

Помните, в .muttrc у нас задано сохранять всю отправленную почту при помощи строки :set record="\$HOME/MuttMail/sent"

Теперь, чтобы завершить испытание, снова запустите fetchmail для получения всей почты и вашего тестового письма, которое вы себе отправили. Как только это тестовое письмо будет найдено, нажмите h для того, чтобы посмотреть все его заголовки и полный путь прохождения почты (mail transfer path).

Примечание: Есть ещё одна программа, которая, возможно, вам пригодится, под названием urlview. Она извлекает ссылки из текстовых сообщений и перенаправляет их в ваш браузер.

Листинг 5.2: Установка urlview
emerge urlview

После чего создайте ~/.urlview копируя конфигурационный файл из /usr/share/doc/urlview*/, и подправьте необходимые настройки под свой браузер.

Чтож, теперь у нас есть мощная почтовая система. Читайте различную документацию и руководства, а также ищите примеры конфигурационных файлов в 'google' по ключевым словам muttrc и procmailrc.

Руководство по файловой системе для устройств

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/devfs-guide.xml>

С версии: 1.0

1. Что такое devfs?

Старые (добрые?) времена

Традиционная реализация Linux предоставляет пользователю абстрактный каталог устройств, называемый `/dev`. Внутри этого каталога находятся `device nodes`, специальные файлы, символизирующие устройства внутри системы. Например, `/dev/hda` символизирует первое IDE устройство в вашей системе. Файлы устройств позволяют создавать программы, которые взаимодействуют с устройствами так, как если бы устройства были обычными файлами, вместо использования специальных API.

Файлы устройств подразделяются на две группы, называемые `character devices` (символьные устройства) и `block devices` (блочные устройства). Первая группа содержит устройства, для которых отсутствует буферизация чтения/записи. Вторая группа, естественно, содержит устройства, для которых чтение/запись буферизируется. Из обоих типов устройств может быть прочитан за раз один символ или блок. Поэтому такой способ присваивания имён может сбивать с толку и на самом деле неправилен.

Если посмотреть на конкретное устройство, можно обнаружить что-то, похожее на это:

Листинг 1.1: Исследование информации о файле устройства

```
# ls -l /dev/hda
```

```
brw-rw---- 1 root disk 3, 0 Jul 5 2000 /dev/hda
```

В предыдущем примере мы увидели, что `/dev/hda` — это блочное устройство. Однако важнее то, что ему присвоено два специальных номера 3, 0. Эта пара называется `major-minor`. Она используется ядром, чтобы соотнести файл устройства и реальное устройство. `major` (старший) относится к типу устройства, `minor` (младший) к конкретному устройству. Выглядит запутано, не правда ли?

Ещё два примера — `/dev/hda4` и `/dev/tty5`. Первое устройство соответствует четвёртому разделу на первом IDE-устройстве. Его пара `major-minor` 3, 4. Другими словами, `minor` соответствует разделу, тогда как `major` соответствует устройству. Во втором примере пара `major-minor` — 4, 5. В этом случае первое число соответствует драйверу терминала, тогда как второе соответствует номеру терминала (в данном случае пятый терминал).

Проблемы

Если вы заглядывали в папку `/dev`, вы обнаружили, что там перечислены не только все ваши устройства, но и все возможные устройства, которые только могут быть. Другими словами, у вас есть файлы устройств для устройств, которых у вас нет. Управление такой кучей устройств по крайней мере можно назвать громоздким. Представьте, что надо поменять права на все устройства, реально существующие в вашей системе, оставив остальные без изменений.

Затем вы добавили новое оборудование в вашу систему, для него может не оказаться уже существующего файла. Продвинутые пользователи знают, что эта задача может быть выполнена с помощью команды `./MAKEDEV` внутри директории `/dev`, но разве вы сразу знаете, что за устройство вам придётся создать?

Когда у вас есть программы, взаимодействующие с оборудованием при помощи файлов устройств, вы не можете смонтировать корневой раздел только для чтения, в то время как в дальнейшем нет необходимости в том, чтобы он был смонтирован на чтение и запись. И вы не можете иметь `/dev` на отдельном разделе, так как `mount` необходим, `/dev` чтобы монтировать разделы.

Решения

Как вы могли себе представить, kernel hackers нашли достаточно решений для вышеперечисленных проблем. Однако многие из этих решений имеют собственные проблемы описанные в <http://www.atnf.csiro.au/people/rgooch/linux/docs/devfs.html#faq-why>. Мы не будем обсуждать эти варианты, а сконцентрируемся на одном способе, который был реализован в официальной версии исходников ядра:

devfs как абсолютный победитель?

devfs решает все перечисленные проблемы. Она просто предоставляет пользователю доступ к существующим устройствам, добавляет новые device nodes (файлы устройств), когда найдены новые устройства, и делает возможным монтировать корневую файловую систему в режиме `read only` (только чтение). А также решает многие проблемы, которые мы раньше не обсуждали, потому что они не так интересны для пользователей...

Как пример, с `devfs` вам не надо беспокоиться о паре `major/minor`. Она продолжает поддерживаться (для обратной совместимости), но в ней нет необходимости. Это позволяет Linux поддерживать ещё больше устройств, так как больше нет

ограничений (числа всегда имеют границы :)

Однако у `devfs` есть свои проблемы, не столь очевидные для пользователей, но достаточно серьезные, чтобы разработчики ядра пометили ее как `obsolete` (устаревшее), порекомендовав использовать `udev`, которая также поддерживается Gentoo.

Чтобы узнать, почему `devfs` считается устаревшей, читайте `udev FAQ` и `udev versus devfs document`.

2. Навигация через дерево устройств

Директории

Одна из первых особенностей которые вы можете заметить это то что `devfs` использует директории для объединения устройств вместе. Это повышает читабельность, так как теперь все связанные между собой устройства находятся внутри одной общей директории.

Например, все устройства, относящиеся к IDE, находятся в директории `/dev/ide/`, а все относящиеся к SCSI в директории `/dev/scsi/`. SCSI и IDE диски во многом похожи, у них одинаковая структура поддиректорий.

IDE и SCSI диски управляются при помощи адаптера (встроенного или отдельной платой), называемого `host`. Каждый адаптер может иметь несколько каналов. Канал называется `bus`. На каждом канале может быть несколько IDs (идентификаторов). ID служит для идентификации диска. Этот ID называется `target`. Многие SCSI устройства могут иметь множество LUN (Logical Unit Numbers (Номер Логического Устройства)), Например устройства которые управляют несколькими носителями одновременно (hi-end tapedrives). У вас скорее всего будет только один `lun`, `lun0/`.

Итак, несмотря на то, что раньше использовался `/dev/hda4`, теперь появился `/dev/ide/host0/bus0/target0/lun0/part4`. Это намного проще... нет, не спорьте со мной... это проще... как бы то ни было! :)

Примечание: Вы также можете использовать более похожие на Unix названия для жёстких дисков, такие как `с0b0t0u0p2`. Они могут быть найдены в `/dev/ide/hd`, `/dev/scsi/hd` и.т.д.

Чтобы дать вам лучше понять идею с директориями, вот листинг директорий которые есть у меня:

Листинг 2.1: Директории в `/dev`

```
cdroms/  cpu/    discs/   floppy/
ide/     input/  loop/    misc/
netlink/ printers/ pts/      pty/
scsi/    sg/     shm/     sound/
sr/      usb/    vc/      vcc/
```

Обратная совместимость при помощи devfsd

Использование этой новой структуры выглядит здорово, но многие утилиты и программы используют предыдущую, старую структуру. Для уверенности, что система не будет нарушена, был создан devfsd. Этот демон создаёт символьные ссылки на новые файлы устройств, но со старыми именами (compatibility symlinks).

Листинг 2.2: Созданные символьные ссылки

```
$ ls -l /dev/hda4
```

```
lr-xr-xr-x  1 root  root      33 Aug 25 12:08 /dev/hda4 -> ide/host0/bus0/target0/lun0/part4
```

При помощи devfsd, вы можете устанавливать права доступа, создавать новые файлы устройств и т.д. Всё это описывается в следующей главе.

3. Администрирование дерева устройств

Перезагрузка devfsd

Если вы изменили файл /etc/devfsd.conf, и хотите чтобы изменения вступили в силу, вам не обязательно перезагружаться. В зависимости от того, что вы хотите, вы можете использовать любой из следующих сигналов:

SIGHUP заставит devfsd перечитать конфигурационный файл, перегрузить разделяемые объекты (shared objects) и сгенерировать событие REGISTER для каждого листа в дереве устройств.

SIGUSR1 сделает то же самое, но не будет событий REGISTER.

Чтобы послать сигнал, просто используйте kill или killall:

Листинг 3.1: Посылка сигнала SIGHUP демону devfsd

```
# kill -s SIGHUP `pidof devfsd`
```

или

```
# killall -s SIGHUP devfsd
```

Удаление compatibility symlinks

Предупреждение: В настоящее время Gentoo не может существовать без этих ссылок.

Если вы хотите удалить из вашей системы ссылки которые засоряют /dev (в Gentoo они используются по умолчанию), отредактируйте /etc/devfsd.conf и удалите следующие две строки:

Листинг 3.2: /etc/devfsd.conf для обратной совместимости
Закомментируйте эти две строки для удаления симлинков
REGISTER . * MKOLDCOMPAT
UNREGISTER . * RMOLDCOMPAT

Вам придётся перезагрузиться, чтобы изменения вступили в силу.

Удаление возможности автосоздания файлов устройств

Когда вы загружаете модуль, devfs автоматически создаёт файлы устройств. Если вы не хотите, чтобы он так делал, удалите эту строку из /etc/devfsd.conf:

Листинг 3.3: /etc/devfsd.conf, autoload functionality
LOOKUP . * MODLOAD

4. Вопросы, относящиеся к правам доступа

Установка/изменение прав доступа при помощи PAM

Хотя вы можете установить права доступа в /etc/devfsd.conf, мы советуем использовать PAM (Pluggable Authentication Modules []). Так как PAM имеет решающий голос при установке прав доступа, и может проигнорировать изменения, которые вы сделали в /etc/devfsd.conf.

PAM использует /etc/security/console.perms для установки прав доступа. Файл состоит из двух частей: в первой описываются группы, а во второй права.

Давайте сначала взглянем на часть с группами. Как пример мы рассмотрим sound-group:

Листинг 4.1: Sound group в /etc/security/console.perms
<sound>=/dev/dsp* /dev/audio* /dev/midi* \
/dev/mixer* /dev/sequencer* \
/dev/sound/* /dev/snd/* /dev/beep \
/dev/admm* \
/dev/adsp* /dev/aload* /dev/amidi* /dev/dmfm* \
/dev/dmmidi* /dev/sndstat

Синтаксис достаточно прост: вы начинаете с имени группы, и заканчиваете списком устройств, принадлежащих этой группе.

Теперь для того, чтобы с группами можно было что-нибудь сделать, рассмотрим следующую часть, описывающую, как управлять правами.

Листинг 4.2: Права доступа для sound group в /etc/security/console.perms

```
<console> 0600 <sound>    0600 root.audio
```

- Первое поле — это проверка терминала. На большинстве систем это console-group. PAM будет проверять это поле при каждом входе в систему. Если вход произошёл на устройстве, содержащемся в console-group, PAM проверит и возможно сменит права на некоторые файлы устройств.
- Второе поле содержит права, которые установятся на файл устройства после удачного входа в систему. Когда человек вошел в систему, а файлы устройств принадлежат пользователю и группе по умолчанию, PAM сменит владельца на вошедшего пользователя и установит на них права из второго поля. В данном случае используется 0600 (пользователь имеет право на чтение/запись, все остальные нет).
- В третьем поле содержатся группы устройств, чьи права будут изменены. В данном случае, вторая группа (все устройства, относящиеся к звуку) будут изменены.
- Четвёртое поле определяет права, которые будут установлены на файлы устройств после возврата в состояние по умолчанию. Другими словами, если человек, который владеет правами на все файлы устройств, выйдет из системы, PAM установит права обратно в состояние по умолчанию, описанному в этом четвёртом поле.
- Пятое поле определяет собственника (с группой если вам надо) к которому будут установлены атрибуты устройства после возврата в состояние по умолчанию. Другими словами, если человек, владеющий правами на все файлы устройств, выйдет из системы, PAM установит собственника обратно в состояние по умолчанию, описанному в пятом поле.

Установка/изменение прав доступа при помощи devfsd

Если вы действительно хотите установить права, используя /etc/devfsd.conf, тогда используйте синтаксис приведённый в этом примере:

Листинг 4.3: Права в /etc/devfsd.conf

```
REGISTER ^cdroms/. * PERMISSIONS root.cdrom 0660
```

Второе поле — это группа устройств, начиная с /dev. Это регулярное выражение, означающее, что вы можете выбрать несколько файлов устройств с одним правилом.

Четвёртое поле — это владелец файла устройства. В отличие от PAM, он не

изменяется (если он не упоминается в `console.perms`, так как PAM главнее).
Пятое поле содержит права на файлы устройств.

Ручная установка прав и их сохранение devfsd

Это обычная ситуация для Gentoo: если вы делаете `chown` (CHange OWNer (Смена владельца)) и `chmod` (CHange MODe (Смена вида)) некоторым файлам устройств, то `devfsd` сохраняют информацию, когда вы выключаете систему. Это происходит из-за того, что файл `/etc/devfsd.conf` содержит следующие строки:

Листинг 4.4: `/etc/devfsd.conf` для сохранения прав доступа

```
REGISTER    ^pt[sy]/.*  IGNORE
CHANGE      ^pt[sy]/.*  IGNORE
CREATE      ^pt[sy]/.*  IGNORE
DELETE      ^pt[sy]     IGNORE
REGISTER    ^log       IGNORE
CHANGE      ^log       IGNORE
CREATE      ^log       IGNORE
DELETE      ^log       IGNORE
REGISTER    .*         COPY    /lib/dev-state/$devname $devpath
CHANGE      .*         COPY    $devpath /lib/dev-state/$devname
CREATE      .*         COPY    $devpath /lib/dev-state/$devname
DELETE      .*         CFUNCTION GLOBAL unlink
/lib/dev-state/$devname
RESTORE      /lib/dev-state
```

Другими словами, изменённые файлы устройств копируются в `/lib/dev-state`, когда выключается система, и копируются в `/dev`, когда система грузится.

Другая возможность - монтировать `/lib/dev-state` в `/dev` во время загрузки. Чтобы это сделать, вы должны быть уверены, что `devfs` не монтируется автоматически (это значит, что вы должны перекомпилировать ядро), и что `/dev/console` существует. Затем, где-то в начале `bootscripts` (загрузочных скриптов) вашей системы, вы должны разместить:

Листинг 4.5: Монтирование `/lib/dev-state` в `/dev`

```
mount --bind /dev /lib/dev-state
mount -t devfs none /dev
devfsd /dev
```

Ebuild HOWTO (Англ.)

Ссылка на оригинал: <http://www.gentoo.org/proj/en/devrel/handbook/handbook.xml?part=2&chap=1>

С версии: 1.4

1.a. The Portage tree

Introduction

The Portage tree is typically found at /usr/portage and is organized in a hierarchical structure consisting of category directories, followed by specific package directories. Here's an example; you can find the util-linux-2.11y.ebuild file in the /usr/portage/sys-apps/util-linux directory. There may be several other versions of util-linux ebuilds alongside util-linux-2.11y.ebuild. This is because all ebuilds for a particular package (regardless of version), share the same mycat/mypkg directory in /usr/portage.

Checking Out the Portage Tree from CVS

If you are unfamiliar with the CVS system, please read the [CVS Tutorial](#) for more information.

The Portage tree can be found in the gentoo-x86 module of the Gentoo Linux tree. To check out the module (about 350 megabytes) you would first set up CVS via the above guide, then check out the gentoo-x86 module.

What (not) to put in the Portage tree

Before writing a new ebuild, check bugs.gentoo.org to see if an ebuild has already been written for the package, but has not yet been added to the Portage tree. Go to bugs.gentoo.org, choose query and select Advanced Search; as product select Gentoo Linux, as component select ebuilds. In the search field put the name of the ebuild and as status select NEW, ASSIGNED, REOPENED and RESOLVED (RESOLVED is important), then submit the query. For you lazy people, click [here](#).

In general, the Portage tree should only be used for storing .ebuild files, as well as any relatively small companion files, such as patches or sample configuration files. These types of files should be placed in the /usr/portage/mycat/mypkg/files directory to keep the main mycat/mypkg directory uncluttered. Exceptions to this rule are for larger patch files (we recommend this for patches above 20KB) which should be put onto the Gentoo mirrors so that people do not waste excessive amounts of bandwidth and hard drive space. Also, you should not add binary (non-ASCII) files to the Portage CVS tree. If you need to do this in another CVS tree, for example, if you need to add a small PNG graphic for whatever reason, be sure to add it to CVS by using the -kb option, like so:

Code Listing 1.1: Adding binary files to CVS

```
# cvs add -kb myphoto.png
```

The -kb option tells CVS that myphoto.png is a binary file and should be treated

specially. For example, merging the differences between two different versions of this file should not be allowed to happen, for obvious reasons. Also, speaking of merging changes, any patches you add to Portage should generally not be compressed. This will allow CVS to merge changes and correctly inform developers of conflicts.

Remember, the packages that you commit must be ready out of the box for end users when committed as stable. Make sure that you have a good set of default settings that will satisfy the majority of systems and users that will use your package. If your package is broken, and you are not sure how to get it to work, check some other distributions that have done their own versions of the package. You can check [Mandriva](#) or [Debian](#) or [Fedora](#) for some examples.

When committing to CVS, all developers should use `repoman commit` instead of `cvs commit` to submit their ebuilds. Before committing, please run `repoman full` to make sure you didn't forget something.

CVS Commit Policy

- Always run `repoman scan` before you commit.
- Please run `repoman full` before you commit.
- Always test that `package.mask` is okay by doing `emerge --pretend mypkg` before you commit and check that it doesn't contain any conflicts.
- Always update the `ChangeLog` before you commit.
- Always commit the updated `package.mask` before the updated package, in case conflicts occur while you commit `package.mask`.
- Always do atomic commits; if you commit a package with a new license, or that is masked, then first commit the revised `package.mask` and/or license, then commit the ebuild, `ChangeLog`, patches and [metadata.xml](#) all in **one** go to avoid breaking users' installations.

The files Directory

As noted earlier, under each package subdirectory is a `files/` directory. Any patches, configuration files, or other ancillary files your package might require should be added to this directory; any files bigger than 20KB-or-so should go to the mirrors to lower the amount of (unneeded) files our users have to download. You may want to consider naming patches you create yourself just to get your package to build with a version-specific name, such as `mypkg-1.0-gentoo.diff`, or more simply, `1.0-gentoo.diff`. Also note that the gentoo extension informs people that this patch was created by us, the Gentoo Linux developers, rather than having been grabbed from a mailing list or somewhere else. Again, you should not compress these patches because CVS does not play well with binary files.

Consider prefixing or suffixing (such as `mypkg-1.0`) every file you put into the `files/` directory, so that the files used for each individual version on an ebuild are distinguishable from one another, and so that the changes between different revisions are visible. This is generally a really good idea :). You may want to use a different suffix if you wish to convey more meaning with the patch name.

If you have many files that should go into the `files/` directory, consider creating subdirectories such as `files/1.0/` and putting the relevant files in the appropriate subdirectory. If you use this method, you do not need to add version information to the names of the files, which is often more convenient.

1.b. Ebuild scripts

Introduction

Ebuild scripts are the basis for the entire portage system. They contain all the information required to download, unpack, compile and install a set of sources, as well as how to perform any optional pre/post install/removal or configuration steps. While most of Portage is written in Python, the ebuild scripts themselves are written in bash, since using bash allows us to call commands as we would from the command-line. One of the important design principles behind ebuild scripts is to have the commands therein be analogs of those one would type on the command-line if installing the package manually. For this purpose, using bash syntax is a very good thing.

Ebuild scripts are interpreted by the ebuild and emerge commands. Think of the ebuild command as a low-level building tool. It can build and install a single ebuild, but no more. It will check to see if dependencies are satisfied, but it will not attempt to auto-resolve them. On the other hand emerge is a high level engine for ebuild, and has the ability to auto-merge dependencies if needed, perform pretend merges so that user can see what ebuilds would be merged, and more. Generally, emerge blows ebuild out of the water except in one area. With ebuild, you can incrementally step through the various parts of a package emerge (fetching, unpacking, compiling, installing and merging) one at a time. For developers, this is an invaluable debugging tool, because it allows you to isolate ebuild problems to a specific portion of the ebuild.

Naming ebuild files

Ebuild file names consist of four logical subsections:

pkg-ver{_suf{#}}{-r#}.ebuild

Note: The brackets ({}) delineate optional fields and do not appear in the literal package name. # represents any non-zero positive integer.

The first subsection, pkg, is the package name, which should only contain lowercase letters, the digits 0-9, and any number of single hyphen (-), underscore (_) or plus (+) characters. Examples: util-linux, syslogd and gtk+. We have some packages in Portage that don't follow these rules, but your packages should.

The second subsection, ver, is the version of the package, which should normally be same as the version on the main source tarball. The version is normally made up of two or three (or more) numbers separated by periods, such as 1.2 or 4.5.2, and may have a single letter immediately following the last digit; e.g., 1.4b or 2.6h. The package version is joined to the package name with a hyphen. For example: foo-1.0, bar-2.4.6.

Important: If you're thinking of using a trailing letter in your version string, note that the trailing letter should not be used to signify alpha or beta status for the package, since alphas and betas are prereleases and letter revisions are newer versions. This is an important distinction because Portage uses an ebuild's version number to determine if it is newer or older than other packages with the same category and name. It's very important that version numbers faithfully represent the version of the package so that Portage properly performs its dependency checking duties.

The third subsection, {_suf{#}}, is optional may contain one of these predefined suffixes, listed in least-recent to most-recent order:

Suffix	Meaning
_alpha	Alpha release

<code>_beta</code>	Beta release
<code>_pre</code>	Prerelease
<code>_rc</code>	Release candidate
(none)	Normal release
<code>_p</code>	Patch level (normally accompanied by trailing integer)

Any of these suffixes may be immediately followed by a non-zero positive integer, e.g., `linux-2.4.0_pre10`. Assuming identical version parts, the suffixes are ordered as follows (lower means older): `_alpha` < `_beta` < `_pre` < `_rc` < (no suffix) < `_p`.

When comparing identical suffixes with trailing integers, the one with the larger integer will be considered most recent. Example: `foo-1.0_alpha4` is more recent than `foo-1.0_alpha3`.

The fourth subsection of the package name is the Gentoo Linux-specific revision number (`{-r#}`). This subsection, like the suffix, is also optional. `#` is a non-zero positive integer; e.g., `package-4.5.3-r3`.

This revision number is independent of the version of the source tarball and is used to inform people that a new and improved Gentoo Linux revision of a particular package is available. Initial releases of ebuilds must have no revision number; e.g., `package-4.5.3` and are considered by Portage to have a revision number of zero. This means that counting goes as follows: 1.0 (initial version), 1.0-r1, 1.0-r2, etc.

If you make non-trivial improvements to an existing ebuild file, you should copy the ebuild file to a new file with the revision number incremented by 1. Remember to always make mentions of your changes in the ChangeLog when you bump a revision **and** in your CVS commit message; not doing so is against policy.

... and I suppose that we actually have a fifth section of the ebuild name as well -- the `.ebuild` extension itself.

Contents of an ebuild file

This section is an introduction to ebuilds. For the full listing of everything possible in an ebuild, there is a manpage which talks about the internal format, variables, and functions in an ebuild script: `man 5 ebuild`.

Headers

When you submit your ebuilds, the header should be exactly the same as the one in `/usr/portage/header.txt`. Most importantly, do not modify it in anyway and make sure that the `$Header: $` line is intact.

The first three lines should look something like this:

Code Listing 2.1: Valid Header

```
# Copyright 1999-2005 Gentoo Foundation
# Distributed under the terms of the GNU General Public License v2
# $Header: $
```

Variables

The first part of every ebuild file is made up of a number of variables. They fall under 3 categories (and are marked below):

- **READ:** variables you can utilize but never set

- **MUST:** variables you must always set
- **OPT:** variables that you should set

Variable	Usage	Description
P	READ	The name and version of the package.
PN	READ	The name of the package.
PV	READ	The version of the package.
PR	READ	Contains the revision number or r0 if no revision number exists.
PVR	READ	Contains the version number with the revision.
PF	READ	Contains the full package name \${PN}-\${PVR}.
A	READ	Space delimited list of the filenames in SRC_URI. This does not contain the URL paths, just the filename.
DISTDIR	READ	Contains the path to the distfiles directory where all the files fetched for a package are stored. Typically, this is /usr/portage/distfiles.
FILESDIR	READ	Contains the path to the files sub folder in the package specific location in the portage tree. Do not modify this variable.
WORKDIR	READ	Base of the build root for the ebuild. Nothing should be built outside of this directory.
S	OPT	The source directory for your package; commonly \${WORKDIR}/\${P}. Portage will default to this value so you may not have to set it!
T	READ	The temporary directory for your package. It is used as a virtual /tmp directory while processing the ebuild.
D	READ	The root directory that the package is installed to, treat it as the virtual /.
SLOT	MUST	Portage handles different versions of the same installed programs. If you would want, say GCC 2.95 and GCC 3.2 installed at the same time, you would specify the SLOT in each ebuild. Here we would set the SLOT of GCC 2.95 to 2 while we would set the SLOT of GCC 3.2 to 3. Note: Using 0 as the SLOT value signifies that this package only has 1 SLOT setting (in other words, this package is not SLOTable).
LICENSE	MUST	This variable specifies what license the program is covered under, i.e. GPL-2, BSD, etc... This field must be set to a valid license (which is any license found in /usr/portage/license/). If the license does not already exist there, it must be added before the ebuild can be added to the portage tree. If the license does not allow redistribution, make sure you place RESTRICT="nomirror" in the ebuild.

KEYWORDS	MUST	This variable now supports a couple of different functions. First of all, this variable specifies what architecture the ebuild is meant for. Some example keywords include: x86, ppc, sparc, mips, alpha, arm, hppa, amd64 and ia64. See the profiles/arch.list file in the Portage tree for more details. Obviously, you would set this to reflect the architecture of the target machine. Portage will not allow an x86 machine to build anything but x86, as specified by the KEYWORDS variable. Packages that do not support the native architecture are automatically masked by Portage. If the KEYWORDS flag has a preceding ~, then that indicates that the particular ebuild works, but needs to be tested in several environments before being moved to the stable profile with the given keyword. If the KEYWORDS flag has a preceding -, then the package does not work with the given keyword. If there is nothing leading KEYWORDS, then the package is considered stable. You can allow installation of these different types of packages through the ACCEPT_KEYWORDS variable in make.conf.
DESCRIPTION	MUST	A short, one line description of your package.
SRC_URI	MUST	The URLs for every source file in your package, separated by whitespace. You should try not to include version numbers in the SRC_URI and S. Always try to use \${PV} or \${P}, and if a version number is not consistent with the name of the source package, make a \${MY_P} variable and use that instead.
HOMEPAGE	MUST	The homepage of the package. If you are unable to locate an official one, try to provide a link from freshmeat.net or a similar package tracking site. Never refer to a variable name in the string; include only raw text.
IUSE	MUST	This is set to whatever USE variables your package utilizes. Remember that KEYWORDS should not be listed in here!
DEPEND	OPT	The package's build dependencies are listed here. See the section Package Dependencies for more details on proper syntax.
RDEPEND	OPT	The package's runtime dependencies are listed here. Once again, see Package Dependencies for more details.

Functions

There are a number of different functions that you can define in ebuild files that control the building and installation process of your package.

Function	Purpose
pkg_setup	Use this function to perform any miscellaneous prerequisite tasks. This might include checking for an existing configuration file.
pkg_nofetch	Inform the user about required actions if for some reason (such as licensing issues) the sources may not be downloaded by Portage automatically. Use

	this in conjunction with <code>RESTRICT="fetch"</code> . You only should display messages in this function, never call <code>die</code> .
<code>src_unpack</code>	Use this function to unpack your sources, apply patches, and run auxiliary programs such as the autotools. By default, this function unpacks the packages listed in <code>A</code> . The initial working directory is defined by <code>WORKDIR</code> .
<code>src_compile</code>	Use this function to configure and build the package. The initial working directory is <code>S</code> .
<code>src_install</code>	Use this function to install the package to an image in <code>D</code> . If your package uses automake, you can do this simply with <code>emake DESTDIR="\${D}" install</code> . Make sure your package installs all its files using <code>D</code> as the root! The initial working directory is <code>S</code> .
<code>src_test</code>	Executed only when <code>FEATURES="test"</code> is set and <code>RESTRICT="test"</code> is unset, the default of this function executes an available testing function from any Makefiles in the <code>\${S}</code> directory, running either <code>"make test"</code> or <code>"make check"</code> depending on what is provided. It can be overridden to create a custom test setup.
<code>pkg_preinst</code>	The commands in this function are run just prior to merging a package image into the file system.
<code>pkg_postinst</code>	The commands in this function are run just following merging a package image into the file system.
<code>pkg_preun</code>	The commands in this function are run just prior to unmerging a package image from the file system.
<code>pkg_postrm</code>	The commands in this function are run just following unmerging a package image from the file system.
<code>pkg_config</code>	You use this function to setup an initial configuration for the package after it's installed. All paths in this function should be prefixed with <code>ROOT</code> which points to user-specified install root which may not happen to be <code>/</code> . This function is only executed if and when the user runs: <code>emerge --config=\${PF}</code> .

Helper Functions

You can also use the following helper functions in your ebuilds.

Function	Purpose
<code>use</code>	Check if one or more given USE-flags are defined. If so, the function will return shell true. In either case, nothing is echoed to standard output. For a verbose version, please use <code>usev</code> which will echo the USE flag if it is defined.
<code>has_version</code>	Returns 1 if the system has the requested version of a certain package. For instance <code>has_version >=sys-libs/glibc-2.3.0</code> .
<code>best_version</code>	Returns category/package-version of the requested category/package. For instance <code>best_version x11-libs/gtk+extra</code> .
<code>use_with</code>	This function checks if a use-flag has been defined and returns <code>"--with-foobar"</code> or <code>"--without-foobar"</code> accordingly. If you only use one argument, that

	argument is both use-flag and with(out)-string. Otherwise the first argument is the use-flag and the second argument the with(out)-string. For instance use_with truetype freetype will echo "--with-freetype" if truetype is in USE.
use_enable	The same as use_with, but returns "--enable-foobar" or "--disable-foobar" accordingly.
check_KV	Checks if Portage knows kernel version. If not, display an error and die. If you need the kernel version in your script, use the KV variable which is automatically defined by Portage. On a system running gentoo-sources-2.4.20-r6, KV would have the value "2.4.20".
keepdir	Creates (if necessary) a .keep file in the given directory so that it isn't auto-cleaned. Never create a .keep file yourself. If portage changes how keepdir works, then creating the file yourself will break the package.
econf	Issues ./configure with the necessary path-changes (prefix, host, mandir, infodir, datadir, sysconfdir, localstatedir). You can optionally pass extra arguments to ./configure by specifying them when you call econf, and users can set the environment variable EXTRA_ECONF if they need to. Options passed to configure take precedence in the reverse order that they were given. In other words, the first argument passed will always be overridden by the last.
einstall	Issues make install with the necessary path-changes (prefix, datadir, mandir, infodir, datadir, sysconfdir, localstatedir). Again, you can pass extra arguments to the make command by specifying them when you call einstall. Please note that the preferred way to install a package is via the emake install DESTDIR="\${D}" command and not via einstall. This command is only a fall back to override broken make files.
die	Causes the current process to be aborted. It will notify the user using the given arguments as a reason. Do not neglect to pass a message to die if you have more than one call to it in a single function. It is harder to track down a failure if you're not sure where the package failed.
elog	Inform the user about something important. The argument given to elog is the message that the user will see. Do not use elog to display banners such as "*****". The fact that you're using elog is enough to get the user's attention. The message is also logged using portages ELOG system.
einfo	Display informative but non-important messages that don't need to be logged.

Helper Functions provided by eutils.eclass

You can use the following helper functions that are provided by the "eutils" eclass in your ebuilds. You must make sure that inherit eutils is present for these functions to work.

Function	Purpose
epatch	This function acts as a friendlier replacement to the patch command and epatch works with .bz2, .gz, .zip and plain text patches. You do not need

	to specify a "-p" option, any options that do need to be explicitly specified should be set in EPATCH_OPTS. The function expects either a file or a directory as an argument - if you specify a directory, all patches in the form of "??_\${ARCH}_" will be applied: for a patch to be applied, it needs to match the running architecture, have "_all_" in the name, or EPATCH_FORCE must be set to "yes".
gen_usr_ldscript	This function generates linker scripts in /usr/lib for dynamic libraries in /lib. This fixes linking problems when a .so is in /lib while a .a is in /usr/lib.
edos2unix	This function performs the same action as the dos2unix binary.
egetent	egetent acts as a wrapper for getent for Linux or nidump for Mac OS X (R).
enewuser	Creates a new user. This function expects a mandatory argument with the username, and several optional arguments can be specified: \$2 contains a UID, pass -1 for the next available ID; \$3 contains the shell, pass -1 for the default; \$4 contains a home directory with /dev/null being the default, \$5 contains any groups to which the user should be added, empty by default and \$6 contains any extra options that you may wish to pass to useradd.
enewgroup	Adds a new group. This function expects a mandatory argument with the group name - an optional second argument makes the group have a specific GID.
make_desktop_entry	Makes a desktop entry: the first argument contains the path to the binary. Optionally, the second contains a name for the icon - the default is \${PN}; the third can contain a path to the icon relative to /usr/share/pixmaps or a full path - the default is \${PN}.png; the fourth can contain an application category , and the fifth argument contains an optional application startup path.
check_license	Displays a license for the user to accept, if no arguments are specified then the license specified by \${LICENSE} is used.
unpack_pdv	Unpacks a pdv generated archive, the first argument must contain the file to unpack and the second should contain "off_t" which has to be manually generated: strace -elseek \${file} and for something like "lseek(3, -4, SEEK_END)" you would pass the value "4".
unpack_makeself	Unpacks a makeself generated archive, requires a file to unpack as the argument.
cdrom_get_cds	Attempts to get a CD, present with files specified by the arguments present on the system and mounted at \${CDROM_ROOT}.
cdrom_load_next_cd	Loads the next CD once you are done with the first CD. If the function returns, \${CDROM_ROOT} would point to the next CD.
strip-linguas	This function makes sure that LINGUAS contains only the languages that a package can support specified by the arguments to the function. If the first argument is -i, then a list of .po files in the specified directories is built and the intersection of the lists is used. If the first argument is -u, then a

	list of .po files in the specified directories is built and the union of the lists is used.

Helper Functions provided by `flag-o-matic.eclass`

You can use the following helper functions that are provided by the "flag-o-matic" eclass in your ebuilds. You must make sure that inherit flag-o-matic is present for these functions to work. You should never modify any compiler settings directly, instead please use flag-o-matic to perform any actions such as filtering flags that cause trouble.

Function	Purpose
filter-flags	This function removes particular flags from C[XX]FLAGS - only complete flags are matched.
append-flags	This function adds extra flags to the existing C[XX]FLAGS variables.
replace-flags	This replaces the flag specified by the first argument with the one in the second argument in the current C[XX]FLAGS.
replace-cpu-flags	Needs two arguments. Replace a given mtune/mcpu/mtune value with the new one (maybe like this: replace-cpu-flags 'i686' 'i586' will replace -mtune/-march/-mcpu=i686 with -mtune/-march/-mcpu=i586).
strip-flags	Strips all flags, except those specified in ALLOWED_FLAGS.
strip-unsupported-flags	Strips C[XX]FLAGS of any flags not supported by the running version of GCC.
get-flag	Finds a flag and outputs its value.
is-flag	This returns true if the flag is set in the current C[XX]FLAGS; only complete matches are performed.
append-ldflags	This function adds extra flags to the existing LDFLAGS variable.
filter-ldflags	Removes the specified flags from LDFLAGS, only complete flags are matched.
fstack-flags	Appends -fno-stack-protector which suppresses -fstack-protector and -fstack-protector-all.

Helper Functions provided by `toolchain-funcs.eclass`

You can use the following helper functions that are provided by the "toolchain-funcs" eclass in your ebuilds. You must make sure that inherit toolchain-funcs is present for these functions to work. You should never explicitly specify any compiler or binutils settings directly, instead please use toolchain-funcs to specify compilers and binutils.

The purpose of using the below functions is to support cross-compiling and the icc compiler. These should be used whenever a package explicitly uses gcc, g++, ld, ranlib or any of the below tools. In general packages that use autoconfiguration tools detect cross compiling automatically and do not need the following functions.

Function	Purpose
tc-getAR	Returns the name of the archiver

tc-getAS	Returns the name of the assembler
tc-getCC	Returns the name of the C compiler
tc-getCXX	Returns the name of the C++ compiler
tc-getLD	Returns the name of the linker
tc-getNM	Returns the name of the symbol/object inspection tool
tc-getRANLIB	Returns the name of the archiver indexer
tc-getF77	Returns the name of the fortran compiler
tc-getGCJ	Returns the name of the java compiler
tc-getBUILD_CC	Returns the name of the C compiler for build
tc-is-cross-compiler	A simple way to see if we're using a cross-compiler
gcc-fullversion	Returns the version as by <code>\$(CC -dumpversion)</code>
gcc-version	Returns the version, but only the <code><major>.<minor></code>
gcc-major-version	Returns the Major version
gcc-minor-version	Returns the Minor version
gcc-micro-version	Returns the Micro version

Rules for writing an ebuild file

Since ebuild files are really just shell scripts, you should use your editor's shell-script mode for editing them. You should use proper indentation, using only tab characters -- no spaces. Make sure you set up your editor to put tab stops at 4 spaces. Always make sure you use braces around your environment variables; e.g. `${P}` instead of just `$P`.

Long lines are wrapped with `'\'`, thus:

Code Listing 2.2: Wrapping lines in ebuilds

```
./configure \
--prefix=/usr || die "configure failed"
```

For further details, refer to `skel.ebuild` (usually residing in `/usr/portage`).

If you use Vim for ebuild/eclass editing, the default Gentoo `vimrc` file, `/etc/vim/vimrc`, already ensures that correct indentation and filetype settings are used for ebuild and eclass files. For better results, including special syntax highlighting for ebuild keywords, emerge `app-vim/gentoo-syntax`.

On non-Gentoo systems, you can obtain similar results by using the following lines in your `vimrc`, or better yet by installing the `gentoo-syntax` scripts which can be downloaded from Gentoo mirrors.

Code Listing 2.3: Configuring vimrc for ebuild-editing

```
au BufRead,BufNewFile *.e{build,class} let is_bash=1|setfiletype sh
au BufRead,BufNewFile *.e{build,class} set ts=4 sw=4 noexpandtab
```

If you're using Emacs, you should emerge `app-emacs/gentoo-syntax` (for GNU Emacs) or `app-xemacs/gentoo-syntax` (for XEmacs). These packages provide Emacs major modes for automatic indentation and syntax highlighting of ebuilds and other Gentoo

specific file types.

If you're using nano, then you're in luck! Just edit `/etc/nanorc` and uncomment the section referring to ebuild's.

USE Variables

The purpose of USE variables is to allow you to configure Portage to globally and automatically enable or disable certain optional build-time features. Here's an example. Let's say you're a GNOME fan, and you'd like any ebuild that has the option of compiling-in optional GNOME support to do so. In this case, you'd add `gnome` to the USE variable in `/etc/make.conf`, and then Portage will automatically add optional GNOME functionality to packages if it is available. Likewise, if you don't want optional GNOME features to be added to your ebuilds if they are available, simply edit `/etc/make.conf` and make sure that `gnome` is not set in the USE variable. Gentoo Linux has an almost overwhelming number of USE options, allowing you to have your system configured exactly the way you want it.

Note: If you unset a USE variable (for example, removing `gnome` from USE), this will only instruct Portage to disable optional build-time support for GNOME. However, if you emerge an ebuild that requires GNOME, the package will obviously have GNOME support enabled, as you would expect. This also means that GNOME will be automatically installed (as a dependency) if it hasn't been already. That's why it's always a good idea to do an `emerge --pretend` before doing the "real" emerge; that way, you'll always know exactly what you're going to get!

In your own ebuilds, you can check whether a USE variable is set by using the `use <variable>` command. You would normally use this command as follows:

Code Listing 2.4: Finding out if a USE-flag is set

```
if use X; then
    # Commands specific to X...
fi
```

USE variables can also be used to set dependencies. For example, you may only want to require a package if a certain USE variable is set. This is done by using the syntax `flag? ( mycat/mypackage )` in the `DEPEND` variable for your ebuild. In this example, `mycat/mypackage` will only be required if `flag` is present in USE. It is also possible to specify what dependency should be used if some USE flag is set, and what dependency to use if it is not set: `flag? ( mycat/mypackage )` and `!flag? ( othercat/otherpackage )`. In this case, if `flag` is not set, `othercat/otherpackage` is used instead of `mycat/mypackage`. Make sure that your ebuilds use this syntax and not Bash IFS. Bash conditionals interfere with Portage's dependency caching, and the use of them will break your ebuild. Here's an important tip about how to use USE. Most of the time, a package will have a `./configure` script used to perform configuration steps. Generally, if your ebuild uses `./configure`, any optional build-time functionality will be enabled or disabled by passing the appropriate arguments to the `./configure` command. Here's the best way to handle this:

Code Listing 2.5: Conditionals based on USE-settings

```
DEPEND="X? ( >=x11-base/xfree-4.3 )
mysql? ( >=dev-db/mysql-3.23.49 )
apache2? ( >=net-www/apache-2 )
!apache2? ( =net-www/apache-1* )"
```

```
src_compile() {
    econf \
        $(use_enable X x11) \
        $(use_enable mysql) \
        || die "Error: econf failed!"
    emake || die "Error: emake failed!"
}
```

This approach has a very nice result. We don't have to worry about what the default setting is for mysql or X (enable/disabled), we explicitly tell econf what we want it to do based upon the USE variable. Not to mention it's quite clean in terms of readability :).

Occasionally, ebuilds will have conflicting optional features. Checking for these conflicts and returning an error is not a viable solution. Instead, you must favor one of the features over the others. As to which, consult upstream (what they use as typical default), or consider which option provides more common functionality, or just flip a coin. One example comes from the msmtplib ebuilds. The package can use either SSL with GnuTLS, SSL with OpenSSL, or no SSL at all. Because GnuTLS has a lot more features compared to OpenSSL, it is favoured:

Code Listing 2.6: Handling conflicting features

```
src_compile() {
    local myconf

    if use gnutls ; then
        myconf="${myconf} --enable-ssl --with-ssl=gnutls"
    elif use ssl ; then
        myconf="${myconf} --enable-ssl --with-ssl=openssl"
    else
        myconf="${myconf} --disable-ssl"
    fi

    econf \
        # Other stuff
        ${myconf} \
        || die "configure failed"

    emake || die "make failed"
}
```

To view a continuously updated table of USE variables, please go [here](#).

1.c. File system Locations

Introduction to the FHS

The file system layout standards used in Gentoo Linux closely follow the FHS, short for File system Hierarchy Standard. A simplified description of the standard is given here; for a complete specification go to <http://www.pathname.com/fhs/>.

Note: The /opt hierarchy is addressed in section 3.12 of the FHS. Section 4.4 deals with the /usr/X11R6 directory. KDE and GNOME are not specifically addressed, and are in fact not even mentioned in the current version of the FHS.

How to fit your packages into the file system

Usually, if the package uses autoconf and automake, the default installation

destinations are mostly correct, with a few exceptions:

- If you're installing a program into `/bin`, `/sbin`, `/usr/bin`, or `/usr/sbin`, then the program's corresponding man page should be installed into the `/usr/share/man` tree. This can often be accomplished by specifying a `./configure --mandir=/usr/share/man` in the ebuild.
- GNU info files should always be installed to `/usr/share/info`, even if the info files are about X11, GNOME or KDE-specific programs or tools. Make a note: `/usr/share/info` is the only official location for GNU info files. Since many `./configure` scripts default to installing GNU info files in `/usr/info`, it's often necessary to call `./configure` with the `--infodir=/usr/share/info` argument.
- Documentation files are installed in `/usr/share/doc`, into a subdirectory reflecting the name, version, and revision of the particular program. This applies to all programs: GNOME, KDE, X11 and console alike. However, some programs may install additional documentation and support files into a `/usr/share` hierarchy for their own purposes.
- X11-specific programs and libraries should always be installed into `/usr`, not directly into `/usr/X11R6`. We reserve the `/usr/X11R6` hierarchy for the X Window System, Version 11 Release 6 itself. This is perhaps a more to-the-letter interpretation of the FHS than some other distributions have made.
- GNOME and KDE programs, similarly, should always be installed into `/usr`.

Important: Some distributions choose to install GNOME and KDE into `/opt`. There exists no standard for these desktop environments in terms of where to actually install their files. In the interests of simplicity and consistency, we elect to install all KDE and GNOME packages into the `/usr` hierarchy.

In general, you should have ebuilds install their files into the `/usr` tree. Some programs can be compiled and linked with or without GNOME, KDE, and X11 libraries, which can cause confusion. Our solution is to install everything into `/usr` which avoids ambiguity and needless complexity for ebuild authors. The location in which to install a program's files should not depend on the presence or absence of specific USE variables. Therefore, the ebuilds in the portage tree almost always install into the `/usr` hierarchy exclusively.

Note: The `/opt` directory is reserved in Gentoo Linux for binary-only packages. Examples include `mozilla-bin`, `acoread`, `netscape` and `realplayer`. Packages that get installed here will usually require a `/etc/env.d/foo` stub file. This is so that paths and additional variables can be included into the environment. For more information on `/etc/env.d`, please visit [this](#) document.

1.d. The Portage scripts and utilities

Public scripts

These are scripts used by the system-administrator to install and remove packages, and maintain the package database.

ebuild is the main engine of the Portage system; it performs all major tasks such as unpacking, compiling, installing, merging, and unmerging packages. It is called using the command: `ebuild path/to/package.ebuild` command. The commands available are:

Command	Description	Related e b u i l d Function

setup*	Performs any miscellaneous commands required before the ebuild can proceed	pkg_setup
depend	Displays the dependencies required to build the package	N/A
merge*	Unpacks, compiles, installs, and merges the package into your file system	N/A
qmerge*	Merges the package into your file system, assuming that the unpack, compile, and install stages have already been executed	N/A
unpack*	Unpacks the source tarballs into the work directory	src_unpack
compile*	Compiles the package	src_compile
rpm	Creates an RPM from the package	N/A
package	Creates a Gentoo tbz2 package	N/A
prerm*	Executes the pre-removal stage of the package	pkg_prerm
postrm*	Executes the post-removal stage of the package	pkg_postrm
preinst*	Executes the pre-installation stage of the package	pkg_preinst
postinst*	Executes the post-installation stage of the package	pkg_postinst
config	Sets up a default configuration once the package is merged	pkg_config
touch*	Updates the mtimes for each source archive in the package	N/A
clean*	Cleans the work directory for the package	N/A
fetch*	Fetches the package source tarballs	N/A
digest*	Creates a digest file for the package	N/A
test*	Runs the self-test routine for the package	src_test
install*	Installs the package into the image directory	src_install
unmerge	Unmerges the package from your file system	N/A

Note: Commands with an asterisk (*) are normally only used by the developer.

emerge recursively merges a package and all of its dependencies into your file system. This command has many options, try `emerge --help` for a list of them.

env-update updates the configuration files (including, but not limited to `/etc/ld.so.conf` and `/etc/profile.env`) to include changes made by installed packages.

Private Scripts and Commands

These are scripts you can use in your ebuild files to perform common tasks.

For you down and dirty people, look at the scripts themselves in `/usr/lib/portage/bin`.

Command	Default Value	Description	Example
diropts	-m0755	Sets the options used when running <code>doDIR</code>	<code>diropts -m0750</code>
dobin	N/A	Installs the specified binaries into <code>DESTTREE/bin</code>	<code>dobin wmacpi</code>

docinto	""	Sets the relative subdir used by dodoc	docinto examples
dodir	N/A	Creates a directory, handling \${D} transparently	dodir /usr/lib/newpacka ge
dodoc	N/A	Installs the specified files into the package's documentation directory (/usr/share/doc/\${PF}/DOCDESTTREE) (see docinto)	dodoc README *.txt
doexe	N/A	Installs the specified files with mode EXEPTIONS (see exeopts) into PATH defined by EXEINTO (see exeinto).	doexe \$ {FILESDIR}/quak e3
dohard	N/A	Creates a hard link, handling \${D} transparently	dohard ls /bin/dir
dohtml	N/A	Installs the specified files and directories into /usr/share/doc/\${PF}/html	dohtml -r doc/html/*
doinfo	N/A	Installs the specified files into /usr/share/info, then compresses them with gzip	doinfo doc/*.info
doins	N/A	Installs the specified files with mode INSOPTIONS (see insopts) into INSDESTTREE (see insinto)	doins *.png icon.xpm
dolib	N/A	Installs the specified libraries into DESTTREE/lib with mode 0644	dolib *.a *.so
dolib.a	N/A	Installs the specified libraries into DESTTREE/lib with mode 0644	dolib.a *.a
dolib.so	N/A	Installs the specified libraries into DESTTREE/lib with mode 0755	dolib.so *.so
doman	N/A	Installs the specified files into /usr/share/man/manX, according to the suffix of the file (file.1 will go into man1)	doman *.1 *.5
dosbin	N/A	Installs the files into DESTTREE/sbin, making sure they are executable	dosbin ksymoops
dosym	N/A	Creates a symlink, handles \${D} transparently	dosym gzip /bin/zcat
emake	N/A	Runs make with MAKEOPTS. Some packages cannot be made in parallel; use emake -j1 instead. If you need to pass any extra arguments to make, simply append them onto the emake command. Users can set the EXTRA_EMAKE environment variable to pass extra flags to emake.	emake
exeinto	/	Sets the root (EXEDESTTREE) for the doexe command	exeinto /usr/lib/\$ {PN}

exeopts	-m0755	Sets the options used when running doexe	exeopts -m1770
fowners	N/A	Applies the specified ownership to the specified file via the chown command, handles \${D} transparently	fowners root:root /sbin/functions.sh
fperms	N/A	Applies the specified permissions to the specified file via the chmod command, handles \${D} transparently	fperms 700 /var/consoles
insinto	/usr	Sets the root (INSDSTTREE) for the doins command	insinto /usr/include
insopts	-m0644	Sets the options used when running doins	insopts -m0444
into	/usr	Sets the target prefix (DESTTREE) for all the 'do' commands (like dobin, dolib, dolib.a, dolib.so, domo, dosbin)	into /
libopts	-m0644	Sets the options used when running dolib	libopts -m0555
newbin	N/A	Wrapper around dobin which installs the specified binary transparently renaming to the second argument	newbin \$ {FILESDIR}/vmware.sh vmware
newdoc	N/A	Wrapper around dodoc which installs the specified file transparently renaming to the second argument	newdoc README README.opengl
newexe	N/A	Wrapper around doexe which installs the specified file transparently renaming to the second argument	newexe \$ {FILESDIR}/xinetd.rc xinetd
newins	N/A	Wrapper around doins which installs the specified file transparently renaming to the second argument	newins ntp.conf.example ntp.conf
newman	N/A	Wrapper around doman which installs the specified file transparently renaming to the second argument	newman xboing.man xboing.6
newsbin	N/A	Wrapper around dosbin which installs the specified file transparently renaming to the second argument	newsbin strings strings-static
prepall	N/A	Runs prepallman, prepallinfo and prepallstrip. Also ensures all libraries in /opt/*/lib, /lib, /usr/lib and /usr/X11R6/lib are executable. also moves any stray aclocal macros into /usr/share/aclocal	prepall
prepalldocs	N/A	Recursively gzips all doc files in /usr/share/doc, transparently fixing up any symlink paths	prepalldocs
prepallinfo	N/A	Recursively gzips all info files in /usr/share/info	prepallinfo

prepallman	N/A	Recursively gzips all man pages in /opt/*/man/*, /usr/share/man/*, /usr/local/man/*, /usr/X11R6/share/man/* and transparently fixes up any symlink paths	prepallman

1.e. Package Dependencies

Why dependencies are important

Portage is more than just a convenience script that gives you a unified way to build any one project (program, library) from source. It will also fetch and install any necessary dependencies if you take care to specify these in your ebuild.

In the official ebuilds, all dependencies have already been specified, so when you issue `emerge net-www/mozilla/mozilla-1.0`, Portage will insure that all libraries necessary for Mozilla to build and run are properly installed before Mozilla itself is built.

Portage even distinguishes between build-time dependencies and run-time dependencies. (Caveat: Currently, Portage installs all build-time and run-time dependencies and leaves it at that. At a later stage, it will be possible to trim your installation so that only the run-time dependencies are left installed).

How to Specify Dependencies in Your ebuild Files (a.k.a. `DEPEND` Atoms)

The `DEPEND` variable inside your `foo-x.y.z.ebuild` tells Portage about which packages are needed to build `foo`. The `RDEPEND` variable specifies which packages are needed for `foo` to run. `RDEPEND` should be set explicitly even if it's the same as `DEPEND` because in the future it defaulting to `DEPEND` is planned to be removed from Portage.

Code Listing 5.1: Depend example

```
DEPEND="virtual/opengl
        dev-libs/libxml2"
RDEPEND="${DEPEND}"
```

This tells Portage that to build `foo-x.y.z`, the packages `virtual/opengl` (more on `virtuals` in a bit) and `dev-lib/libxml2` are needed. It does not say anything about which version of `opengl` or `libxml2` that are needed, which means "anything goes".

The "anything goes" is of course a bit scary, and will not work in the general case. But for libraries, which strive very hard to be 100% binary compatible all the time, it actually works. For other libraries, we can of course specify version dependencies.

Code Listing 5.2: Version example

```
>=sys-apps/bar-1.2
=sys-apps/baz-1.0
```

`>=` and `=` do what you would expect; `sys-apps/bar` version 1.2 or newer is okay (this means that `sys-apps/bar-2.0` is okay), while `sys-apps/baz` version 1.0 is the only version that is accepted. For more information on the version schema of packages, see the section above on [Naming ebuild Files](#).

Other methods of specifying version dependencies are as follows:

Code Listing 5.3: Specifying version dependencies

```
~sys-apps/qux-1.0
=sys-apps/foo-1.2*
```

```
!sys-libs/gdbm
```

~sys-apps/qux-1.0 will select the newest portage revision of qux-1.0.

=sys-apps/foo-1.2* will select the newest member of the 1.2 series, but will ignore 1.3 and later/earlier series. That is, foo-1.2.3 and foo-1.2.0 are both valid, while foo-1.3.3, foo-1.3.0, and foo-1.1.0 are not.

!sys-libs/gdbm will prevent this package from being emerged while gdbm is already emerged.

Important Notes

There are many things that go wrong with the DEPEND and RDEPEND variables. Here are some important points to follow when you write the dependencies.

- Always include the CATEGORY.
For example, use `>=x11-libs/gtk+-2` and not `>=gtk+-2`.
- Do not put an asterisk (*) for `>=` dependencies.
For example, it should be `>=x11-libs/gtk+-2` rather than `>=x11-libs/gtk+-2*`.
- Never depend on a meta-package.
So don't depend on `gnome-base/gnome`, always depend on the specific libraries like `libgnome`.
- One dependency per line.
Don't put multiple dependencies on the same line. It makes it ugly to read and hard to follow.
- GTK: Always use `=x11-libs/gtk+-1.2*` for GTK+1 apps.

Additionally, it is important to ensure that all the dependencies are complete for your package:

- Look in `configure.in` or `configure.ac`
Look for checks for packages in here. Things to look out for are `pkg-config` checks or `AM_*` functions that check for a specific version.
- Look at included `.spec` files
A good indication of dependencies is to look at the included `.spec` files for relevant deps. However, do not trust them to be the definitive complete list of dependencies.
- Look at the application/library website
Check the application website for possible dependencies that they suggest are needed.
- Read the README and INSTALL for the package
They usually also contain useful information about building and installing packages.
- Remember non-binary dependencies such as `pkg-config`, `doc generation programs`, etc.
Usually the build process requires some dependencies such as `intltool`, `libtool`, `pkg-config`, `doxygen`, `scrollkeeper`, `gtk-doc`, etc. Make sure those are clearly stated.

For all the latest details about these DEPEND Atoms, please see the section 5 manpage on ebuids: `man 5 ebuild`.

1.f. Testing and deploying

ChangeLog

Whenever you update (or write a new) an ebuild, you must also update its (or create a new) ChangeLog. The `skel.ChangeLog` contains a sample ChangeLog that you can use as a basis.

The purpose of the ChangeLog is to document what is being done, why it is being done, and by whom. This allows both developers and users to trace the changes made in an easy way.

The ChangeLog is primarily targeted at users, so be sure to keep your writing short, to the point, and avoid getting verbose about the internal technical details.

Storing your own ebuilds locally

In order to be able to test your ebuilds and let Portage know about them, you must place those in a known directory. Portage will use the `PORTDIR_OVERLAY` variable which you can define in `/etc/make.conf`. You should set this variable to your directory (e.g. `/usr/local/portage`).

In that directory, you must use the same structure (and categories) as in `/usr/portage`.

Using this `PORTDIR_OVERLAY`, your ebuilds remain on your system, even after an `emerge sync`, and they are still known to Portage.

Testing the package

Have a think about how you will test whether this package works. Sometimes the developers have already included a `make test` or `make check` routine that will test the basic functionality of the package. If so, then running `env FEATURES=test ebuild foo-x.y.z.ebuild test` will execute it. If it is broken try to fix it so that it works (and submit the patch to the upstream developers).

If this is not the case consider adding a `src_test` routine to your ebuild. This is executed before the `src_install` routine and can be very helpful for testing the program works across various architectures. The architecture developers will appreciate if you add a routine here so that they do not require knowledge of the package's functionality.

Please keep in mind the general requirements of an ebuild here. The `src_test` routine must not be interactive. If the test routine depends on other packages use the `test USE` flag to specify the optional compile time `DEPENDencies`. Also, please note that `src_test` routines are not recommended for graphical X applications as the user running portage often cannot run them successfully.

Useful testing tools

We have a few useful tools to help you with writing and maintaining your ebuilds.

Tool	Package	Description
repoman	sys-apps/portage	Developer-only tool to assist with the CVS check in procedure. It does a lot of common QA and tries to make sure that files added to cvs will not break the portage tree.
ccache	dev-util/ccache	Tool that keeps pre-processed files so that recompilation gets done much faster. Be sure to add <code>ccache</code> to the <code>FEATURES</code> variable in <code>/etc/make.conf</code> !
sandboxshell	app-shells/sandboxshell	Launch a shell that creates a sandbox environment. Useful for entering the same environment that portage builds packages inside of and debugging

		things by hand.
echangelog	app- portage/gentoolkit- dev	Can create a new ChangeLog or add an entry to an existing one.

Использование rpm-пакетов в Gentoo

Ссылка на оригинал: <http://ylsoftware.com/?action=news&na=viewfull&news=402>

Автор: [Калинников В.В. а.к.а. MooSe](#)

С версии: 1.5.

Дата оригинала: 15.02.2008

Эта статья написана для [сборника статей о Gentoo](#) по просьбе одного из его авторов - Евгения Батогова а.к.а JohnBat26. Здесь я постараюсь описать возможные способы установки rpm-пакетов в Gentoo. Сразу хочу отметить, что в любом случае лучше всего ставить программы из исходных текстов, используя команду `emerge`. Использовать же rpm нужно только в самом крайнем случае, когда архив с исходными текстами не доступен (например для коммерческих программ).

Итак. Есть два способа установить rpm-пакет:

1. Установить в систему менеджер пакетов rpm и пользоваться им.
2. Написать `ebuild` для распаковки и установки rpm-пакета.

Первый способ проще, но пакет придётся устанавливать с ключом `--nodeps` и есть большой риск сломать систему или просто получить не рабочий пакет. Второй способ сложнее. Он тоже не решает всех проблем первого, но по крайней даёт возможность избежать поломки системы.

Будем полагать что мы хотим установить в систему пакет `prog-0.1.i386.rpm`. Независимо от того, какой способ выбран понадобится утилита rpm:

```
emerge rpm
```

Теперь рассмотрим первый способ установки. Для начала полезно просмотреть зависимости пакета:

```
rpm -qp --qf "[%{REQUIRENAME} %{REQUIREFLAGS:depflags} %{REQUIREVERSION}\n]"  
prog-0.1.i386.rpm
```

Получаем примерно вот такой список:

```
/bin/bash  
/bin/sh  
/bin/sh  
/bin/sh  
/bin/sh  
libX11.so.6  
libXext.so.6  
libXt.so.6  
libc.so.6  
libc.so.6(GCC_3.0)  
libc.so.6(GLIBC_2.0)  
libc.so.6(GLIBC_2.1)  
libc.so.6(GLIBC_2.1.2)  
libc.so.6(GLIBC_2.1.3)  
libc.so.6(GLIBC_2.2)
```



```

libc.so.6(GLIBC_2.3)
libdl.so.2
libdl.so.2(GLIBC_2.0)
libdl.so.2(GLIBC_2.1)
libfontconfig.so.1
libfreetype.so.6
libglib-2.0.so.0
libgobject-2.0.so.0
libgtk-x11-2.0.so.0
libm.so.6
libm.so.6(GLIBC_2.0)
libpthread.so.0
libpthread.so.0(GLIBC_2.0)
libpthread.so.0(GLIBC_2.1)
libpthread.so.0(GLIBC_2.2)
libpthread.so.0(GLIBC_2.2.3)
libpthread.so.0(GLIBC_2.3.2)
rpmllib(CompressedFileNames) <= 3.0.4-1
rpmllib(PayloadFilesHavePrefix) <= 4.0-1

```

Убеждаемся что всё это у нас есть. Если чего-то не хватает - прикидываем в каком пакете оно может быть и ставим этот пакет. Наконец, когда всё что нужно у нас установлено, устанавливаем сам rpm-пакет:

```
rpm -Uvh --nodeps prog-0.1.i386.rpm
```

Всё. Пакет установлен. Однако всё-таки лучше было бы написать для него ebuild. Для работы с rpm-пакетами в portage есть класс rpm. Самый простой способ ознакомиться с ним это прочитать его исходный код, которые расположен в файле `/usr/portage/eclass/rpm.eclass`. Приведу простой пример ebuild'a для установки rpm:

```

# Подключаем класс rpm
inherit rpm

# Описываем программу
DESCRIPTION="Cool prog"
HOMEPAGE="http://example.com"
SLOT="0"
KEYWORDS="~x86"
IUSE=""
RESTRICT="nomirror"

# Описываем зависимости
DEPENDS="x11-libs/qt"

# Указываем откуда брать пакет
SRC_URI="http://example.com/files/prog-0.1.i386.rpm"

# Дальше всё будет сделано само. Кроме собственно установки.

# Устанавливаем пакет:
src_install() {
    cp -vR ${S}/* ${D}/
}

```

Дальше нужно сохранить этот ebuild в нужном месте (обратитесь к документации portage) и выполнить:

```
ebuild /path/to/file.ebuild digest  
ebuild /path/to/file.ebuild merge
```

Всё. Пакет установлен. Причём штатными средствами Gentoo (что немало важно). Огромное спасибо пользователю [Alex](#) за помощь в написании статьи.

Изучаем параметры gcc

Ссылка на оригинал: [Getting Familiar with GCC Parameters](#)

Ссылка на перевод: <http://netsago.org/ru/docs/1/9/>

С версии: 1.5

Перевод: n0xi0uzz

Перевод статьи «[Getting Familiar with GCC Parameters](#)», автор — Mulyadi Santosa

gcc (GNU C Compiler) — набор утилит для компиляции, ассемблирования и компоновки. Их целью является создание готового к запуску исполняемого файла в формате, понимаемом вашей ОС. Для Linux, этим форматом является ELF (Executable and Linking Format) на x86 (32- и 64-битных). Но знаете ли вы, что могут сделать для вас некоторые параметры gcc? Если вы ищете способы оптимизации получаемого бинарного файла, подготовки сессии отладки или просто наблюдать за действиями, предпринимаемыми gcc для превращения вашего исходного кода в исполняемый файл, то знакомство с этими параметрами обязательно. Так что, читайте.

Напомню, что gcc делает несколько шагов, а не только один. Вот небольшое объяснение их смысла:

- **Препроцессирование:** Создание кода, более не содержащего директив. Вещи вроде «#if» не могут быть поняты компилятором, поэтому должны быть переведены в реальный код. Также на этой стадии разворачиваются макросы, делая итоговый код больше, чем оригинальный. [1]
- **Компиляция:** Берется обработанный код, проводятся лексический и синтаксический анализы, и генерируется ассемблерный код. В течение этой фазы, gcc выдает сообщения об ошибках или предупреждениях в случае, если анализатор при парсинге вашего кода находит там какие-либо ошибки. Если запрашивается оптимизация, gcc продолжит анализировать ваш код в поисках улучшений и манипулировать с ними дальнейшем. Эта работа происходит в многопроходном стиле, что показывает то, что иногда требуется больше одного прохода по коду для оптимизации. [2]
- **Ассемблирование:** Принимаются ассемблерные мнемоники и производятся объектные коды, содержащие коды команд. Часто недопонимают то, что на стадии компиляции не производятся коды команд, это делается на стадии ассемблирования. В результате получаются один или более объектных файла, содержащие коды команд, которые являются действительно машинозависимыми. [3]
- **Компоновка:** Трансформирует объектные файлы в итоговые исполняемые. Одних только кодов операции недостаточно для того, чтобы операционная система распознала и выполнила их. Они должны быть встроены в более полную форму. Эта форма, известная как бинарный формат, указывает, как ОС загружает бинарный файл, компоует перемещение и делает другую необходимую работу. ELF является форматом по умолчанию для Linux на x86. [4]

Параметры gcc описаны здесь, прямо и косвенно затрагивая все четыре стадии, поэтому для ясности, эта статья построена следующим образом:

- Параметры, относящиеся к оптимизации
- Параметры, относящиеся к вызову функций
- Параметры, относящиеся к отладке
- Параметры, относящиеся к препроцессированию

Прежде всего, давайте ознакомимся с вспомогательными инструментами, которые помогут нам проникать в итоговый код:

- Коллекция утилит ELF, которая включает в себя такие программы, как `objdump` и `readelf`. Они парсят для нас информацию о ELF.
- [Oprofile](#), один из стандартных путей подсчета производительности аппаратного обеспечения. Нам нужна эта утилита для просмотра нескольких аспектов производительности кода.
- `time`, простой способ узнать общее время работы программы.

Следующие инструкции могут быть применены в gcc версий 3.x и 4.x, так что они достаточно общие. Начнем копать?

Параметры, относящиеся к оптимизации кода

gcc предоставляет очень простой способ производить оптимизацию:

опция `-O`. Она и ускоряет выполнение вашего кода, и сжимает размер итогового кода. У неё существует пять вариантов:

- от `-O0` (O ноль) до `-O3`. "0" означает отсутствие оптимизации, а "3" — наивысший уровень оптимизации. "1" и "2" находятся между этими краями. Если просто используете `-O` без указания номера, это будет означать `-O1`.
- `-Os` говорит gcc оптимизировать размер. В общем-то, это похоже на `-O2`, но пропускает несколько шагов, которые могут увеличить размер.

Какое ускорение в действительности можно от них получить? Что ж, предположим, у нас есть такой код:

```
#include

int main(int argc, char *argv[])
{
    int i,j,k;

    unsigned long acc=0;

    for(i=0;i<10000;i++)
        for(j=0;j<5000;j++)
            for(k=0;k<4;k++)
                acc+=k;

    printf("acc = %lu\n",acc);

    return 0;
```

```
}
```

С помощью gcc, создадутся четыре разных бинарных файла, используя каждый из -O вариантов (кроме -Os). Утилита time запишет их время исполнения, таким образом:

```
$ time ./non-optimized
```

	Без оптимизации	-O1	-O2	-O3
real	0.728	0.1	0.1	0.1
user	0.728	0.097	0.1	0.1
sys	0.000	0.002	0.000	0.000

Для упрощения, будем использовать следующие обозначения:

- *Non-optimized* обозначает исполняемый файл, скомпилированный с опцией -O0.
- *OptimizedO1* обозначает исполняемый файл, скомпилированный с опцией -O1.
- *OptimizedO2* обозначает исполняемый файл, скомпилированный с опцией -O2.
- *OptimizedO3* обозначает исполняемый файл, скомпилированный с опцией -O3.

Как вы могли заметить, время выполнения программы, скомпилированной с -O1 в семь раз меньше, чем время выполнения программы, при компиляции которой не использовалась оптимизация. Обратите внимание, что нет большой разницы между -O1, -O2 и -O3, — на самом деле, они почти одинаковы. Так в чем же магия -O1?

После беглого изучения исходного кода, вы должны отметить, что такой код конечен для оптимизации. Прежде всего, давайте посмотрим на короткое сравнение дизассемблированных версий non-optimized и optimizedO1:

```
$ objdump -D non-optimized
```

```
$ objdump -D optimizedO1
```

(Примечание: вы можете получить другие результаты, поэтому используйте эти как основные)

Non-optimized	OptimizedO1
mov 0xffffffff4(%ebp)	add \$0x6,%edx
add %eax,0xffffffff8(%ebp)	add \$0x1,%eax
addl \$0x1,0xffffffff4(%ebp)	cmp \$0x1388,%eax
cmpl \$0x3,0xffffffff4(%ebp)	

Приведенные примеры реализуют самый вложенный цикл (for (k=0;k<4;k++)). Обратите внимание на различие: неоптимизированный код напрямую загружает и хранит из адреса памяти, в то время как optimizedO1 использует регистры ЦПУ в качестве сумматора и счетчик цикла. Как вам, возможно, известно, доступ к регистрам может быть получен в сотни или тысячи раз быстрее, чем к ячейкам ОЗУ.

Не удовлетворяясь простым использованием регистров ЦПУ, gcc использует другой трюк оптимизации. Давайте снова посмотрим дизассемблированный код `optimizedO1` и обратим внимание на функцию `main()`:

.....

08048390 :

...

```
80483a1:      b9 00 00 00 00      mov     $0x0,%ecx
80483a6:      eb 1f              jmp     80483c7
80483a8:      81 c1 30 75 00 00    add     $0x7530,%ecx
```

0x7530 это 30000 в десятичной форме, поэтому мы можем быстро угадать цикл. Этот код представляет собой самый вложенный и самый внешний циклы (`for(j=0;j<5000;j++) ... for(k=0;k<4;k++)`), так как они являются буквальным запросом на 30000 проходов. Примите во внимание, что вам нужно всего лишь три прохода внутри. Когда `k=0`, асс остается прежним, поэтому первый проход можно проигнорировать.

```
80483ae:      81 f9 00 a3 e1 11    cmp     $0x11e1a300,%ecx
80483b4:      74 1a                je      80483d0
80483b6:      eb 0f              jmp     80483c7
```

Хмм, теперь это соответствует 300 000 000 (10 000*5 000*6). Представлены все три цикла. После достижения этого числа проходов, мы переходим прямо к `printf()` для вывода суммы (адреса 0x80483d0 - 0x80483db).

```
80483b8:      83 c2 06              add     $0x6,%edx
80483bb:      83 c0 01              add     $0x1,%eax
80483be:      3d 88 13 00 00        cmp     $0x1388,%eax
80483c3:      74 e3                je      80483a8
80483c5:      eb f1              jmp     80483b8
```

Шесть добавляется в сумматор при каждой итерации. В итоге, `%edx` будет содержать всю сумму после выполнения всех трех циклов. Третья и четвертая строки показывают нам, что после выполнения 5000 раз, должен быть переход к адресу 0x80483a8 (как указано ранее).

Мы можем заключить, что gcc создает здесь упрощение. Вместо прохода три раза в самый вложенный цикл, он просто добавляет шесть для каждого среднего цикла. Это звучит просто, но это заставляет вашу программу сделать только 100 000 000 проходов вместо 300 000 000. Это упрощение, называемое

разворачиванием цикла, одно из тех задач, которые делают -O1/2/3. Конечно же, вы и сами можете это сделать, но иногда неплохо знать, что gcc может определить такие вещи и оптимизировать их.

С опциями -O2 и -O3 gcc тоже пытается произвести оптимизацию. Обычно она достигается посредством переупорядочивания [5] и трансформацией кода. Целью этой процедуры является устранить столько ошибочных ветвей, сколько возможно, что повышает качество использования конвейера. Например, мы можем сравнить, как non-optimized и optimizedO2 выполняет самый внешний цикл.

```
80483d4:      83 45 ec 01          addl    $0x1,0xffffffffec(%ebp)

80483d8:      81 7d ec 0f 27 00 00   cmpl    $0x270f,0xffffffffec(%ebp)

80483df:      7e c4                jle     80483a5
```

Бинарный файл non-optimized использует jle для выполнения перехода. Математически это означает, что вероятность выбора ветви 50%. С другой стороны, версия optimizedO2 использует следующее:

```
80483b4:      81 c1 30 75 00 00      add     $0x7530,%ecx

80483ba:      81 f9 00 a3 e1 11      cmp     $0x11e1a300,%ecx

80483c0:      75 e1                jne     80483a3
```

Теперь, вместо jle используется jne. При условии, что любое целое может быть сопоставлено в предыдущем cmp, нетрудно сделать вывод, что это увеличит шанс выбора ветви почти до 100%. Это небольшое, но полезное указание процессору для определения того, какой код должен быть выполнен. Хотя, для большинства современных процессоров, этот вид трансформации не является ужасно необходимым, так как предсказатель переходов достаточно умен для того, чтобы сделать это сам.

Для доказательства того, как сильно это изменение может помочь, к нам на помощь придет OProfile. Oprofile выполнен для записи числа изолированных ветвей и изолированных ошибочных ветвей. *Изолированные* здесь обозначает «выполненные внутри конвейера данных ЦПУ»

```
$ opcontrol --event=RETIRED_BRANCHES_MISPREDICTED:1000
--event=RETIRED_BRANCHES:1000;
```

Мы запустим non-optimized и optimizedO2 пять раз каждый. Затем мы возьмем максимум и минимум примеров. Мы посчитаем степень ошибки, используя эту формулу (выведена [отсюда](#)).

Степень ошибки = изолированные ошибочные ветви / изолированные ветви

Теперь вычислим степень ошибки для каждого бинарного файла. Для non-optimized получилось 0,5117%, в то время как optimizedO2 получил 0,4323% — в нашем случае, выгода очень мала. Фактическая выгода может различаться для реальных случаев, так как gcc сам по себе не может много сделать без внешних указаний. Пожалуйста, прочтите о __builtin_expect() в [документации по gcc](#) для подробной информации.

Параметры, относящиеся к вызову функций

По существу, gcc предоставляет вам несколько путей управления тем, как вызывается функция. Сначала давайте рассмотрим встраивание. С помощью встраивания, вы сокращаете стоимость вызова функции, так как тело функции подставлено прямо в вызывающую функцию. Пожалуйста, учтите, что это не по умолчанию, а только когда вы используете -O3 или, по крайней мере, -finline-functions.

Как полученный бинарный файл выглядит после того, как gcc сделает встраивание? Рассмотрим следующий листинг:

```
#include

inline test(int a, int b, int c)
{
    int d;

    d=a*b*c;

    printf("%d * %d * %d is %d\n",a,b,c,d);
}

static inline test2(int a, int b, int c)
{
    int d;

    d=a+b+c;

    printf("%d + %d + %d is %d\n",a,b,c,d);
}

int main(int argc, char *argv[])
{
    test(1,2,3);

    test2(4,5,6);
}
```

Скомпилируем этот код со следующим параметром:

```
$ gcc -S -O3 -o
```

-S указывает gcc остановиться сразу после стадии компиляции (мы расскажем о ней позже в этой статье). Результат будет следующим:

....


```

test:

    pushl    %ebp

    movl     %esp, %ebp

    pushl    %ebx

    ....

main:

    leal     4(%esp), %ecx

    andl     $-16, %esp

    pushl    -4(%ecx)

    ...

    movl     $6, 16(%esp)

    movl     $3, 12(%esp)

    movl     $2, 8(%esp)

    movl     $1, 4(%esp)

    movl     $.LC0, (%esp)

    call     printf

    ...

    movl     $15, 16(%esp)

    movl     $6, 12(%esp)

    movl     $5, 8(%esp)

    movl     $4, 4(%esp)

    movl     $.LC1, (%esp)

    call     printf

    ...

```

И `test()`, и `test()` действительно встроены, но вы также можете видеть `test()`, который остался вне `main()`. Вот где играет роль ключевое слово `static`. Написав, что функция — `static`, вы сообщаете gcc, что эта функция не будет вызываться из какого-либо внешнего объектного файла, поэтому нет нужды порождать коды. Таким образом, это экономит размер, и если вы можете сделать функцию статичной, сделайте это где только возможно. С другой стороны, будьте благоразумны при решении, какая функция должна быть встраиваемой.

Увеличение размера для небольшого увеличения скорости не всегда оправдано.

С помощью некоторой эвристики, gcc решает, должна быть функция встраиваемой, или нет. Одним из таких доводов является размер функции в терминах псевдо-инструкций. По умолчанию, лимитом является 600. Вы можете поменять этот лимит, используя `-finline-limit`. Протестируйте для нахождения лучших лимитов встраивания для вашего конкретного случая. Также возможно переделать эвристику так, чтобы gcc всегда встраивал функцию. Просто объявите вашу функцию так:

```
__attribute__((always_inline)) static inline test(int a, int b, int c)
```

Теперь перейдем к передаче параметров. На архитектуре x86, параметры помещаются в стек и позже достаются из стека для дальнейшей обработки. Но gcc дает вам возможность изменить это поведение и использовать вместо этого регистры. Функции, у которых меньше трех параметров могут использовать эту возможность указанием `-mregparm=`

...

test:

```
    pushl    %ebp
    movl     %esp, %ebp
    subl     $56, %esp
    movl     %eax, -20(%ebp)
    movl     %edx, -24(%ebp)
    movl     %ecx, -28(%ebp)
```

...

main:

...

```
    movl     $3, %ecx
    movl     $2, %edx
    movl     $1, %eax
    call     test
```

Вместо стека, используются EAX, EDX и ECX для хранения первого, второго и третьего параметров. Поскольку доступ к регистру происходит быстрее, чем к ОЗУ, это будет одним из способов уменьшить время работы. Хотя вы должны обратить внимание на следующие вещи: — Вы ДОЛЖНЫ компилировать весь ваш код с таким же числом `-mregparm` регистров. Иначе у вас будут проблемы с вызовом функций из другого объектного

файла, если они будут принимать разные соглашения. — Используя `-mregparm`, вы разрушаете совместимый с Intel x86 бинарный интерфейс приложений (ABI). Поэтому, вы должны учитывать это, если вы распространяете свое ПО только в бинарной форме.

Возможно, вы заметили эту последовательность в начале каждой функции:

```
push    %ebp

mov     %esp,%ebp

sub     $0x28,%esp
```

Эта последовательность, также известная как *пролог функции*, написана чтобы установить указатель фрейма (EBP). Это приносит пользу, помогая отладчику делать трассировку стека. Следующая структура поможет вам понять это [6]:

[ebp-01] Последний байт последней локальной переменной

[ebp+00] Старое значение ebp

[ebp+04] Возвращает адрес

[ebp+08] Первый аргумент

Можем мы пренебречь этим? Да, с помощью `-fomit-frame-pointer`, пролог будет укорочен, так что функция начнется просто с выделения стека (если есть локальные переменные):

```
sub     $0x28,%esp
```

Если функция вызывается очень часто, вырезание пролога спасет несколько тактов ЦПУ. Но будьте осторожны: делая это, вы также усложняете отладчику задачу по изучению стека. Например, давайте добавим `test(7,7,7)` в конец `test2()` и перекомпилируем с параметром `-fomit-frame-pointer` и без оптимизации. Теперь запустите `gdb` для исследования бинарного файла:

```
$ gdb inline

(gdb) break test

(gdb) r

Breakpoint 1, 0x08048384 in test ()

(gdb) cont

Breakpoint 1, 0x08048384 in test ()

(gdb) bt
#0  0x08048384 in test ()
#1  0x08048424 in test2 ()
#2  0x00000007 in ?? ()
#3  0x00000007 in ?? ()
#4  0x00000007 in ?? ()
```

Изучаем параметры gcc

```
#5 0x00000006 in ?? ()  
#6 0x0000000f in ?? ()  
#7 0x00000000 in ?? ()
```

При втором вызове `test`, программа остановилась, и `gdb` вывел трассировку стека. В нормальной ситуации, `main()` должна идти в фрейме №2, но мы видим только знаки вопроса. Запомните, что я сказал про расположение стека: отсутствие указателя фрейма мешает `gdb` находить расположение сохраненного возвращаемого адреса в фрейме №2.

Опции, относящиеся к отладке.

Каждый иногда нуждается в отладке его или её кода. Когда это время приходит, обычно вы запускаете `gdb`, ставите точки останова там и тут, анализируете бэктрейсы, и так далее, чтобы выявить расположение нарушающего работу кода. А что получаете на самом деле? Если вы не используете опции отладки, вы просто получаете адрес, указывающий на регистр EIP.

Вот в чем проблема, в действительности вы не хотите адрес. Вы хотите, чтобы `gdb` или другой отладчик просто показал требуемые строки. Но `gdb` не может этого сделать без определенного вида помощи. Эта помощь, называемая отладкой с приписываемыми форматами записи (Debugging With Attributed Record Formats — DWARF), помогает вам выполнять отладку на уровне исходного кода.

Как это сделать? Используйте `-g` при компиляции в объектный код, то есть:

```
gcc -o -g test test.c
```

Что такого добавляет `gcc`, что отладчик может сопоставлять адрес с исходным кодом? Вам нужна `dwarfdump` [7] чтобы узнать это. Это утилита находится внутри тарболла `libdwarf` или в RPM, так что вы не найдете её в виде отдельного пакета. Скомпилируйте её сами или просто установите из репозитория вашего дистрибутива; оба варианта должны сработать. В этой части статьи я использую версию 20060614 RPM.

Используя `readelf`, вы можете отметить, что в неотлаженной версии первого листинга существует 28 разделов:

```
$ readelf -S ./non-optimized
```

Но в отлаженной версии есть 36 разделов. Новые разделы:

- * `debug_arranges`
- * `debug_pubnames`
- * `debug_info`
- * `debug_abbrev`
- * `debug_line`
- * `debug_frame`
- * `debug_str`
- * `debug_loc`

Вам не нужно копаться во всех этих разделах; для быстрого изучения, будет

достаточно рассмотреть `.debug_line`. Команда, которая вам нужна:

```
$ /path/to/dwarfdump -l
```

Вот пример того, что вы получите:

```
.debug_line: line number info for a single cu

Source lines (from CU-DIE at .debug_info offset 11):

      [row,column]          //
/code/./non-optimized.c:  [ 3,-1]          0x8048384          // new statement
/code/./non-optimized.c:  [ 5,-1]          0x8048395          // new statement
.....
```

Интерпретация этих сообщений довольно простая. Возьмите первую запись (идущую за строкой

```
line number 3 in file non-optimized.c is located in address 0x8048384.
```

gdb дает ту же информацию:

```
$ gdb non-optimized-debugging

(gdb) l *0x8048384

0x8048384 is in main (./non-optimized.c:3).
```

readelf также предоставляет похожую информацию, используя `--debug-info`:

```
$ readelf --debug-dump=line

Line Number Statements:

Extended opcode 2: set Address to 0x8048384

Special opcode 7: advance Address by 0 to 0x8048384 and Line by 2 to 3

Advance PC by constant 17 to 0x8048395

Special opcode 7: advance Address by 0 to 0x8048395 and Line by 2 to 5

....
```

И `readelf`, и `dwarfdump` могут анализировать информацию отладки, так что вы вольны выбирать сами.

Что вы должны понимать, так это то, что исходный код сам по себе не встроен в объектный файл. На самом деле, отладчик должен проверять отдельный файл исходного кода. Запись в колонке

```
$ gcc -O2 -ggdb -o debug-optimized listing-one.c

$ readelf --debug-dump=line debug-optimized
```

Изучаем параметры gcc

```
..  
Special opcode 107: advance Address by 7 to 0x80483aa and Line by 4 to 11  
...
```

Но что говорит gdb?

```
$ gdb debug-optimized  
  
(gdb) l *0x80483aa  
  
0x80483aa is in main (./non-optimized.c:11).  
  
...  
  
11             printf("acc = %lu\n", acc);  
  
...  
  
(gdb) disassemble main  
  
...  
  
0x080483aa :    add     $0x6,%edx  
  
0x080483ad :    cmp     $0x1388,%eax  
  
...
```

Здесь вы видите полное расхождение. Изучая одну информацию об отладке, вы будете ожидать, что указанный адрес содержит что-то вроде инструкции CALL. Но в действительности, вы получите инструкции ADD и CMP, что больше похоже на конструкцию цикла. Это побочный эффект действий оптимизации — в этом случае меняется порядок инструкций. Так что возьмите себе за правило не смешивать опции -g (или её варианты) с -O.

Опции, управляющие стадиями компиляции.

В целях изучения, иногда вы хотите узнать, как ваш исходный код трансформируется в исполняемый. К счастью, gcc предоставляет вам опции для остановки на любой стадии обработки. Вспомните, что gcc имеет несколько стадий завершения, — например, компоновку. Есть такие опции:

- -s останавливает на стадии ассемблирования, но пропускает компоновку. Результатом является объектный код.
- -E останавливает на стадии препроцессинга. Все директивы препроцессора развернуты, так что вы видите только чистый код.
- -S останавливает после компиляции. Она оставляет вас с ассемблерным кодом.

с наиболее часто используется, когда у вас есть несколько исходных файлов и вы хотите скомбинировать их для получения итогового исполняемого файла. Так что, вместо такого:

```
$ gcc -o final-binary test1.c test2.c
```

будет лучше разделить их так:

```
$ gcc -c -o test1.o test1.c
```

```
$ gcc -c -o test2.o test2.c
```

и затем:

```
$ gcc -o final-binary ./test1.o ./test2.o
```

Возможно, вы заметили, что такая же последовательность используется, если вы собираете программу, используя Makefile. Преимущество использования -с ясно: вам нужно перекомпилировать только измененные исходные файлы. Только фаза, на которой переделывается компоновка всех объектных файлов, и это очень экономит время, особенно в больших проектах. Очевидным примером этого является ядро Linux.

Е будет полезна, если вы хотите посмотреть, как ваш код в действительности выглядит после разворачивания макросов, определений и т.п. Возьмите следующий листинг в качестве примера:

```
#include

#define A 2

#define B 4

#define calculate(a,b) a*a + b*b

void plain_dummy()
{
    printf("Just a dummy\n");
}

static inline justtest()
{
    printf("Hi!\n");
}

int main(int argc, char *argv[])
{
#ifdef TEST
    justtest();
}
```

Изучаем параметры gcc

```
#endif

    printf("%d\n", calculate(A,B));

    return 0;

}
```

Скомпилируем его следующим образом:

```
$ gcc -E -o listing2.e listing2.c
```

Учтите, что мы не указываем параметров `-D`, что означает, что `TEST` не определен. Так и что мы имеем в препроцессорном файле?

```
void plain_dummy()

{

    printf("Just a dummy\n");

}

static inline justtest()

{

    printf("Hi!\n");

}

int main(int argc, char *argv[])

{

    printf("%d\n", 2*2 + 4*4);

    return 0;

}
```

А где вызов `justtest()` внутри `main()`? Нигде. `TEST` не определен — вот почему код исключен. Вы также можете видеть, что макрос `calculate` уже развернут в умножение и сложение констант. В конечной исполняемой форме, эти числа будут заменены результатами операций. Как вы видите, `-E` довольно удобна для проверки корректности директив.

Обратите внимание, что `plain_dummy()` все ещё здесь, не смотря на то, что она ни разу не вызывается. Это не удивительно, так как ещё не была произведена компиляция, вот почему не произошло исключение «мертвого» кода на этой стадии. `stdio.h` также развернут, но не показан в листинге выше.

Я нашел интересное приложение использования `-E` в качестве утилиты создания

HTML. Вкратце, она помогает вам перенимать обычные действия в программировании, такие как модуляризация кода и макросы в мир HTML — то, что не может быть сделано на чистом HTML. -S дает вам код на ассемблере, больше похожий на то, что вы видите с помощью `objdump -d/-D`. Хотя с помощью -S вы продолжите видеть директивы и имена символов, который делают код проще к изучению. Например, вызов `printf("%d\n", 20)` может быть трансформирован в:

```
.section      .rodata.str1.1,"aMS",@progbits,1

.LC0:

    .string "%d\n"

...

    movl     $20, 4(%esp)

    movl     $.LC0, (%esp)

    call     printf
```

Вы можете видеть, что format string `%d` помещена в область данных, доступную только для чтения (`.rodata`). Также, вы можете удостовериться, что аргументы помещаются в стек справа налево, со строкой форматирования на верху стека.

Заключение.

gcc дает нам много полезных опций для превращения нашего кода во что угодно. Понимая, что эти опции делают на самом деле, мы можем сделать программу быстрее и ловчее. Хотя, не будьте целиком от них зависимы: вы должны больше внимания уделять написанию эффективного и хорошо структурированного кода.

Благодарности.

Я хочу поблагодарить сообщества на irc каналах OFTC (`#kernelnewbies` и `#gcc`) и `#osdev` (Freenode) за их полезные идеи.

Ссылки

1. [Статья в Википедии о Препроцессоре](#);
2. [Статья в Википедии о Компиляторе](#);
3. [Статья в Википедии об Ассемблере](#);
4. [Статья в Википедии о Компоновщике](#).
5. [Пример переупорядочивания кода](#) (англ.);
6. [Frame pointer omission \(FPO\) optimization and consequences when debugging, Часть 1 и Часть 2](#) (англ.);
7. [Описание DWARF](#) (англ.);
8. [Описание stabs](#) (англ.);
9. [Описание COFF](#) (англ.);
10. [Описание XCOFF \(варианта COFF\)](#) (англ.);

Изучаем параметры gcc

11. [Использование препроцессора C в качестве утилиты создания HTML](#) (англ.);
12. [Документация по gcc](#) (англ.);
13. [AMD Athlon Processor x86 Code Optimization Guide](#) (англ., pdf).

Создание LiveCD дистрибутива на базе Gentoo Linux

Ссылка на оригинал: <http://linuxportal.vrn.ru/?q=node/58>

Автор: [Pizgin](#)

С версии: 1.5

Версия статьи: 1.4 от 23.03.2008

Введение

В руководстве рассказывается о том как создать свой LiveCD диск на основе Gentoo Linux. Диск будет полностью русифицирован, произведена установка KDE и обеспечено автоматическое монтирование flash накопителей. В качестве загрузчика рассматривается ISOLINUX. Раньше был GRUB, но пришлось от него отказаться, так как он не работает на некоторых ноутбуках. Созданный по этой инструкции диск можно будет постоянно совершенствовать, устанавливать новые пакеты, вообще можно делать все то что можно делать с обычным дистрибутивом установленным на вашем ПК.

Подготовительные действия

Для сборки необходимо создать определенную структуру каталогов, где создавать - в принципе без разницы. Я создавал в домашней директории. Структура следующая:

```
livedcd
conf          // набор конфигурационных файлов
source        // создаваемый дистрибутив
scripts       // набор вспомогательных скриптов
distr         // исходники: portage, stage, может картинки какие-то и пр.
target        // здесь сборочный скрипт будет создавать squashfs образ
```

Создаем:

```
$ cd ~
```

```
$ mkdir -p livedcd/{conf,distrib,scripts,source,target}
```

Теперь из прикрепленного к статье файла, берем его содержимое и расписываем поуказанным директориям.

Собирать livedcd будем из второй стадии. Скачиваем ее из сети и копируем в distr и распаковываем в каталог сборки. Распаковывать следует с root правами, иначе будут проблемы с созданием устройств в каталоге /dev.

```
# tar -C source/ -pxjvf distrib/stage2-i686-2007.0.tar.bz2
```

Архив с портежами и дистфайлами распаковывать в каталог сборки не будем. Вместо этого скрипты будут автоматически монтировать эти каталоги от основной системы.

Для того чтобы легко отличать консоль в chroot окружении от консоли основной системы, рекомендую для первой изменить приглашение командной строки и вместо просто "#написать например "(LIVECD) #". Для этого копируем заготовленный в conf директории файл root/bashrc в каталог source/root/ (добавив в начало названия точку).

Для сборки образа, на ПК предварительно должны быть установлены пакеты squashfs-tools и cdrtools. Первый для работы с файловой системой SquashFS, второй для записи CD дисков.

emerge -av squashfs-tools cdrtools

Собираем базовую систему

Сейчас можно входить (chroot'иться как еще говорят) в собираемую систему и начинать подгонять ее под свои потребности.

cd scripts

./enter.sh

Если все прошло нормально - на экране не должно быть никаких ругательных сообщений, а приглашение командной строки выглядеть вот так: "(LIVECD) #". Сейчас мы находимся в только-что распакованной из stage2 системе.

Общий план работ таков:

- 1) Установить имя машины/домена, профиль, дописать USE флагов в make.conf, создать /etc/fstab.
- 2) Установить часовой пояс, перевести часы в режим Local, сгенерировать русские локали, установить русскую раскладку клавиатуры и экранный шрифт.
- 3) Выполнить emerge -e system и emerge -e world для получения Stage3.
- 4) Не забыть установить пароль root'у.
- 5) Собрать ядро, настроить загрузчик и попробовать перезагрузиться.
- 6) Установить свои приложения.
Создать ISO образ и записать его на диск.

Первые два пункта за вас может выполнить подготовленный скрипт "_prepare.sh". Написан он был потому как собрать livedcd получилось не с первого раза, и делать одно и тоже на только-что распакованных stage-2 порядком надоело. Его нужно скопировать куда нибудь в source директорию и выполнить один раз для свежескопированного stage-2. Скопировать можно например в /root или прямо в корень (важно не забыть перед созданием ISO образа его оттуда удалить).

Итак начинаем:

Напомню что все действия происходят в chroot окружении.

1. Подготавливаем и русифицируем систему.
(LIVECD) # **./_prepare.sh**
(LIVECD) # **rm _prepare.sh**
2. Устанавливаем основные утилиты для управления пакетами
(LIVECD) # **emerge -av gentoolkit**
3. Собираем Stage3
(LIVECD) # **emerge -e system**
Обновляем конфигурационные файлы обновленных приложений
(LIVECD) # **dispatch-conf**
Здесь нужно быть внимательным и не затереть те конфигурационные

файлы которые

мы сами изменяли (или их изменил скрипт `_prepare.sh`). Это касается русского шрифта, раскладки клавиатуры и прочее. Вообще прежде чем в ответ на вопрос `dispatch-conf`'а жать 'u', внимательно посмотрите какой файл он хочет обновить. Если это файлы: `clock`, `consolefont`, `hostname` или `keymaps` - жмите 'z' (не обновлять).

Проверяем целостность зависимостей системы

```
(LIVECD) # revdep-rebuild
```

Тоже самое про мир, пересобираем, обновляем конфигурационные файлы и проверяем целостность зависимостей.

```
(LIVECD) # emerge -e world
```

```
(LIVECD) # dispatch-conf
```

```
(LIVECD) # revdep-rebuild
```

Пересборка system на Turion64X2 заняла приблизительно 2.5 часа, world - 3 ч.

При emerge -e system могут быть проблемы с perl. Если такое случиться

-делать так (ставиться будет примерно минут 15):

```
(LIVECD) # emerge --oneshot gdbm db
```

```
(LIVECD) # emerge -N --oneshot --nodeps perl
```

После можно снова пробовать emerge -e system.

Возможно где-то в середине сборки system прервется с ошибкой на пакете `sys-apps/attr`. Ошибка будет выглядеть так: `"libexpat.so.0: cannot open shared objects file: No such file or directory"`. Если это случилось - создаем символическую ссылку с `libexpat.so` на `libexpat.so.0`, и затем пробуем продолжить сборку, т.е. делаем

```
(LIVECD) # ln -s /usr/lib/libexpat.so /usr/lib/libexpat.so.0
```

```
(LIVECD) # emerge --resume
```

Обнаружил небольшой недочет при сборке с использованием portage от 05.03.2008. emerge -e system прерывается на пакете 'which-2.19' с ошибкой `"error: readline/rldstdc.h: No such file or directory"`. На момент сборки этого пакета уже должна стоять библиотека readline, но ее нет. Устанавливаем ее сами и продолжаем сборку system. Если подробно, то:

```
(LIVECD) # emerge -av readline
```

```
(LIVECD) # emerge --resume
```

Если будут еще какие-либо проблемы - скачайте или обновитесь до самого свежего архива портежей. Не поможет - идите на bugzilla.gentoo.org.

4. Устанавливаем пароль root'u

```
(LIVECD) # passwd
```

5. Создаем пользователя livecd

```
(LIVECD) # useradd -m -G users,wheel,audio,video,cdrom,cdrw,usb -s /bin/bash livecd
```

```
(LIVECD) # passwd livecd
```

6. Устанавливаем splash темы для красивой графической загрузки

```
(LIVECD) # emerge -av splash-themes-livecd
```

7. Устанавливаем и компилируем ядро

```
(LIVECD) # emerge -av gentoo-sources
```

genkernel должен быть не старше чем 3.4.10_pre4. На момент написания этого руководства такой версии в стабильной ветке небыло. Если у вас тоже-самое - разрешаем устанавливать его из тестовой (~x86), для этого выполним

```
(LIVECD) # echo 'sys-kernel/genkernel ~x86' >> /etc/portage/package.keywords
```

```
(LIVECD) # emerge -av genkernel
```

```
(LIVECD) # genkernel all —gensplash=livedcd-2007.0
```

8. Устанавливаем и добавляем в автозагрузку Gentoo LiveCD скрипты

Снимаем маскировку (установлена разработчиками для того чтобы предупредить о том, что скрипты предназначены только для использования вместе с livedcd)

```
(LIVECD) # echo 'app-misc/livedcd-tools' >> /etc/portage/package.unmask
```

```
(LIVECD) # echo 'x11-misc/mkxf86config' >> /etc/portage/package.unmask
```

```
(LIVECD) # echo 'sys-apps/hwsetup' >> /etc/portage/package.unmask
```

livedcd-tools нужен версии не ниже 1.0.40_pre1. На момент написания этого руководства такой версии в стабильной ветке небыло. Если у вас тоже-самое - разрешаем устанавливать его из тестовой (~x86), для этого выполним

```
(LIVECD) # echo 'app-misc/livedcd-tools ~x86' >> /etc/portage/package.keywords
```

```
(LIVECD) # emerge -av livedcd-tools
```

libkudzu нужен не ниже чем 1.2.57.1, если будет устанавливаться более старая версия - отвечаем 'no' и разрешаем libkudzu из тестовой ветки.

```
(LIVECD) # echo 'sys-libs/libkudzu ~x86' >> /etc/portage/package.keywords
```

```
(LIVECD) # rc-update add autoconfig default
```

9. Устанавливаем загрузчик

```
(LIVECD) # emerge -av syslinux
```

```
(LIVECD) # mkdir /boot/isolinux
```

```
(LIVECD) # cp /usr/lib/syslinux/isolinux.bin /boot/isolinux
```

```
(LIVECD) # cp /boot/kernel-genkernel-x86-2.6.23-gentoo-r6/boot/isolinux/vmlinuz
```

```
(LIVECD) # cp /boot/initramfs-genkernel-x86-2.6.23-gentoo-r6 /boot/isolinux/initrd
```

Из директории с конфиг. файлами копируем в /boot/isolinux файл isolinux.cfg.

10. Создаем образ и пробуем его загрузить. Выходим из chroot окружения и запускаем скрипт "build.sh"

```
(LIVECD) # exit
```

```
# ./build.sh
```

Процесс сборки образа длится примерно минут 5. После него забираем iso файл в директории livedcd. Можно записать его на болванку, но лучше для этих целей поставить например VirtualBox или VMWare, потому как удобнее и быстрее.

Записать на CD можно так:

```
# cdrecord -v -eject speed=10 fs=8m dev=/dev/cdrw image.iso
```

или если это DVD то так:

```
# growisofs -dvd-compat -Z /dev/dvd=image.iso
```

Образ должен загрузиться, когда дойдет до приглашения - введите root и ваш пароль. Если все так - пол дела сделано. LiveCD грузиться. Сейчас желательно создать архив с каталогом livedcd на случай если при дальнейших манипуляциях что нибудь пойдет не так - можно будет откатиться.

Установка KDE

1. Устанавливаем Xorg
(LIVECD) **# emerge -av xorg-server**
2. Устанавливаем оригинальный драйвера nVidia
(LIVECD) **# emerge -av nvidia-drivers**
3. Правим таблицу соответствия устройство - драйвер для nVidia карт. Открываем файл /usr/share/hwdata/Cards.
 - 3.1. Находим строку NAME NVIDIA Legacy и меняем название драйвера 'vesa' на 'nv'.
 - 3.2. Находим строку NAME NVIDIA GeForce и меняем название драйвера с 'vesa' на 'nvidia'.
4. Устанавливаем минимальный набор KDE(LIVECD) **# emerge -av kdm kdebase-startkde kde-i18n**
5. Добавляем в автозапуск xdm и указываем в нем запускаемый оконный менеджер В файле /etc/conf.d/xdm переменной DISPLAYMANAGER присваиваем значение "kdm"
(LIVECD) **# rc-update add xdm default**
6. Настройка автомонтирования съемных устройств
(LIVECD) **# emerge -auv dbus hal pmount**
(LIVECD) **# rc-update add dbus default**
(LIVECD) **# rc-update add hald default**
(LIVECD) **# gpasswd -a livedcd plugdev**
7. Включаем русскую раскладку и переключатель en/ru
 - 7.1. Открываем файл /usr/sbin/mkxf86config.sh и удаляем строку вида
"-e 's|"XkbLayout" *"[^"]*"|"XkbLayout" "'\${XKEYBOARD}""'|g;'"\${DEADKEYS}"" \"
 - 7.2. Открываем файл /etc/X11/xorg.conf.in и в секции InputDevice, Keyboard0 меняем последние три строчки на следующие:
Option "XkbLayout" "us,ru(winkeys)"
Option "XkbVariant" "us"
Option "XkbOptions" "grp:alt_shift_toggle,grp_led:scroll"

Что еще можно сделать

* Автологин в текстовой консоли не под root'ом

1. Устанавливаем mingetty

(LIVECD) # **emerge -av mingetty**

2. Прописываем его в /etc/inittab вместо agetty.

Как было:

```
c1:12345:respawn:/sbin/agetty 38400 tty1 linux
```

Как нужно исправить:

```
c1:12345:respawn:/sbin/mingetty --autologin root --noclear tty1
```

Естественно что вместо root можно вписать любого пользователя.

3. Правим файл /sbin/rc

Находим вот такую секцию (приблизительно это строка N 500)

```
if [ -f "/sbin/livecd-functions.sh" -a -n "${CDBOOT}" ]
then ebegin "Updating inittab" livecd_fix_inittab eend $? /sbin/telinit
q &>/dev/null fi
```

и делаем ее такой

```
if [ -f "/sbin/livecd-functions.sh" -a -n "${CDBOOT}" ]
then ebegin "Updating inittab" /bin/true #livecd_fix_inittab eend $?
/bin/true #/sbin/telinit q &>/dev/null
fi
```

Свое сообщение после init'a

Открываем файл /sbin/rc, ищем строку вида 'echo -e " Copyright' и добавляем ниже нее свое сообщение.

Автологин в KDE

Открываем файл /usr/kde/3.5/share/config/kdm/kdmrc и прописываем пользователя в строки 'DefaultUser' и 'AutoLoginUser'.

Красивый, настроенный и подогнанный под себя рабочий стол

Запускаем созданный LiveCD, загружаемся в KDE и настраиваем его под себя. Все, шрифты, поведение окон, курсор занятости, панели и пр. Затем сжимаем свой домашний каталог (/home/livecd) и копируем его на флешку. Выходим из LiveCD, chroot'имся в source и распаковываем в каталог /home/livecd/ сохраненные на флешке настройки.

Приложение

Версии основных используемых пакетов

sys-fs/squashfs-tools-3.1_p2

app-misc/livecd-tools-1.0.40_pre1

sys-kernel/genkernel-3.4.10_pre4

sys-kernel/gentoo-sources-2.6.23-r9

sys-apps/hwdata-gentoo-0.3

sys-apps/hwsetup-1.2

x11-misc/mkxf86config-0.9.9

stage2-i686-2007.0.tar.bz2

portage-20080305.tar.bz2

Тюнинг Gentoo

Ядро 2.6: Тонкости настройки

Ссылка на оригинал: http://ru.gentoo-wiki.com/Ядро_2.6

С версии: 1.5

Параметры виртуальной памяти

Ядро Linux позволяет регулировать параметры свопинга двумя способами: автоматически, исходя из того, насколько активно используется та или иная программа, либо вручную задавая агрессивность свопинга.

Содержимое файла `/proc/sys/vm/swappiness` определяет агрессивность свопинга, а истинность (неравенство нулю) значения в файле `/proc/sys/vm/autoswappiness` определяет какой из двух режимов будет использоваться. Таким образом, если вы хотите установить агрессивность свопинга вручную, то следует воспользоваться командой:

```
echo 0 > /proc/sys/vm/autoswappiness
```

чтобы отключить автоматический режим и уже затем установить желаемый режим вручную. Значения в файле `/proc/sys/vm/swappiness` изменяются от 0 до 100. При минимальном значении ядро стремится большую часть содержимого держать в физической памяти, при максимальном стремится к экономии физической памяти. Чтобы максимально сэкономить оперативную память введите:

```
echo 100 > /proc/sys/vm/swappiness
```

По умолчанию значение 60, но если поставить 0, это должно побудить систему пользоваться подкачкой только в критических ситуациях. Для медленных систем может ускорить работу. Значение 100 не рекомендуется для тех у кого достаточно медленный жесткий диск. Казалось бы, это значение вообще не желательно для использования, однако, Andrew Morton, один из разработчиков ядра Linux, заявляет, что на своих рабочих компьютерах (десктопах, не серверах) использует значение 100, т.к. считает, что ему совершенно ни к чему эти сотни занятых мегабайт оперативной памяти. И он в некоторых случаях прав, так как известно, что Linux использует свободную оперативную память для кэширования дисковых операций, таким образом, значение близкое или равное 100 может и ускорить систему. Таким образом, решать вам, исходя из количества и ресурсоемкости используемых вами задач. Хотя, вряд ли вы так же активно используете свой десктоп, как Эндрю ;)

Также, возможно заставить систему устанавливать нужный режим при загрузке, прописав в файл `/etc/sysctl.conf` строку

Файл: `/etc/sysctl.conf`

```
vm.swappiness=100
```

разумеется, вместо 100 следует подставить то значение, которое вам подходит.

Руководство по предварительному связыванию в Gentoo Linux

Ссылка на оригинал: <http://www.gentoo.org/doc/ru/prelink-howto.xml>

С версии: 1.0

Перенесено в: 1.5

1. Введение

Что такое предварительное связывание (Prelink) и как оно может помочь мне?

Большинство приложений используют разделяемые библиотеки. Эти разделяемые библиотеки должны быть загружены в память во время выполнения программы, при этом необходимо определить различные символьные ссылки. Для большинства небольших программ динамическое связывание происходит очень быстро. Но для программ, написанных на C++, имеющих много библиотечных зависимостей, динамическое связывание может занять много времени.

На большинстве систем, библиотеки обычно остаются неизменными и когда программа запускается, операции необходимые для связывания, каждый раз одинаковы. Предварительное связывание использует данную особенность, выполняя связывание и сохраняя библиотеки в выполняемом файле, фактически уже связанными. Для связывания, вам потребуется ld-linux.so в glibc; для распознавания предварительного связывания версия glibc должна быть $\geq 2.3.1-r2$

Предварительное связывание может уменьшить время загрузки приложений. Например, типичные программы KDE могут загружаться на 50% быстрее. Необходимым условием является перезапуск процесса предварительного связывания каждый раз при обновлении библиотек для программ, которые на них ссылаются.

Резюме

Предварительное связывание производится программой, имеющей название, как не странно, prelink. Она изменяет выполняемый файл таким образом, что он запускается быстрее.

Если зависимые от приложения библиотеки были изменены после того, как были предварительно связаны с приложением, то они потребуют повторного предварительного связывания, иначе вы потеряете преимущества в скорости загрузки.

Изменения выполняемого файла полностью обратимы, поскольку программа prelink имеет функцию undo.

Новые версии portage могут обрабатывать, при помощи программы prelink,

изменяющиеся контрольные суммы MD5 и поля mtime выполняемых файлов.

У вас должен быть glibc-2.3.1-r2 или более новый, и бинарные файлы должны быть скомпилированы с binutils-2.13.90.0.xx или выше.

2. Инициализация предварительного связывания

Установка программ

Примечание: Я подразумеваю, что у вас есть Gentoo-1.4, который был собран при помощи gcc-3.2 или более поздней версии и binutils-2.13.90.0.xx. Это требуется для того, что бы выполняемые файлы могли быть предварительно связаны.

Предупреждение: Должен быть установлен glibc 2.3.1 или более новый, иначе программа prelink может испортить все ваши выполняемые файлы!

Для начала обновите свое дерево портежей, потому что многие из необходимых приложений часто обновляются и постоянно выходят исправления ошибок.

Листинг 2.1: Обновление вашего дерева портежей

```
# emerge sync
```

Далее, убедитесь, что у вас установлен portage-2.0.26 или более новый. Это требуется для того, что бы portage могло распознать предварительно связанные выполняемые файлы и при необходимости корректно их деинсталлировать. Т.к. предварительное связывание изменяет контрольную сумму MD5 бинарных файлов.

Листинг 2.2: Проверка версии дерева портежей

```
# emerge ">=portage-2.0.46"
```

Теперь вы можете вызвать emerge для программы предварительного связывания. Процесс emerge автоматически проверит, что ваша система может безопасно выполнять предварительное связывание.

Листинг 2.3: Устанавливаем предварительное связывание

```
# emerge prelink
```

Некоторые люди, в ходе выполнения тестов, получают ошибки выполняя emerge для программы prelink. Тесты размещены в пакете из соображений безопасности, предварительное связывание будет непредсказуемым, если их отменить. Эти ошибки обычно связаны с базовыми пакетами, такими как binutils, gcc и glibc. В таком случае попробуйте выполнить emerge этих пакетов повторно.

Примечание: Если ошибки продолжают появляться, попробуйте скомпилировать и протестировать программу prelink самостоятельно (`./configure ; make ; make check`). В случае сбоя вы можете просмотреть *.log файлы в директории с набором тестов. Они могут дать вам некоторые полезные подсказки.

Если вы имеете набор действий, которые демонстрируют ошибки в другой системе, пожалуйста отправьте его на Stefan Jones.

Настройка

Portage автоматически генерирует файл `/etc/prelink.conf`, который говорит программе prelink какие файлы требуется предварительно связать.

К сожалению, вы не сможете предварительно связать файлы, которые были скомпилированы старыми версиями пакета binutils. Большинство таких приложений поставляется уже скомпилированными, не имея исходных кодов, и устанавливаются в каталоге `/opt`. Создайте следующий файл, для того, что бы запретить программе prelink использовать предварительное связывание для подобных файлов.

Листинг 2.4: `/etc/env.d/99prelink`
`PRELINK_PATH_MASK="/opt"`

Примечание: Вы можете добавить больше или меньше директорий, в список, разделенный двоеточиями.

3. Предварительное связывание

Применение предварительного связывания

Я использую следующую команду для предварительного связывания всех бинарных файлов в директориях, заданных в файле `/etc/prelink.conf`.

Листинг 3.1: Предварительное связывание заданных файлов
`# prelink -afmR`

Предупреждение: Наблюдалось, что при наличии небольшого свободного пространства в момент выполнения предварительного связывания для всей системы, существует вероятность усечения выполняемых файлов. Результатом может стать испорченная система. Используйте команду `file` или `readelf` для проверки целостности выполняемых файлов. Также вы можете заранее проверить наличие свободного места командо `df -h`

Описание опций:

- `-a` All, применить предварительное связывание ко всем выполняемым

файлам.

- **-f** Вынуждает повторить предварительное связывание для файлов, которые уже подвергались ему. Это требуется т.к. программа prelink прекращает обработку существующих старых связанных файлов, зависимые библиотеки которых могли измениться.
- **-m** Сохраняет пространство виртуальной памяти. Это требуется, если вы имеете много библиотек, которые необходимо связать.
- **-R Random, --** Выбирает случайный порядок адресов, это увеличивает безопасность против переполнений буферов.

Примечание: Более подробно об опциях можно узнать, набрав `man prelink`

4. Известные проблемы и методы их решения

"Предварительное связывание не работает с разделяемой библиотекой, собранной без опции PIC"

Причиной данной проблемы являются объектные файлы разделяемой библиотеки, скомпилированные без опции `gcc -fPIC`.

Далее следует список проблемных библиотек и список пакетов, для которых требуется повторно вызвать `emerge`, в случае возникновения данной проблемы.

Листинг 4.1: Исправления

(Для библиотеки ORBit /usr/lib/liblIOP.so.0.5.17)

emerge ">=sys-apps/tcp-wrappers-7.6-r4" ORBit

(Для библиотеки zlib /usr/lib/libz.so.1.1.4)

emerge ">=sys-libs/zlib-1.1.4"

(Для svgalib, /usr/lib/libsvglib.so.xx)

emerge ">=media-libs/svgalib-1.9.16"

(Для библиотеки XFree OpenGL, libGLU.so.1)

emerge ">=x11-base/xfree-4.2.1-r2"

(Для libpcap.so.0.6)

emerge ">=net-libs/libpcap-0.7.1-r2"

(Для библиотеки lcms, /usr/lib/liblcms.so.1)

emerge ">=media-libs/lcms-1.09"

Примечание: Многие библиотеки статично связаны с `zlib` и/или `tcp-wrappers`, так что сначала попробуйте выполнить `emerge` для них, после чего, повторно запустите `emerge` для проблемных библиотек.

Если вы имеете проблемы с предварительным связыванием QT/KDE, то сначала

попытайтесь обновить x11-base/xfree до версии 4.2.1-r2 или более новой и x11-libs/qt до версии 3.1.0-r1 или более новой. Если QT так и не работает, тогда попробуйте скомпилировать его без поддержки xinerama, задав строку `myconf="-no-xinerama ${myconf}"` в файле `ebuild` для `qt`.

Далее приводится список библиотек, которые пока не работают или не могут работать вообще:

- Библиотеки в пакете `wine`, включая `windex`. Они в любом случае не могут ускорить выполняемые файлы MS Windows.
- Библиотека в `media-video/mjpgtools`, `/usr/lib/liblavfile-1.6.so.0`

Если у вас есть проблемы с библиотекой, которой нет в списке, пожалуйста сообщите об этом, предпочтительно, добавив `-fPIC` к соответствующим `CFLAGS`.

Прерывание предварительного связывания файла, с выводом сообщения, вида: "1631 Aborted"

Вам необходимо использовать опцию `-f` программы `prelink`; т.е. повторить предварительное связывание для всей системы с нуля. Попробуйте запустить `prelink -af`

"Ошибка: <file>: error while loading shared libraries: unexpected reloc type..."

Это было исправлено в пакете `sys-libs/glibc-2.3.1-r2` 2002/11/18, выполните `emerge` для `glibc`, если у вас более старая версия.

Так же запустите `prelink -u -a -m`; `prelink -a -m` это должно помочь. Если все это не помогает, просто запустите `prelink -u <file>`

У меня проблемы с библиотеками Nvidia OpenGL

Ускоренные библиотеки `OpenGL` идущие с пакетом `nvidia-glx` скомпилированы нестандартным образом, и программа `prelink` выводит массу предупреждений. Здесь нет повода для беспокойства, и это не может быть исправлено кем-либо помимо компании `nvidia`. Вы всегда можете вернуться к `xfree` версии библиотеки `libGL.so`, если вам не требуется ускорение 3D. Хотя драйвер `xfree nvidia` работает нормально.

После предварительного связывания моей системы, некоторые статичные выполняемые файлы более не работают

Там где участвует `glibc`, не бывает такой вещи как 100% статичный выполняемый файл. Если вы статично скомпилировали выполняемый файл с `glibc`, он может продолжать оставаться зависимым от других системных файлов. Далее

приводится объяснение от Dick Howell.

"Я предполагал, что все должно находиться в скачанном файле и не должно быть никаких зависимостей с локальными библиотеками на целевой системе. К сожалению, в Linux, и я думаю везде, где используется GLIBC, это до сих пор не совсем верно. Существует библиотека "libnss" (name service switch - переключатель сервиса имен, некоторые люди думают, что это network security system - сетевая система безопасности) которая предоставляет функции для доступа к различным базам данных для аутентификации, сетевой информации и для других вещей. Эта библиотека предполагает сделать приложения независимыми от отдельно сконфигурированных сетевых окружений машины. Хорошая идея, но изменения GLIBC может вести к проблемам ее загрузки. И вы не можете статически связать "libnss" так как она сконфигурирована для каждой машины индивидуально. Я думаю, что, главным образом, это происходит из-за статического связывания других GLIBC библиотек, особенно "libpthread", "libm" и "libc", откуда приходят несовместимые вызовы к функциям "libnss"

**Предварительное связывание обрывается с ошибкой

```
prelink: dso.c:306: fdopen_dso: Assertion `j == k' failed.
```**

Это известная проблема, любезно продиагностирована здесь. Программа prelink не может справиться с выполняемыми файлами сжатыми методом UPX. Для версии prelink-20021213 нет другого способа решить эту проблему, кроме как спрятать выполняемые файлы во время предварительного связывания. Смотрите раздел конфигурации выше, где обсуждается вопрос о том, как это осуществить.

5. Заключение

Предварительное связывание может решительно ускорить время запуска для некоторых больших приложений. Его поддержка встроена в дерево портежей. Предварительное связывание безопасно, т.к. вы всегда можете отменить его действие для любого выполняемого файла, если вы столкнетесь с какими-либо проблемами. Помните что когда вы обновляете glibc или другие библиотеки, с которыми вы осуществили предварительное связывание, вам надо будет перезапустить prelink. В общем, удачи!

Тюнинг системы и запуска некоторых программ

Ссылка на оригинал: <http://ru.gentoo-wiki.com/Тюнинг системы и запуска некоторых программ>

С версии: 1.5

Введение

Многие программы и части системы Linux работают по-умолчанию не на полную "мощность", т.к. они должны корректно работать на широком спектре системных конфигураций. Иногда, это может создать иллюзию о том, что Linux - более медленная система чем *другие*. Это не так. В данной статье будут рассмотрены наиболее важные настройки, которые дадут вам наибольший прирост скорости и эффективности системы.

Тонкая настройка дисковой подсистемы (hdparm, планировщики ввода-вывода) перенесена в [отдельную статью](#)

mplayer

- Желательно использовать `-vo xv` или `-vo xvidix` или `-vo xv`, чтобы перенести нагрузку с CPU на вашу видео-карту
- Собирайте mplayer с поддержкой особенностей именно вашего процессора (используйте `emerge mplayer -pv` - там много всего полезного), многие части кода mplayer оптимизированы для использования на конкретных процессорах.
- При проигрывании используйте

```
hdparm -u1 -A1 -a 128 /dev/hda (или что там у вас вместо hda)
```

- Использование команды `nice` при запуске mplayer с приоритетами `-5` и `-10` также дает преимущества

XMMS

- Как и для mplayer не забудьте собрать XMMS с поддержкой особенностей вашего процессора
- Установите значение для `output-buffer` побольше (где-то 2500-3000)

```
Preferences → Output Plugin → Configure → Buffering → Buffer size(ms)
```

или то же самое, но для локализованной версии

```
Настройки → ВАШ плагин вывода звука → Настройка → Дополнительные  
настройки → Размер буфера (милисек)
```

- Старайтесь использовать качественные звуковые карты, которые имеют более полную обработку звука на уровне железа

Тюнинг настроек программ из семейства Mozilla

Ссылка на оригинал: [http://ru.gentoo-wiki.com/Тюнинг настроек программ из семейства Mozilla](http://ru.gentoo-wiki.com/Тюнинг_настроек_программ_из_семейства_Mozilla)

С версии: 1.5

Navigator(Seamonkey/Firefox)

Браузеры семейства Mozilla имеют большое количество настроек, все из которых легко доступны, если набрать в строке адреса вместо URL строку 'about:config'. Следующие пять настроек позволят ускорить загрузку страниц:

1. network.http.pipelining
2. network.http.pipelining.firstrequest
3. network.http.pipelining.maxrequests
4. network.http.proxy.pipelining
5. nglayout.initialpaint.delay

Настройка pipelining позволит браузеру делать одновременно несколько запросов к серверу. По-умолчанию используется одно соединение. Соответственно, значения `network.http.pipelining`, `network.http.pipelining.firstrequest` и `network.http.proxy.pipelining` следует выставить в **true**. Параметр `network.http.pipelining.maxrequests` отвечает за максимальное количество соединений. Но не стоит его делать слишком большим, т.к. это может привести к ошибкам при загрузке, рекомендуется выставить его в районе **12**. И, наконец, `nglayout.initialpaint.delay` это визуальное ускорение, то есть задержка между тем как браузер начал получать ответ от сервера и началом отображения в окне браузера. Стоит поставить здесь **0** т.к. это позволит вам получить доступ к уже загруженной части страницы.

Примечание: вторая и пятая настройки по пока невыясненным причинам могут отсутствовать в вашем браузере. Чтобы их задействовать, можно добавить их вручную (правая кнопка мыши→New→тип значения).

`browser.turbo.enabled`

И выставляем ему значение 'true'

Далее ищем

1. network.http.max-connections-per-server
2. network.http.max-persistent-connections-per-proxy
3. network.http.max-persistent-connections-per-server

Выставляем всем этим параметрам значение '8'

Управление Кешем в Firefox и Seamonkey

С версий Firefox 1.5.x и Seamonkey 1.0.x стала доступна функция быстрого перехода по открытым ранее страницам за счет хранения отрендеренной страницы в оперативной памяти. Не всем это понравилось т.к. браузер стал потреблять больше оперативной памяти и много и в качестве лекарства предлагали

отключить эту функцию, установив в about:config опцию

- browser.cache.memory.enable

соответственно в false. В то время как можно тонко оттюнинговать отведенный для этой функции размер памяти с помощью опции

- browser.cache.memory.capacity

Которая по умолчанию отсутствует, соответственно тип имеет - integer, а значение задается в килобайтах. Проверить изменения, а также просто просматривать статистику кешей (memory и disk) можно открыв about:cache до и после.

Управление логотипами сайтов

Есть две опции для включения/выключения логотипов сайтов в табках и адресной строке:

- browser.chrome.favicons boolean
- browser.chrome.site_icons boolean

Как я понял это два стандарта (если не прав, поправьте[favicon.ico обычно лежит в корне сайта, как обстоит с site_icon не знаю]). Например gentoo.ru использует favicon, а на странице mult.ru (непосредственно на той на которой загружается выбранный мульт, на главной странице никакого лого не показывается) показывалось лого при выключенном favicons и включенном site_icons.

Для перешедших с Firefox на Seamonkey.

Не знаю как вам а мне в Firefox нравится функция сохраняющая и соответственно показывающая эти самые лого при просмотре закладок. И перейдя с недавних пор на Seamonkey мне очень ее не доставало, ибо зрительно с ней гораздо удобней отыскивать нужную закладку, но по умолчанию эта функция там отключена. А управляется она оказывается опцией

- browser.chrome.load_toolbar_icons integer
 - 0 - соответственно отключена
 - 1 - включена (но сохраняются только при следующем после включения, посещении сайта)
 - 2 - включаются иконки для всех сайтов для которых это возможно (толком не понял, толи проверяет все сайты на их наличие [и тогда не понятно при каждом запуске или... ?], толи показывает сохраненные ранее т.к. закладки я скопировал из Firefox)

Thumbnail'ы просматриваемых картинок в табках

Итак с недавних пор появилась такая небольшая функция как thumbnail просматриваемой картинки на табе и в адресной строке, но к сожалению у некоторых эта малозначимая функция вызывает тормоза интерфейса, но не все знают как ее отключить не отключая favicon(логотип сайта). Делается это с помощью опции

- browser.chrome.image_icons.max_size

для которой можно задать размер картинки, по умолчанию ее значение **1024** (килобайт или разрешение?), соответственно чтобы отключить ее, нужно установить значение **0**.

Тонкая настройка IDE дисков с помощью *hdparm*

Ссылка на оригинал: http://www.opennet.ru/base/sys/htparm_tune.txt.html

Перенесено в: 1.5

Введение

К сожалению настройки по-умолчанию для IDE жёстких дисков сделаны с уклоном в надёжность работы, в том числе на некачественном оборудовании. На большинстве же современных материнских плат и жёстких дисков можно заметно увеличить производительность IDE подсистемы, причём это не потребует много времени.

Характеристики IDE устройства

Для получения характеристик IDE устройства выполните команду:

hdparm -i /dev/hda

Вместо /dev/hda можно указать любое другое IDE устройство

Расшифровка

Max Mult Sect?: Это поле содержит максимальное число секторов, которое ваш жесткий диск может прочитать за «один присест».

Mult Sect?: Текущая настройка количества секторов, считываемых за один раз.

PIO modes и DMA modes: Здесь перечисляются режимы, которые поддерживает жесткий диск. Режим, помеченный символом «звездочка» (*), является текущим.

Advanced PM?: Признак 'yes' указывает на то, что жесткий диск поддерживает APM (Advanced Power Management – Расширенное Управление Питанием).

Запустив другую команду, вы получите дополнительную информацию:

darkstar:\$ hdparm /dev/hda

Результат работы команды:

```
/dev/hda:
multcount = 0 (on)
I / O? support = 0 (16-bit)
unmaskirq = 0 (off)
using_dma = 0 (off)
keepsettings = 0 (off)
nowerr = 0 (off)
readonly = 0 (off)
readahead = 8 (on)
geometry = 2482/255/63, sectors = 39876480, start = 0
```

Коротко...

- **multcount** – число секторов, считываемых одновременно;
- **I / O? support** – режим обмена с жестким диском (16/32/32sync);
- **using_dma** – указывает, используется ли в настоящий момент режим DMA или нет;

- **keepsettings** — указывает, сохраняются ли настройки после программного сброса (изменять эту настройку не рекомендуется, если вы не уверены в том, что делаете);
- **readonly** — обычно установлен в 1 только для CD-ROM, этот флаг сообщает системе является ли устройство read-only или нет;
- **readahead** — количество секторов для опережающего чтения;

Настройка

ПРЕДУПРЕЖДЕНИЕ! Неправильная настройка может привести к потере данных на жёстком диске, а возможно и самого дика.

I / O? Support

-c0 — установка 16-и битного режима (по-умолчанию)\\
-c1 — установка 32-х битного режима (обычно то, что вам нужно)\\
-c3 — установка 32-х битного синхронного режима (мне не известно, когда именно это надо)

Multicount

Количество секторов передаваемых сразу (одним блоком). Степень двойки, максимальное число — параметр Max Mult Sect? в выводе hdparm -i. Чаще всего вам именно его и нужно указывать.

Включение DMA

Большинство современных чипсетов и HDD поддерживают DMA под линуксом. Для некоторых самых новых чипсетов в данный момент поддержки DMA может не быть, однако это не так часто бывает. Обычно в течении нескольких недель после выхода нового чипсета появляется его поддержка в последнем ядре, так что если DMA у вас не включается, то следует, скорее всего, обновить ядро.

-d0 — запрещение DMA\\
-d1 — включение DMA

DMA mode и PIO mode

Самая опасная настройка — её неправильное использование чаще всего приводит к повреждению данных или оборудования.

-X<число> — установка режима PIO или DMA

Режимы multiword DMA:

32 (базовое число) + номер режима:

-X32 — mdma0
 -X33 — mdma1
 -X34 — mdma2

Для PIO базовое число 8, для Ultra DMA? 64.

-X64 — udma0
 -X65 — udma1
 -X66 — udma2
 -X67 — udma3
 -X68 — udma4
 -X69 — udma5

Readahead (опережающее чтение)

На сколько секторов вперёд читать при запросе. Ускоряет производительность при чтении файлов большого размера, однако замедляет работу с файлами небольшого размера.

Лучше всего установить то же значение, что и для **multicount**.

-aN — опережающее чтение N секторов

Тестирование

Для тестирования наберите:

hdparm -t <имя устройства>

Желательно это делать, когда заметной дисковой активности нет.

Потом попробуйте какой-нибудь вариант конфигурации, например:

hdparm -u1c1d1m8a8 <имя устройства>

После чего опять повторите:

hdparm -t <имя устройства>

И сравните результаты.

По результату этого теста не стоит подбирать опции -a и -t, для этого лучшим тестом будут реальные приложения, исполняемые на вашей машине. Если это домашняя машина, а не боевой сервер, на котором важна сверхтонкая подстройка (но на таких серверах редко оказываются IDE-диски, о которых здесь идёт речь), то тогда параметр multicount (-m) ставите на максимум, а параметр read-ahead (-a) равным multicount.

Ускорение загрузки системы

Ссылка на оригинал: [http://ru.gentoo-wiki.com/Ускорение загрузки системы](http://ru.gentoo-wiki.com/Ускорение_загрузки_системы)

Перенесено в: 1.5

Введение

Одно из достоинств Gentoo Linux - это возможность настроить и оптимизировать буквально ВСЮ систему, в том числе и её загрузку. Данное направление оптимизации практически неисчерпаемо. При желании (но и при большом риске убить систему) загрузку можно довести и до нескольких секунд. Здесь же мы рассмотрим некоторые наиболее простые и безопасные методы ускорения загрузки.

Параллельный запуск

Первое, что может дать заметный прирост скорости загрузки, это распараллеливание запуска системных сервисов. В файле `/etc/conf.d/rc` замените NO на YES в переменной RC_PARALLEL_STARTUP. Файл: `/etc/conf.d/rc`

...

```
RC_PARALLEL_STARTUP="YES"
```

...

Если у вас нет такой переменной, обновите пакет `baselayout emerge -u baselayout`

Примечание: Если эта настройка создаст проблемы при загрузке (некоторые сервисы могут пытаться загрузиться не тогда, когда это нужно), то верните значение NO.

Значительно был улучшен механизм параллельного запуска в новых (нестабильных) версиях `baselayout` `echo "sys-apps/baselayout ~x86" >> /etc/portage/package.keywords emerge sync && emerge sys-apps/baselayout -u`

Для совсем реактивного запуска используйте **initng**
emerge sys-apps/initng

Локальная сеть

Если вы часто работаете без подключения к локальной сети, то наверняка заметили, как много времени уходит на определение настроек сети (точнее, на определение того, что настраивать нечего). В этом случае поможет демон `ifplugd`,

который определяет, подключен ли сетевой кабель (точнее, активна ли сеть).
Итак:

emerge -n ifplugd

теперь добавляем ifplugd в список запускаемых сервисов:

rc-update add ifplugd boot

Обновление зависимостей и переменных окружения

Каждый раз при загрузке Gentoo обновляет зависимости модулей и переменные окружения, разумеется, что это имеет смысл лишь тогда, когда вы действительно внесли изменения в вашу систему. Чтобы это осуществить, сделаем следующие изменения.

Меняем:

Файл: /etc/init.d/modules - оригинал

```
...
ebegin "Calculating module dependencies"
    /sbin/modules-update &>/dev/null
    eend $? "Failed to calculate dependencies"
...
```

Файл: /etc/init.d/modules - замена

```
...
if [ /etc/modules.d -nt /etc/modules.conf ]
then
    ebegin "Calculating module dependencies"
    /sbin/modules-update &>/dev/null
    eend $? "Failed to calculate dependencies"
else
    einfo "Module dependencies are up-to-date"
fi
...
```

Меняем:

Файл: /etc/init.d/bootmisc - оригинал

```
...
if [ -x /sbin/env-update.sh ]
then
    ebegin "Updating environment"
    /sbin/env-update.sh >/dev/null
    eend 0
fi
...
```

Файл: /etc/init.d/bootmisc - замена

```
...
if [ -x /sbin/env-update.sh ]
then
    if [ /etc/env.d -nt /etc/profile.env ]
    then
        ebegin "Updating environment"
        /sbin/env-update.sh >/dev/null
        eend 0
    else
        einfo "Environment up-to-date"
    fi
fi
...
```

Примечание: при использовании данного трюка следует быть внимательным при обновлении baselayout, т.к. обновление каждый раз будет пытаться вернуть изменения, сделанные вами.

Монтирование локальных файловых систем

Можно подправить сервис **localmount**, чтобы он монтировал локальные файловые системы не одну за другой, а все сразу.

Замените:

Файл: /etc/init.d/localmount - оригинал

```
...
mount -at ... >/dev/null
```

Файл: /etc/init.d/localmount - замена

```
...
mount -aFt ... >/dev/null
...
```


Что делать, если система «зависла»

Ссылка на оригинал: http://perlovka.homelinux.net/?node=gentoo_articles&id=4

Автор: Perlovka

С версии: 1.5

«Подвесить» Линукс вполне реально, несмотря на стабильность системы. «Подвесить» взято в кавычки потому, что при кажущихся признаках капитального останова системы ядро продолжает работать.

Рассмотрим некоторые ситуации, в которые может попасть пользователь:

1. «Завис» X сервер

При зависании или неадекватном поведении X сервера, можно воспользоваться командой перезапуска сервера — **Ctrl+Alt+Backspace**. При этом, если сервер был запущен командой *startx*, вы вернетесь в приглашение консоли; если же он был запущен с помощью менеджера дисплея *xdm* (*gdm*, *kdm*), то вы окажетесь на экране вашего графического менеджера загрузки.

2. Приложение не воспринимает команды

Если система перестала отвечать на команды клавиатуры, и **Ctrl+Alt+Backspace** не помогает, приходится использовать более жесткие методы воздействия. Один из этих методов заключается в использовании низкоуровневых сигналов ядра. Они реализуются с помощью клавиши **SysRq** (*System Request*), также известной как **PrintScreen**.

Для использования этих сигналов вам может понадобиться пересобрать ядро с их поддержкой:

```
Kernel hacking --->
[*] Magic SysRq key
```

Также можно включить эту опцию «на лету», но работать это будет до первой перезагрузки компьютера:

```
echo "1" > /proc/sys/kernel/sysrq
```

Итак, вы можете использовать следующие комбинации клавиш в аварийной ситуации:

Alt+SysRq+R — изменение режима клавиатуры с Raw в XLAT. Если приложение зависло, и заблокировало клавиатуру, не оставляя вам возможности перейти в консоль простым нажатием **Ctrl+Alt+F1**, воспользуйтесь этой комбинацией. Теперь вы сможете перейти в консоль и прервать выполнение приложения.

Alt+SysRq+K — эта комбинация носит название *Secure Access Key* (SAK). Она убивает все запущенные в текущей виртуальной консоли процессы. Исторически SAK предназначался для других целей. Применять его советовали пользователям перед тем как залогиниться в систему. Дело в том, что довольно легко написать программу или скрипт, эмулирующие логин. Неискушенный пользователь вводит свой логин и пароль, а «троянский конь» сохраняет их в надежном месте, чтобы злоумышленник воспользовался плодами вашей доверчивости. Если же юзер

перед тем как зайти в систему нажмет **SAK**, то прибьет тройна и получит нормальное, «оригинальное» приглашение системы.

Alt+SysRq+E — посылает сигнал SIGTERM всем процессам, кроме `init`. Сигнал SIGTERM означает, что система завершает некий процесс и прибирает за ним мусор — закрывает открытые процессом файлы, убивает временные и так далее;

Alt+SysRq+I — посылает сигнал SIGKILL всем процессам, кроме `init`. В отличие от предыдущего сигнала, SIGKILL грубо прерывает выполнение программы, не выполняя после этого «зачистку».

Alt+SysRq+L — посылает сигнал SIGKILL всем процессам, включая `init`. В этом случае система уходит в полный даун. Эту команду лучше не трогать.

Alt+SysRq+S — запускает аварийную синхронизацию (запись кэша) всех примонтированных дисков. Позволяет избежать потери данных.

Alt+SysRq+U — Перемонтировать все примонтированные файловые системы в режиме «только для чтения». Если операция прошла успешно, `fsck` не будет проверять файловые системы после «жесткого» перезапуска системы.

Alt+SysRq+B — перезагрузка системы. Очень быстрая, потому что этапы синхронизации и размонтирования пропускаются. Употреблять только в самых крайних случаях. С таким же успехом можете нажать на `Reset`.

Alt+SysRq+O — действует на систему подобно выстрелу в висок. Никакой синхронизации буфера, размонтирования и прочего. Сразу гасится свет, то бишь вырубается питание. Поэтому до нажатия этих клавиш следует синхронизировать буфер (**Alt+SysRq+S**), затем перемонтировать файловые системы в режиме «только для чтения» (**Alt+SysRq+U**) и только потом — **Alt+SysRq+O**. И запомните — сначала синхронизация, потом размонтирование, однако не наоборот.

Русское WIKI

Ссылка на оригинал: <http://ru.gentoo-wiki.com>

С версии: 1.0

Обновлено: 1.5

HOWTO hal и устройства ввода

Дата: 28.01.2008

hal-0.5.10

В связи с выходом hal-0.5.10 и включении поддержки hal в xorg возникло много вопросов: где раскладки, где мышка, где тачпад, и прочее.

Сейчас xorg использует hal для устройств ввода и xorg.conf игнорирует
Данное руководство предназначено для безболезненного перехода на новую систему

В данном руководстве используются пакеты из ~x86(~amd64) ветки. Посему заботу о keywords оставляю на вашей совести.

Обновляем portage

```
# emerge --sync
```

```
(# eix-sync -v)
```

Обновляем установку xorg

```
#emerge -avDNt xorg-x11
```

убеждаемся в наличии флага hal. При отсутствии — включаем. / В окончании сборки пакета xorg-server выведется список пакетов для пересборки, их надо пересобрать. Если не заметили или пропустили вот команда для получения списка пакетов:

```
emerge portage-utils; qlist -I -C x11-drivers/
```

Клавиатура

Правила(rules) пользователя можно разместить в каталогах:

```
/usr/share/hal/fdi/policy/20thirdparty/  
или аналогичном /etc/hal/fdi/policy
```

получить список устройств "узнанных" hal можно командой lshal у меня системе(ноутбук с мультимедийными клавишами) оказалось несколько устройств которые совместимы с понятием "клавиатура", но основная была описана так:

```
udi =  
'/org/freedesktop/Hal/devices/platform_i8042_i8042_KBD_port_logicaldev_input'  
info.addons.singleton = {'hald-addon-input'} (string list)
```

```

info.capabilities = {'input', 'input.keyboard', 'input.keypad',
'input.keys', 'button'} (string list)
info.category = 'input' (string)
info.parent = '/org/freedesktop/Hal/devices/platform_i8042_i8042_KBD_port'
(string)
info.product = 'AT Translated Set 2 keyboard' (string)
info.udi =
'/org/freedesktop/Hal/devices/platform_i8042_i8042_KBD_port_logicaldev_input'
(string)
input.device = '/dev/input/event2' (string)
input.originating_device =
'/org/freedesktop/Hal/devices/platform_i8042_i8042_KBD_port' (string)
input.physical_device =
'/org/freedesktop/Hal/devices/platform_i8042_i8042_KBD_port' (string)
input.product = 'AT Translated Set 2 keyboard' (string)
input.x11_driver = 'evdev' (string)
input.xkb.layout = 'us' (string)
input.xkb.model = 'evdev' (string)
input.xkb.rules = 'base' (string)
input.xkb.variant = (string)
linux.device_file = '/dev/input/event2' (string)
linux.hotplug_type = 2 (0x2) (int)
linux.subsystem = 'input' (string)
linux.sysfs_path = '/sys/devices/platform/i8042/serio0/input/input2/event2'
(string)

```

Заметно что

```

input.xkb.layout = 'us'
input.xkb.rules = 'base'
input.xkb.variant =

```

нас не устраивает

Создаём новый файл

```
nano /usr/share/hal/fdi/policy/20thirdparty/10russian-layout.fdi
```

со следующим содержимым:

Файл: /usr/share/hal/fdi/policy/20thirdparty/10russian-layout.fdi

```

<?xml version="1.0" encoding="ISO-8859-1"?><!-- -*- SGML -*- -->
<match key="input.product" contains="AT Translated Set 2 keyboard">
<merge key="input.xkb.layout" type="string">us,ru</merge>
<merge key="input.xkb.variant" type="string">,winkeys</merge>
<merge key="input.xkb.options"
type="string">grp:alt_shift_toggle,grp_led:scroll</merge>
</match>

```

если у вас одно устройство ввода типа "клавиатура", то можно сточку

```
<match key="input.product" contains="AT Translated Set 2 keyboard">
```

заменить на

```
<match key="info.capabilities" contains="input.keyboard">
```

Так мы получим более универсальное правило. Можно заметить что названия опций перекликается с аналогами xorg.conf. Так и есть, в принципе здесь можно

записать все те параметры которые вы использовали в xorg.conf только в приложении к XML.

ВАЖНО: Если у вас остались проблемы с расположением клавиш, то возможно так и следует сделать?

например:

```
Option "XkbRules" "xorg"
```

тоже самое что

```
<merge key="input.xkb.rules" type="string">xorg</merge>
```

```
Driver "kbd"
```

=

```
<merge key="input.x11_driver" type="string">kbd</merge>
```

Сохраняем файл.

```
#!/etc/init.d/hald restart
```

Проверяем системный лог и убеждаемся, что правило успешно добавлено.

```
#lshal
```

должно вывести следующие строчки:

```
...
input.x11_driver = 'evdev'    (string)
input.xkb.layout = 'us,ru'   (string)
input.xkb.model = 'evdev'    (string)
input.xkb.options = 'grp:alt_shift_toggle,grp_led:scroll' (string)
input.xkb.rules = 'base'     (string)
input.xkb.variant = ',winkeys' (string)
...
```

Если так и есть, то дело сделано. Если нет--внимательно перечитываем до просветления.

Перезапускаем X-ы

После чего можно перезапустить X-ы(перезагрузить компьютер, перезапустить xdm). Проверяем. Радует. Если не получается, то внимательно читаем лог Xorg, Скорее всего там можно найти ответ.

Мышка

Пока проблема одна. Hal по умолчанию использует драйвер evdev, которые не всегда корректно работает в полноэкранных приложениях(игры)

```
#echo "SDL_VIDEO_X11_DGAMOUSE=0" >>/etc/env.d/02locale
#env-update
#source /etc/profile
```

Должно решить эту проблему.

Touchpad

Пока в разработке.

HOWTO Ivman

Цель

Целью статьи является установка [ivman](#) для автомонтирования устройств.

Внимание: часть информации может быть недействительна для старых версий Ivman, особенно Ivman 0.5.x. Пожалуйста, используйте последние стабильные версии Ivman.

Требуемые программы

- [UDEV](#)
- HAL
- D-BUS
- pmount
- [ivman](#)

Все требуемые пакеты есть в портежах, поэтому минимальным действием будет:

```
emerge -av ivman
```

Размаскируйте требуемые пакеты, если необходимо.

Если установка HAL выдаёт предупреждения, что не установлены "u64" или "BLKGETSIZE64" и вы используете ядро 2.6, обновите пакет linux-headers:

```
emerge --oneshot linux-headers
```

Когда linux-headers полностью обновится, пересоберите glibc:

```
emerge --oneshot glibc
```

Использование

Есть два пути использования Ivman: от root, или от обычного пользователя. У любого из методов имеются как преимущества, так и недостатки.

Запуск Ivman от root

- Для запуска Ivman выполните: `/etc/init.d/ivman start`
- Чтобы Ivman стартовал при каждом запуске выполните: `rc-update add ivman default`
- Все действия Ivman после запуска выполняются с привилегиями пользователя «ivman» и группы «plugdev».
- По умолчанию, сменные устройства будут монтироваться для доступа на чтение/запись только входящим в группу «plugdev» (группа пользователей,

которым разрешено использовать `pmount`). Это более безопасно, чем запуск `lvman` от обычного пользователя.

- Размонтирование дисков обычным пользователем может быть затруднено. Вам может потребоваться использование `sudo pmount` или `sudo umount`. Размонтирование используя `'media:/' ioslave` в KDE может не работать.
- Для легкого размонтирования дисков обычными пользователями отредактируйте файл `/usr/share/hal/fdi/policy/10osvendor/10-storage-policy.fdi`, изменив `user` на `users` в строке `<merge key="storage.policy.default.mount_option.user" type="bool">true</merge>`. Если у вас нет такого файла, поищите похожий в `/usr/share/hal`. **(Внимание: это работает для `lvman 0.5.x`, но не для последних версий `lvman`.)**

Запуск `lvman` от обычного пользователя

- Для начала, у вас должен быть запущен HAL. Выполните как root: `/etc/init.d/hald start`
- Для запуска HAL каждый раз при загрузке выполните: `rc-update add hald default`
- Для запуска `lvman` единично выполните команду `lvman` под вашим пользовательским аккаунтом. Чтобы `lvman` запускался автоматически, когда вы входите в систему, вы должны сделать следующее:
 - KDE: создать символическую ссылку для `lvman` в `~/.kde/Autostart`: `ln -s $ (which lvman) ~/.kde/Autostart/lvman`
 - X: поместите `lvman-launch --exit-with-session xinit &` в `~/.xinitrc` (требуется `lvman 0.6.9` или более новый).
 - E17: [Создание `lvman.eap`](#) (секция 'приложения GTK+'), и добавьте это в `~/.e/e/applications/all/`. После отредактируйте `~/.e/e/applications/startup/.order`, добавив `lvman.eap`.
 - fluxbox: поместите `lvman-launch --exit-with-session xinit &` в `~/.fluxbox/startup` (требуется `lvman 0.6.9` или более новый)
 - Другие: **пожалуйста, добавьте ваши инструкции здесь!**
- `lvman` запустится под вашим пользовательским аккаунтом.

Примечание: Вы можете не находиться в группе `plugdev`, потому что сейчас устройства монтируются так, что только вы имеете к ним доступ.

- По умолчанию сменные устройства будут монтироваться на запись/чтение для вас и не допускать больше никого.
- Размонтирование работает как обычно. Работает стандартный `pmount`. Размонтирование используя `'media:/' ioslave` в KDE работает.
- Когда возможно (а, как правило, это возможно), запускайте `lvman` одновременно от root и от обычного пользователя. В этом случае автоматическое монтирование будет произведено от пользовательского запуска `lvman`, а если пользовательский `lvman` отсутствует, автоматическое монтирование будет передано запуску `lvman` от root. Не запускайте более одного пользовательского `lvman`, даже под различающимися пользовательскими

аккаунтами – это приведёт к войне за ресурсы.

Ivman не требует дополнительной конфигурации для автомонтирования. Данные fstab игнорируются, сменные устройства/диски будут смонтированы в /media. Однако, Ivman может быть использован не только для монтирования. Обретите свободу для добавления ваших супер-правил на этой странице :-)

Примеры некоторых правил

Code: Смонтировать всё, что есть

```
<ivm:Match name="ivm.mountable" value="true">
  <ivm:Option name="mount" value="true" />
</ivm:Match>
```

Code: Открыть мой MP3-плеер в тс, когда он подключен

```
<ivm:Match name="hal.info.product" value="IAUDIO">
  <ivm:Option name="exec" value="xterm -e mc /home/share/music
/media/IAUDIO" />
</ivm:Match>
```

Code: Открыть мою камеру в тс, когда она подключена

```
<ivm:Match name="hal.info.vendor" value="FUJIFILM">
  <ivm:Option name="exec" value="xterm -e mc /home/share/pics
/media/usbdisk/DCIM/100_FUJI" />
</ivm:Match>
```

Для KDE

Code: Выводит всплывающее окошко, когда что-нибудь подключено

```
<ivm:Match name="hal.info.category" value="storage">
  <ivm:Match name="hal.storage.bus" value="usb">
    <ivm:Option name="exec" value="kdialog --passivepopup 'USB storage
device detected: $hal.info.vendor$ $hal.info.product$' 4" />
  </ivm:Match>
</ivm:Match>
```

Проблемы

Использование записей в fstab (ivman 0.6.x или более новый)

Ivman 0.6.x или более новый использует rmount для монтирования устройств, и не нуждается в записях fstab. Но если хочется, то можно.

rmount (и Ivman, таким образом) будет автоматически учитывать записи fstab. С версии 0.6.0, на каждую символьную ссылку будет создано правило, без использования [IvmConfigMappings.xml](#). Устройства, не отмеченные в fstab, будут монтироваться в /media.

Использование записей в fstab (ТОЛЬКО ivman 0.5.x!)

Ivman 0.5.x может использовать ваш [/etc/fstab](#). Если вы до этого пользовались supermount, вам необходимо изменить записи для использования ivman. Например:

```
/dev/cdroms/cdrom0      /mnt/cdrom      iso9660      noauto,ro      0 0
```


Может также потребоваться отключение devfs.

Монтирование устройств с размонтированием от обычного пользователя (ТОЛЬКО ivman 0.5.x!)

Когда вы подключаете USB-носитель, ivman, запущенный от root монтирует его автоматически, но вы не можете размонтировать его иначе, чем от root. Это можно устранить, настроив HAL на монтирование всех устройств хранения данных с опцией "users", чего вам может очень не хотеться.

Создайте файл с названием **whatever.fdi** в директории **/usr/share/hal/fdi/95userpolicy/** со следующим содержимым:

Файл: whatever.fdi

```
<?xml version="1.0" encoding="ISO-8859-1"?> <!-- -*- SGML -*- -->
<deviceinfo version="0.2">
  <!-- Append users to default mount options -->
  <device>
    <merge key="storage.policy.default.mount_option.users" type="bool">true</merge>
  </device>
</deviceinfo>
```

Для дополнительной информации смотрите [HAL Specifications](#).

Решение проблемы неразмонтирования флешки после ее физического удаления

Иногда может возникать ситуация что после удаления флешки команда mount выдает что она подмонтирована, в результате следующие подсоединение флешки приводит к тому что ей присваивается следующая буква и в результате чего возникает множество мертвых точек монтирования

Code: Пример ситуации когда два раза вынута и вставлена одна и таже флешка

```
#mount
/dev/sdb1          on          /media/sdb1      type             vfat
(rw,noexec,nosuid,nodev,quiet,shortname=mixed,uid=104,gid=412,umask=007,fmask=0117,dmask=0007,iocharset=utf8)
/dev/sdc1          on          /media/sdc1      type             vfat
(rw,noexec,nosuid,nodev,quiet,shortname=mixed,uid=104,gid=412,umask=007,fmask=0117,dmask=0007,iocharset=utf8)
```

это происходит из за того что по умолчанию ivman для размонтирования вызывает команду `rmount /dev/sdb1`, которая нечего не выполняет выдавая что `/dev/sdb1` нет (udev успевает удалить это устройство)

решение проблемы следующее правим `/etc/ivman/lvmConfigBase.xml` таким образом

Файл: /etc/ivman/lvmConfigBase.xml

```
<!-- mount command. default is autodetected. Must be specified with
      umountcommand. -->
  <ivm:Option name="mountcommand" value="/usr/bin/pmount -u 007
'$hal.block.device$' " />

  <!-- umount command. default is autodetected. Must be specified with
```

```
mountcommand. -->
<ivm:Option name="umountcommand" value="/usr/bin/sudo /bin/umount -l
'$hal.block.device$' " />
```

а так же добавляя в /etc/sudoers строчку

Файл: /etc/sudoers

```
ivman ALL = NOPASSWD: /bin/umount
```

sudo конечно же должен быть установлен.

Разное

HAL может потребовать добавление coldplug на уровень загрузки boot для работы распознавания USB-устройств.

Если у вас проблемы с пользовательскими запусками ivman для автомонтирования USB flash, вам может потребоваться включить поддержку utf-8 в вашем ядре. Смотрите это [обсуждение на форуме](#) для подробностей.

Другие проблемы

Если ivman не работает, для начала остановите его:

```
/etc/init.d/ivman stop
```

Отредактируйте базовые настройки:

```
nano -w /etc/ivman/lvmConfigBase.xml
```

Отключите fork, и включите отладку. Потом запустите ivman из консоли. Смотрите сообщения.

Источник — «http://ru.gentoo-wiki.com/HOWTO_ivman»

Настройка дисковой подсистемы

Всеобщий ликбез

Под IDE понимаются устройства, подключаемые к IDE-интерфейсу. Обычно это жесткие диски и дисководы CD-ROM. Эти устройства должны быть документированы, как:

- IDE
- ATA
- ATAPI
- Enhanced IDE (EIDE)
- Fast ATA или Fast ATA-2

IDE устройства могут передавать данные по шине в двух режимах - PIO и DMA. Учтите, что это именно передача данных по шине, с пластины данные читаются медленнее, а вот из аппаратного кеша на полной скорости.

PIO - Программный ввод/вывод, метод передачи данных между двумя устройствами, использующий процессор как часть маршрута данных (процессор выполняет команду чтения порта, считывает байт или слово данных в свой регистр, после чего переписывает его в память, затем повторяет эту процедуру до тех пор, пока вся необходимая информация не будет считана из устройства в память).

Бывает PIO Mode 0 1 2 3 4. Чем больше номер режима, тем быстрее. IDE ZIP100 приводы от Iomega например умеют только PIO mode 0. Старые CD-ROM обычно умеют PIO mode 4, если не могут DMA.

- PIO Mode 0 = 3.3 Mb/s
- PIO Mode 1 = 5.2 Mb/s
- PIO Mode 2 = 8.3 Mb/s
- PIO Mode 4 = 11.1 Mb/s
- PIO Mode 5 = 16.7 Mb/s

DMA - Direct Memory Access - прямой доступ к памяти - собирательное название протоколов, позволяющих периферийному устройству передавать информацию непосредственно в системную память без участия центрального процессора, жесткие диски используют эту возможность в сочетании с возможностью перехватывать управление шиной и самостоятельно управлять передачей информации (bus mastering), что уменьшает нагрузку на процессор и повышает скорость передачи данных.

DMA встречается двух типов: UDMA и MDMA.

UDMA - ultra DMA - наиболее предпочитаемый тип, основной на сегодня. Бывает UDMA 0 1 2 3 4 5 6. Реально встречаются:

- UDMA 2 = 33 mb/s
- UDMA 4 = 66 mb/s
- UDMA 5 = 100 mb/s
- UDMA 6 = 133 mb/s

Последний встречается на не Intel матерях и не все винты его умеют. Кстати SATA винчестеры используют UDMA = 150 Mb/s.

Для UDMA 66 - 100 - 133 необходим 80-жильный шлейф, кроме того старые матери умеют его не на всех каналах, смотрите в инструкцию. Форсирование этих режимов с 40-жильным шлейфом может убить Ваши данные.

MDMA - multiword dma, более древний режим, предпочтителен для старых винчестеров и CD-ROM.

- MDMA0 = 4.2 mb/s
- MDMA1 = 13.3 mb/s

- MDMA2 = 16.7 mb/s

На большинстве современных систем ядро автоматически определяет и настраивает IDE подсистему на максимальную производительность, если правильно его сконфигурировать. Но настроить что-нибудь всё равно можно.

ОЧЕНЬ ВАЖНО

Всё нижеописанное может убить Ваши данные, сломать Вам винчестер, спалить Вашу машину, удивить Вашу кошку и т.п. отмазы :)

Всё нижеописанное тестировалось, работало и работает на пяти машинах с ядром 2.6.9-gentoo-r4 и hdparm-5.7-r1, ACCEPT_KEYWORDS="~x86".

У Вас что-нибудь может не работать, работать не так.

ВАЖНО

Всё нижеописанное не относится к SATA дискам работающим через libata интерфейс (то есть видимым как sdX а не hdX). Поскольку sata интерфейс гораздо ближе к scsi чем к pata, разработчики вполне обоснованно используют scsi подсистему для работы с sata. Поэтому на текущий момент настройка sata винчестеров средствами hdparm невозможна, так как в libata не реализована специфичная для подобных програм функциональность.

В Сети есть патчик на ядро, добавляющий нужный функционал, но он **ОЧЕНЬ** сырой и с вероятностью в 100% убьёт вам раздел. Даже ссылку давать не буду, если Вы камикадзе, сами найдёте.

Для начала

Убедитесь, что Ваши IDE диски используют DMA интерфейс, и ядро правильно настроено.

Для этого сделайте :

dmesg | less

или загляните в логи.

Нас будут интересовать сообщения о настройке `ide` контроллера и дисков. Мой кусок `kern.log` выглядит так (с моими комментариями):

Nov 14 17:45:54 tsotimus kernel: ide: Assuming 33MHz system bus speed for PIO modes: override with idebus=xx

[illegible]

Можно при загрузке сказать ядру `idebus=66`, но это работает только для не использующих DMA винчестеров и сидиромов. Подробности в `/usr/src/linux/Documentation/ide.txt`

Nov 14 17:45:54 tsotimus kernel: ICH2: IDE controller at PCI slot 0000:00:1f.1

Nov 14 17:45:54 tsoptimus kernel: ICH2: chipset revision 2

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

Определили первый контроллер.

Nov 14 17:45:54 tsotimus kernel: ICH2: not 100%% native mode: will probe irqs later

^^

Это из-за того что интеловский контроллер использует разные прерывания для

каждого из каналов. Это нормально.

Nov 14 17:45:54 tsoptimus kernel: ide0: BM-DMA at 0xf000-0xf007, BIOS settings: hda:DMA, hdb:pio

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

контроллер работает в busmaster режиме. dma для первого диска включен в BIOS.

Nov 14 17:45:54 tsoptimus kernel: ide1: BM-DMA at 0xf008-0xf00f. BIOS settings: hdc:pio, hdd:DMA

```
Nov 14 17:45:54 tsotimus kernel: Probing IDE interface ide0...
Nov 14 17:45:54 tsotimus kernel: hda: ST340016A, ATA DISK drive
Nov 14 17:45:54 tsotimus kernel: ide0 at 0x1f0-0x1f7,0x3f6 on irq 14
Nov 14 17:45:54 tsotimus kernel: Probing IDE interface ide1...
Nov 14 17:45:54 tsotimus kernel: hdd: DV-516E, ATAPI CD/DVD-ROM drive
Nov 14 17:45:54 tsotimus kernel: ide1 at 0x170-0x177,0x376 on irq 15
Nov 14 17:45:54 tsotimus kernel: PDC20265: IDE controller at PCI slot 0000:02:0a.0
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Определили интегрированный promise контроллер
Nov 14 17:45:54 tsotimus kernel: ACPI: PCI interrupt 0000:02:0a.0[A] -> GSI 17 (level, low) -> IRQ 17
Nov 14 17:45:54 tsotimus kernel: PDC20265: chipset revision 2
Nov 14 17:45:54 tsotimus kernel: PDC20265: 100%% native mode on irq 17
Nov 14 17:45:54 tsotimus kernel: PDC20265: (U)DMA Burst Bit ENABLED Primary MASTER Mode Secondary MASTER Mode.
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BIOS этого уродца не включает DMA, поэтому его включило ядро :)
Nov 14 17:45:54 tsotimus kernel: ide2: BM-DMA at 0xac00-0xac07, BIOS settings: hde:pio, hdf:pio
AAAAAAA
Что мы собственно и наблюдаем.
Nov 14 17:45:54 tsotimus kernel: ide3: BM-DMA at 0xac08-0xac0f, BIOS settings: hdg:pio, hdh:pio
Nov 14 17:45:54 tsotimus kernel: Probing IDE interface ide2...
Nov 14 17:45:54 tsotimus kernel: hde: FUJITSU MPG3204AT E, ATA DISK drive
Nov 14 17:45:54 tsotimus kernel: ide2 at 0x9c00-0x9c07,0xa002 on irq 17
Nov 14 17:45:54 tsotimus kernel: Probing IDE interface ide3...
Nov 14 17:45:54 tsotimus kernel: hda: max request size: 128KiB
Nov 14 17:45:54 tsotimus kernel: hda: 78165360 sectors (40020 MB) w/2048KiB Cache, CHS=65535/16/63, UDMA(100)
AAAAAAA
Говорит само за себя
Nov 14 17:45:54 tsotimus kernel: hda: cache flushes not supported
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
А вот это странно :( Ядро не сможет принудительно синхронизировать аппаратный кеш винчестера, в случае panic или oops будет не сладко. До этого было supported, разберусь...
Nov 14 17:45:54 tsotimus kernel: /dev/ide/host0/target0/lun0: p1 p2 p3 p4
Nov 14 17:45:54 tsotimus kernel: hde: max request size: 128KiB
Nov 14 17:45:54 tsotimus kernel: hde: 40031712 sectors (20496 MB) w/512KiB Cache, CHS=39714/16/63, UDMA(100)
Nov 14 17:45:54 tsotimus kernel: hde: cache flushes not supported
Nov 14 17:45:54 tsotimus kernel: /dev/ide/host2/bus0/target0/lun0: p1 p2
Nov 14 17:45:54 tsotimus kernel: hdd: ATAPI 48X DVD-ROM drive, 256kB Cache, UDMA(33)
Nov 14 17:45:54 tsotimus kernel: Uniform CD-ROM driver Revision: 3.20
Проверьте настройки ядра. Мой кусок касающийся IDE выглядит так (с моими комментариями)
# ATA/ATAPI/MFM/RLL support
#
CONFIG_IDE=y
CONFIG_BLK_DEV_IDE=y

#
# Please see Documentation/ide.txt for help/info on IDE drives
#
# CONFIG_BLK_DEV_IDE_SATA is not set
# CONFIG_BLK_DEV_HD_IDE is not set
CONFIG_BLK_DEV_IDEDISK=y
#CONFIG_IDEDISK_MULTI_MODE is not set
CONFIG_BLK_DEV_IDECD=y
# CONFIG_BLK_DEV_IDETAPE is not set
# CONFIG_BLK_DEV_IDEFLOPPY is not set
# CONFIG_BLK_DEV_IDESCSI is not set
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Был нужен для пишущих cdrom, сейчас не нужен (даже вреден)
```

```
CONFIG_IDE_TASK_IOCTL=y
CONFIG_IDE_TASKFILE_IO=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Не разбирался :)

```
#
# IDE chipset support/bugfixes
#
# CONFIG_IDE_GENERIC is not set
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Не нужен если Ваши IDE контроллеры известны ядру. У 99% людей они известны. Посему выключен.

```
# CONFIG_BLK_DEV_CMD640 is not set
CONFIG_BLK_DEV_IDEPCI=y
CONFIG_IDEPCI_SHARE_IRQ=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Разрешает устройству совместно использовать одно прерывание с другими устройствами.

В случае проблем можно выключить.

```
# CONFIG_BLK_DEV_OFFBOARD is not set
# CONFIG_BLK_DEV_GENERIC is not set
# CONFIG_BLK_DEV_OPTI621 is not set
# CONFIG_BLK_DEV_RZ1000 is not set
CONFIG_BLK_DEV_IDEDMA_PCI=y
# CONFIG_BLK_DEV_IDEDMA_FORCED is not set
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

ОПАСНО!!!! Если ваш винт или контроллер совсем никак не умеет DMA или blacklisted, то есть риск потерять раздел.

```
CONFIG_IDEDMA_PCI_AUTO=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

А вот это значительно нежнее :) Если с винтом и контроллером всё в порядке, ядро само включит Вам DMA.

```
# CONFIG_IDEDMA_ONLYDISK is not set
Ежели ваш CDROM глючит с DMA или работает медленнее чем в PIO режиме, ядро включит DMA только для винчестеров.
# CONFIG_BLK_DEV_AEC62XX is not set
# CONFIG_BLK_DEV_ALI15X3 is not set
# CONFIG_BLK_DEV_AMD74XX is not set
# CONFIG_BLK_DEV_ATIIXP is not set
# CONFIG_BLK_DEV_CMD64X is not set
# CONFIG_BLK_DEV_TRIFLEX is not set
# CONFIG_BLK_DEV_CY82C693 is not set
# CONFIG_BLK_DEV_CS5520 is not set
# CONFIG_BLK_DEV_CS5530 is not set
# CONFIG_BLK_DEV_HPT34X is not set
# CONFIG_BLK_DEV_HPT366 is not set
# CONFIG_BLK_DEV_SC1200 is not set
CONFIG_BLK_DEV_PIIX=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Контроллер моей материнки

```
# CONFIG_BLK_DEV_NS87415 is not set
CONFIG_BLK_DEV_PDC202XX_OLD=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Второй контроллер моей материнки

```
CONFIG_PDC202XX_BURST=y
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Форсируем DMA для promise, так как его биос DMA не включает. Странный он.

```
# CONFIG_BLK_DEV_PDC202XX_NEW is not set
# CONFIG_BLK_DEV_SVWKS is not set
# CONFIG_BLK_DEV_SIIIMAGE is not set
# CONFIG_BLK_DEV_SIS5513 is not set
```

```
# CONFIG_BLK_DEV_SLC90E66 is not set
# CONFIG_BLK_DEV_TRM290 is not set
# CONFIG_BLK_DEV_VIA82CXXX is not set
# CONFIG_IDE_ARM is not set
CONFIG_BLK_DEV_IDEDMA=y
# CONFIG_IDEDMA_IVB is not set
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Форсирование UDMA 5 6 если ядро само его не включает.

ОПАСНО - если у Вас 40-жильный или повреждённый шлейф, вы можете убить данные.

```
CONFIG_IDEDMA_AUTO=y
```

```
# CONFIG_BLK_DEV_HD is not set
```

Ускоряемся :)

Сделайте, если ещё не сделали,

emerge hdparm

На одной консоли сделайте:

man hdparm

А на другой мы начнём играть :))

Делайте:

hdparm -tT /dev/hdX

Где X - буква вашего винчестера

Мы запустили тест скорости чтения из кеша и с пластины. Для усреднения результатов запустите его несколько раз.

Должны увидеть следующее:

```
/dev/hda:
Timing cached reads: 616 MB in 2.00 seconds = 307.74 MB/sec
Timing buffered disk reads: 74 MB in 3.04 seconds = 42.33 MB/sec
```

Первый результат с учётом программного кеширования, он у всех большой :)

Второй - собственно чтение с диска с учётом аппаратного кеширования.

Второй результат мы собственно и будем улучшать :) Цифры в районе 35-55 mb/s хороший результат, выше 50 без raid контроллера вы на параллельном интерфейсе вряд ли получите (хотя... кто знает...). Но в любом случае кроме скорости можно

Разбираемся с железом на железном уровне

Имеет смысл заглянуть в BIOS и в корпус.

ВНИМАНИЕ!!! Если вы страдаете хронической неуверенностью, криворукостью, боитесь потерять гарантию, пропустите этот раздел!

Иногда сборщики компьютеров страдают излишней жадностью или

криворукостью, поэтому всегда имеет смысл проверить, всё ли внутри корпуса как надо.

Каждый канал IDE/ATA интерфейса поддерживает подключение двух устройств - master и slave. Конфигурация обычно задается переключком на устройстве. Кроме этих двух позиций там обычно присутствует и третья - cable select. Для работы устройств в положении переключки cable select требуется специальный Y-образный шлейф, центральный разъем которого подключается к материнской плате. Крайние разъемы такого кабеля неравноправны - устройство, подключенное к одному разъему, автоматически становится master, к другому - slave.

Проверьте, чтобы шлейфы были 80-жильные, на 40-жильных вы не сильно ускоритесь :).

Каждый канал в каждый момент времени может обрабатывать только один запрос к одному устройству. Следующий запрос, пусть даже к другому устройству, будет ожидать завершения текущего. Разные каналы при этом могут работать независимо. Поэтому не стоит подключать два активно используемых устройства (например, два жестких диска), к одному каналу.

В идеале каждое IDE-устройство стоит подключать к отдельному каналу (в этом заключается основное преимущество SATA).

Практически все современные чипсеты поддерживают возможность использования различных режимов передачи данных для устройств, подключенных к одному каналу. Однако злоупотреблять этим все-таки не стоит. Два устройства, существенно различающихся по скорости, лучше все-таки разнести по разным каналам.

Не рекомендуется подключать к одному каналу жесткий диск и ATAPI-устройство (например, CD-ROM). Как было сказано выше, протокол ATAPI использует другую систему команд, и, кроме того, даже самые быстрые ATAPI-устройства намного медленнее жесткого диска, что может замедлить работу последнего.

Идеальная конфигурация.

Каждый винт и CD-ROM на отдельном шлейфе.

Приемлемая конфигурация.

На первом канале основной винт, на втором дополнительный master и CD-ROM slave

Тоже неплохо.

На первом канале два винчестера одного поколения, CD-ROM на втором.

Плохо!!!

На одном канале современный винт и древняя древность на полтора гига :)

Смотрим в BIOS. Включаем UDMA где только можно, включаем Bus master для контроллеров IDE, включаем IDE Block mode.

Разбираемся с железом на софтверном уровне

А что умеет наш винчестер? И что у него включено сейчас?

Делаем:

hdparm -ilv /dev/hdX | less

Получаем длинный листинг с описанием возможностей нашего винчестера. Пример моего с комментариями приведён ниже.

Эта секция коротко описывает, что у нас включено прямо сейчас.

```
/dev/hda:
multcount   = 16 (on)
IO_support  = 1 (32-bit)
unmaskirq   = 1 (on)
using_dma   = 1 (on)
keepsettings = 0 (off)
readonly    = 0 (off)
readahead   = 256 (on)
geometry    = 65535/16/63, sectors = 40020664320, start = 0
Эта секция показывает необработанные для читабельности данные, прочитанные напрямую с
винчестера.
Model=ST340016A, FwRev=3.19, SerialNo=3HS9R2GG
Config={ HardSect NotMFM HdSw>15uSec Fixed DTR>10Mbs RotSpdTol>.5% }
RawCHS=16383/16/63, TrkSize=0, SectSize=0, ECCbytes=4
BuffType=unknown, BuffSize=2048kB, MaxMultSect=16, MultSect=16
CurCHS=4047/16/255, CurSects=16511760, LBA=yes, LBAssects=78165360
IORDY=on/off, tPIO={min:240,w/IORDY:120}, tDMA={min:120,rec:120}
PIO modes: pio0 pio1 pio2 pio3 pio4
DMA modes: mdma0 mdma1 mdma2
UDMA modes: udma0 udma1 udma2 udma3 udma4 *udma5
AdvancedPM=no WriteCache=enabled
Drive conforms to: device does not report version:
```

* signifies the current active mode

А вот это более интересно. Читайте комментарии в ней самой.

ATA device, with non-removable media

```
Model Number:   ST340016A
Serial Number:  3HS9R2GG
Firmware Revision: 3.19
```

Standards:

```
Supported: 5 4 3 2
Likely used: 6
```

Configuration:

```
Logical      max    current
cylinders    16383  4047
heads        16     16
sectors/track 63     255
```

--

```
CHS current addressable sectors: 16511760
LBA  user addressable sectors: 78165360
device size with M = 1024*1024:  38166 MBytes
device size with M = 1000*1000:  40020 MBytes (40 GB)
```

Capabilities:

```
LBA, IORDY(can be disabled)
bytes avail on r/w long: 4   Queue depth: 1
Standby timer values: spec'd by Standard
R/W multiple sector transfer: Max = 16 Current = 16
```

[illegible]

Аппаратное блочное чтение. Может читать зараз 16 блоков. Так и делает.

Recommended acoustic management value: 128, current value: 254

[illegible]

Управление акустикой. Чем больше значение, тем шумнее и быстрее.

DMA: mdma0 mdma1 mdma2 udma0 udma1 udma2 udma3 udma4 *udma5

Cycle time: min=120ns recommended=120ns

PIO: pio0 pio1 pio2 pio3 pio4

[illegible]

Поддерживаемые режимы передачи. Звёздочка показывает текущий.

Cycle time: no flow control=240ns IORDY flow control=120ns

Commands/features:

Enabled Supported:

- * READ BUFFER cmd
- * WRITE BUFFER cmd
- * Host Protected Area feature set
- * Look-ahead
- * Write cache
- * Power Management feature set
- * Security Mode feature set
- * SMART feature set
- * Device Configuration Overlay feature set
- * Automatic Acoustic Management feature set
- SET MAX security extension
- * DOWNLOAD MICROCODE cmd

Security:

Master password revision code = 65534

supported

not enabled

not locked

not frozen

not expired: security count

not supported: enhanced erase

HW reset results:

CBLID- above Vih

Device num = 1

Checksum: correct

Имея эту информацию, мы можем ручками настраивать наш винчестер. Те параметры, которые и так настроены, настраивать не обязательно :). Сверяйтесь с map для hdparm, я расскажу про наиболее интересные параметры. \

Собственно настраиваем

Итак, параметры мы передаём как:

hdparm -параметр /имя/диска

Если `hdrparm` ругается, значит или режим не поддерживается, или параметр не верный. Если режим не поддерживается, команда `dmesg` покажет нам последней строчкой ругань драйвера.

Интересные параметры

-a Количество секторов упреждающего чтения для файловой системы, то есть софтверное упреждающее чтение. Обычно значение по умолчанию приемлемое, а наличие аппаратного упреждающего чтения смазывает эффект. Чем меньше, тем лучше в случае чтения кучи мелких файлов разбросанных хаотично по диску. Чем больше число, тем лучше для копирования фильмов и mp3.

-A Включение аппаратного упреждающего чтения винчестером. Обычно всегда включено по умолчанию. Изменять следует, если в вышеописанном информационном выводе не стоит звёздочка перед Look-ahead в разделе Enabled Supported. 1 - включено, 0 - выключено.

-B Имеет смысл для ноутбуков, управляет настройкой энергосбережения. Чем значение ниже, тем больше у винта желание остановить блин и заснуть :), от чего растёт время отклика. Значение 255 вырубает управление питанием, соответственно уменьшая время отклика. Не все винчестеры имеют собственное управление питанием. Проверьте параметр AdvancedPM= в коротком листинге с сырыми данными.

-c Значение 1 включает 32 битную передачу по внутренней шине, по шлейфу возможно только 16 бит. Значение 3 включает то-же самое с контролем чётности. Это более надёжно, но чуть медленнее чем 1. По умолчанию всегда выключено, имеет смысл включить. разницы между 1 и 3 не заметил ни по скорости, ни по надёжности.

-d 1 - включено, 0 - нет. Признак использования DMA. Если у Вас стоит звёздочка напротив одного из dma режимов в вышеприведённых листингах, значит Вам этот параметр не нужен. Если не стоит, можно попробовать включить. ВНИМАНИЕ!!! Машина может зависнуть, раздел может накрутиться!!! Если это случилось, что-то крепко не в порядке с железом или ядром!!!

-E Скорость cdrom. Если сильно воет... Просто укажите желаемую скорость. Для винтов бесполезен :)

-k и **-K** Сохранение настроек и опций винчестером. Это не означает что они автоматически сохраняются после перезагрузки. Это означает, что они сохраняются, если драйвер сделает reset контроллеру, или после спячки, но не гибернации (aka suspend to disk). 1 - сохранять. 0 - нет.

-m Количество секторов для аппаратного упреждающего чтения. Не может быть больше чем умеет винт. Смотреть в диагностический листинг, параметр MaxMultSect=. По умолчанию обычно всегда максимум.

-M Управление акустикой. 254 - шумно и быстро. 0 - медленно и тихо. Многие драйвы умеют только 128 и 254. Многие вообще нисколько не умеют :)

-P Ещё одна ручка для регулирования аппаратного упреждающего чтения. Не работает ни на одном виденном мною винте. Может у Вас заработает?

-u Включить размаскирование прерываний. Если значение 1, то контроллер разрешает другие прерывания во время операции ввода-вывода. Это снижает нагрузку на систему и повышает отклик. Нужно включать. По умолчанию выключено. На очень древних контроллерах система может повиснуть.

-W 1 - включает кеширование записи. Включено по умолчанию у всех виденных мною винтов. Но пригодится может.

-X Самый мощный параметр. Позволяет вам принудительно выставить режим работы DMA. В последних версиях **hdparm** задаётся буквенно, например **-X udma5**. Внимание!!! Не включайте режимы, не поддерживаемые Вашим контроллером, или с плохим (40-жильным) шлейфом!!! Обычно ядро само выбирает максимальный режим DMA, и если оно не смогло, или выбрало как Вам кажется меньший режим чем можно, ПОДУМАЙТЕ!!! Вдруг тому есть ОБЪЕКТИВНЫЕ причины?.

Пробуйте параметры по одному, запускайте **hdparm -tT**, смотрите в **dmesg**. Сидиромы понимают не все параметры из указанных выше. У разных винчестеров понимание тоже может отличаться :) После того как наиграетесь, будем закреплять.

Закрепляем

В **/etc/conf.d/hdparm** заносим понравившиеся параметры. У меня это выглядит так:

```
hda_args="-u1c3M254Kk"
hde_args="-u1c3M254Kk"
#hdd - cdrom
hdd_args="-u1c3Kk"
```

Делаем: **rc-update add hdparm default**

Затем: **/etc/init.d/hdparm start**

Выбор планировщика ввода-вывода

В ядрах 2.6 появилась возможность выбирать между четырьмя планировщиками ввода-вывода. У каждого планировщика есть свои достоинства и недостатки. По умолчанию в ядро всунуты все четыре, и выбран **anticipatory io scheduler**. Кратенько опишу их.

no-op - очень мелкий и лёгкий планировщик. Мало чего умеет. Для винтов не пригоден. В основном используется если вместо винта - флеш.

anticipatory - выбирается по умолчанию. Середина-наполовину для всего на свете. И там хорош, и тут хорош.

deadline - более лёгкий и простой чем **anticipatory**, лучше себя ведёт при "взрывных" нагрузках. При равномерной нагрузке имеет особенность задумываться и притормаживать.

CFQ - размазывает ввод-вывод равномерно между всеми процессами. Ввод-

вывод медленный, но плавный и равномерный независимо от загрузки. Это мой выбор на сегодня.

Как попробовать?

Передайте ядру в строке загрузки GRUB или другого загрузчика параметр `elevator=[cfq|as|deadline|noop]`, поработайте, сравните. Ненужные планировщики ввода/вывода можно убрать из конфигурации ядра перед его компиляцией. Подробнее можно узнать в статье "Компиляция ядра Linux". Они в **general setup ->**

Configure standart kernel features...

Какнибудь потом расскажу про тюнинг vm и свопа ...

Работа с CD/DVD

Основные программы

Для начала, убедитесь что у вас установлены программы **cdrecord** и **mkisofs**, которые входят в пакет **cdrtools**. Если нет -- установите его:

emerge cdrtools

Конфигурация ядра и загрузчика

Убедитесь, что у вас ядро настроено соответствующим образом.

Ядра серии 2.4.x

Для ядер серии 2.4.x в ядро необходимо включить поддержку SCSI эмуляции: Linux Kernel Configuration:

Конфигурация ядра 2.4.x

```
ATA/IDE/MFM/RLL Support -->
  IDE, ATA, and ATAPI Block Devices -->
    <M> SCSI Emulation Support
SCSI Support -->
  <M> SCSI Support
  <M> SCSI CDROM Support
  <M> SCSI Generic Support
```

Теперь, еще необходимо ядру при загрузке передать параметр `hdc=ide-scsi` (в данном случае предполагается, что ваш CD привод - это hdc).

Т.е. если вы используете grub, то его конфиг должен выглядеть примерно так:

Файл: /boot/grub/grub.conf

```
title Gentoo Linux
root (hd#, #)
        kernel      (hd#, #)/YOUR_KERNEL_NAME_HERE      root=/dev/hdx#      hdc=ide-scsi
ANY_OTHER_KERNEL_OPTIONS_NEEDED
```

Т.е. если вы используете grub, то его конфиг должен выглядеть примерно так:

Файл: /boot/grub/grub.conf

```
title Gentoo Linux
root (hd#, #)
        kernel      (hd#, #)/YOUR_KERNEL_NAME_HERE      root=/dev/hdx#      hdc=ide-scsi
ANY_OTHER_KERNEL_OPTIONS_NEEDED
```

Если вы используете lilo, то соответствующая часть конфига:

Файл: /etc/lilo.conf

```
# Linux bootable partition config
image = /boot/YOUR_KERNEL_NAME_HERE
append = "hdc=ide-scsi"
root   = /dev/hdx#
label  = Gentoo
read-only
```

Ядра серии 2.6.x

В ядрах серии 2.6.x вам уже нет необходимости включать SCSI (и, следовательно, не надо переконфигурировать загрузчик). Вместо этого, надо указать поддержку ATAPI CDROM: Linux Kernel Configuration:

Конфигурация ядра 2.6.x

```
Device Drivers -->
ATA/ATAPI/MFM/RLL support -->
  <*> Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support
  <*> Include IDE/ATAPI CDROM support
```

Подготовка ISO образа

Для начала, создайте папку:

mkdir /tmp/iso

а затем скопируйте все файлы, которые вы хотите записать, в созданную папку.
ср *.doc /tmp/iso

Затем создайте image.iso mkisofs -o /tmp/image.iso /tmp/iso

Дополнительные опции при создании ISO образа

-r : добавляет (частично) поддержку так называемых расширений Rock Ridge, т.е. сохраняет специфичные для unix атрибуты файлов. Например, добавляет поддержку длинных имён (до 255 символов, но не для MS Windows), символических ссылок. В отличие от опции -R, которая полностью соответствует Rock Ridge, права доступа на файлы не сохраняются, они становятся доступны для чтения любому пользователю.

-jcharset=koi8-r: подразумевает под собой 2 опции: 1) -J включает поддержку Joliet, которая используется для распознавания длинных имён файлов под MS

Windows; 2) **-input-charset**: устанавливает перекодировку символов для корректного отображения русских имён файлов. ("koі8-r" здесь дано для примера - замените на кодировку, используемую в вашей системе)

-joliet-long: Позволяет, чтобы имена Joliet могли быть установлены в 103 Unicode символы. Это нарушает Joliet спецификацию - но работает. Используйте с осторожностью.

-f : следование символическим ссылкам. Если вместо копирования файлов в /tmp/iso, вы просто создаёте в этой директории симлинки на них, то вы должны включить эту опцию.

-C : эта опция необходима, если вы хотите дописать мультисессионный диск (см. раздел [Как дописывать диски])

Определение ATAPI-приводов

Выполните **cdrecord --dev=ATAPI --scanbus**

Вы должны увидеть нечто похожее на следующее:

```
Cdrecord-Clone 2.01a25 (i686-pc-linux-gnu) Copyright (C) 1995-2004 J?rg Schilling
scsidev: 'ATAPI'
devname: 'ATAPI'
scsibus: -2 target: -2 lun: -2
Warning: Using ATA Packet interface.
Warning: The related libscg interface code is in pre alpha.
Warning: There may be fatal problems.
Using libscg version 'schily-0.8'.
scsibus0:
  0,0,0  0) 'SONY   ' 'CD-RW  CRX175A1 ' '5YS2' Removable CD-ROM
  0,1,0  1) *
  0,2,0  2) *
  0,3,0  3) *
  0,4,0  4) *
  0,5,0  5) *
  0,6,0  6) *
  0,7,0  7) *
```

Здесь показано, что на шине 0,0,0 есть CD-RW привод. Предупреждение: В новых версиях cdrtools (2.01.01) и ядра 2.6.x надо писать **cdrecord --dev=ATA --scanbus**

Запись ISO образа

cdrecord --dev=ATA:0,0,0 /tmp/image.iso

(Замените шину 0,0,0 на необходимую) Примечание: Вместо dev=ATA:0,0,0 можно указывать просто имя устройства, например, dev=/dev/cdrw

Дополнительные возможности

Используйте опцию **--speed=#** для указания скорости записи

cdrecord --dev=ATAPI:0,0,0 --speed=4 /tmp/image.iso

Используйте **--blank=fast** для быстрой очистки CD-RW диска.

cdrecord --dev=ATAPI:0,0,0 --blank=fast

Большинство современных записывающих устройств поддерживает технологию BURN-free. К сожалению она не включена по умолчанию при запуске **cdrecord**. Для включения, добавьте опцию **--driveropts=burnfree**. Вы можете посмотреть, какие ещё опции поддерживает ваш CD-ROM посредством команды

cdrecord dev=/dev/cdrw driveropts=help -checkdrive

Если в файле `/etc/default/cdrecord` установить переменную `CDR_DEVICE=ATAPI:0,0,0` (можно и `CDR_DEVICE=/dev/hdc`), то не придется каждый раз указывать устройство (ключ `--dev=ATAPI:0,0,0`) и тогда для очистки CD-RW диска понадобится набрать только следующее:

cdrecord --blank=fast

Если нужно разрешить пользователю запись на CD, то включите его в группу `cdrom`.

Копирование дисков

Если у вас всего один CD-ROM, то необходимо вначале создать файл-образ диска. Это можно сделать многими способами (с помощью `dd`, `cat`, `cdrdao`, `readcd` и т.д.). Лучше воспользоваться `readcd`, т.к. он уже входит в пакет `cdrttools`, и имеет множество интересных опций (смотри `man readcd`): `readcd dev=/dev/cdrw -clone f=/tmp/image.iso`

После этого запишем образ на болванку с помощью `cdrecord`:

cdrecord -v -eject dev=/dev/cdrw -raw96r -clone /tmp/image.iso

ВАЖНО: Ключи `-raw96r` и `-clone` необходимы ТОЛЬКО если вы создавали файл-образ с помощью команды `readcd -clone`.

Как дописывать диски

Прежде всего, при создании и последующем дописывании мульти-сессионных дисков, необходимо всегда для `cdrecord` указывать опцию `-multi`, а для `mkisofs` использовать ключ `-r` или `-R` (т.е. включать расширение Rock Ridge). Таким образом создание первой сессии будет выглядеть примерно так: Code: Запись первой сессии

mkisofs -r -f -jcharset=koi8-r -o /tmp/image.iso /tmp/iso/

cdrecord -v -multi driveropts=burnfree -eject dev=/dev/cdrw /tmp/image.iso

Для создания последующих сессий, нам нужна информация о номере начального сектора последней сессии и о номере сектора с которого будет начинаться новая сессия. Эта пара чисел выводится командой `cdrecord -msinfo dev=/dev/cdrw`. После этого данные числа указываются в команде `mkisofs` в качестве аргументов к опции `-C`. Также, чтобы были видны файлы из предыдущей сессии, необходим ключ `-M`, аргументом к ключу `-M` служит имя устройства вашего CD-ROM (т.е. в нашем примере - это `/dev/cdrw`). Команда `cdrecord` остаётся такой же. Резюмируя вышесказанное получаем для записи последующей сессии: Code: Запись последующих сессий

```
mkisofs -r -f -jcharset=koi8-r -C $(cdrecord dev=/dev/cdrw -msinfo) -M /dev/cdrw -o /tmp/image.iso /tmp/iso/
```

```
cdrecord -v -multi driveropts=burnfree -eject dev=/dev/cdrw /tmp/image.iso
```

Полезные ссылки

Классическое how-to по записи дисков на английском языке:

<http://www.tldp.org/HOWTO/CD-Writing-HOWTO.html>

То же самое на русском:

<http://www.opennet.ru/docs/HOWTO-RU/CD-Writing-HOWTO.html>

Создание аудио дисков (на английском языке): en:

[HOWTO Create an Audio CD](#)

Создание DVD

Введение

О чем эта статья

В данной статье будет подробно рассмотрен вопрос формирования и записи видео дисков формата DVD в Gentoo Linux.

Необходимые программы

Для начала, убедитесь что у вас установлены следующие программы: **transcode**

```
emerge -n transcode  
cdrecord и mkisofs
```

```
emerge -n cdrtools  
growisofs
```

**emerge -n dvd+rw-tools
dvdauthor**

emerge -n dvdauthor

также могут пригодиться: **kino**

emerge -n kino

cinelerra

emerge -n cinelerra-cvs

dvdrip

USE='cdr mplayer ogg subtitles xvid rar' emerge -n dvdrip

replex

**emerge -n replex
cdrecord-prodvd**

emerge -n cdrecord-prodvd

Кодирование видео в формате DVD

Использование transcode для декодирования видео произвольного формата и кодирования его в формат MPEG2, совместимый с DVD

Хотелось бы заметить, что если вы не любители программ с кучей непонятных опций, то вам следует попробовать программу dvdrip, которая является неплохим фронтом для transcode с понятным интерфейсом.

Transcode, пожалуй, лучший инструмент для нашей задачи. Вам, возможно, придётся заглянуть в man страницу по transcode, чтобы понять те команды, которые используются здесь. Также вам могут пригодиться и другие инструменты для декодирования и кодирования вашего формата, так что не поленитесь заглянуть и в их man странички. Здесь будет приведен пример того, как конвертировать стандартный divx файл в широкоэкранный формат (без черных полос сверху и снизу). Затем мы конвертируем все это в формат letterbox 1.5:1. Зачем, спросите вы. Затем, что изначально широкоэкранный формат имеет соотношение 1.73 (16:9) с разрешением 720x416. DVD в свою очередь поддерживает размеры изображения только 720x480, 704x480, 352x480 и 352x240 (возможно это не все). Поэтому самое простое и качественное решение здесь - это привести наше видео к формату 720x480 (соотношение 1.5), добавив по 32 черных полосы сверху и снизу к изображению.

transcode -i test-divx.avi -V -y mpeg -F d -Y -32,0,-32,0 -E 44100 -b 128 -o showgirls-

dvd

Отрицательные значения для параметра -Y сообщают transcode о том, чтобы добавить нужное количество черных полосок сверху и снизу. Если использовать положительные значения, то полосы будут соответственно убираться сверху и снизу. Часто вам может попасться видео уже в формате 720x480 или еще каком-то "хорошем" формате, тогда не нужно изменять его размеры, то есть просто не используйте опцию -Y.

Here's what some of those command line arguments mean: Вот, что означают остальные опции:

-V использовать YV12/I420 в качестве встроенного видео кодека [выключено изначально]

Это, как правило, намного быстрее чем RGB процессинг, но некоторые импортированные модули могут не поддерживать этот формат. Используйте эту опцию всегда, когда это возможно.

-y vmod[,amod]

видео[,аудио] модули для экспортирования [отсутствуют изначально].

Если опустить, то transcode не будет кодировать никаким модулем.

Полностью установленный transcode имеет следующие модули:

<вырезано>

mpeg - (видео) MPEG 1/2 | (аудио) MPEG 1 Layer II

asm должен быть доступен во время компиляции. Этот модуль не имеет зависимостей. Этот модуль хорошо поддерживается.

Интерфейс к библиотеке libmpeg (включена в transcode). Может кодировать в mpeg1, VCD, SVCD, MPEG2 и DVD видео форматы.

Кодированное видео является отдельным файлом, который следует объединить

с соответствующим аудио файлом после кодирования.

-F codec_string

параметры енкодера [зависят от модуля].

Параметр -F имеет разные значения для разных модулей.

<вырезано>

-y mpeg:

Синтаксис: -F "<base-profile>[,<resizer-mode>[,user-profile]]"

<base-profile> может быть одним из:

'1' = MPEG 1 (по умолчанию)

'b' = big MPEG 1 (экспериментальный)

'v' = VCD

's' = SVCD

'2' = MPEG2

'd' = DVD совместимый

<resizer-mode> может быть одним из:

0 = выключить resizer (по умолчанию)

1 = 352x288

2 = 480x480

3 = 480x576

4 = 352x240

<user-profile> Имя файла профиля. Вы можете

можно указать как абсолютный путь так и относительный

(относительно директории из которой запускаете transcode).

-Y top[,left[,bottom[,right]]]

определить (для енкодера) рамку по котрой будет обрезано изображение.
Отрицаельные значения будут добавлять рамку [отключено изначально].

-E r[,b[,c]]

частота сэмплирования выходного аудио [Гц], битов на сэмпл и каналы
[входящие данные]. Опция "-J resample" должна быть передана
не совместимым с ресэмплингом.

-B n[,m[,M]]

изменить размеры до высота-п*M горизонтальных полосок
[,ширина-м*M] вертикальных полосок [отключено изначально,32].
M должно иметь одно из значений 8, 16 или 32. Неважно какое M
вы используете. Также взгляниет на флаг fast опции -Z если
не хотите считать n и m вручную.

-b b[,v[,q[,m]]]

битрейт аудио енкодера кБит/с [,vbr[,quality[,mode]]] [128,0,5,0]
Параметр mode оперделяет какие модули будет использовать lame
при кодировании. Возможные значения для mode:

- 0 Joint Stereo (по умолчанию)
- 1 Full stereo
- 2 Mono

-o file

имя выходного файла, по умолчанию [/dev/null].

Если бы, к примеру, наше видео имело формат 728x424, то следовало бы воспользоваться командой типа следующей:

transcode -i showgirls-divx.avi -V -y mpeg -F d -B 1,1,8 -E 44100 -b 128 -o showgirls-dvd

Также обратитесь к опции -X для того, чтобы увеличить размеры видео.

Использование tcmlplex для объединения аудио и видео данных

Transcode должен был создать *.m2v (mpeg-2 видео) и *.mpa (аудио) файлы в текущей директории. Теперь мы их объединим так:

tcmlplex -o showgirls-dvd.vob -i showgirls-dvd.m2v -p showgirls-dvd.mpa -m d

Использование kino (альтернатива transcode+tcmlplex)

Kino используется для получения видео с dv камер. Эта программа имеет возможности простейшего видеомонтажа (вырезать сцену, вставить, поменять местами). Она имеет удобный интерфейс для захвата видео (хотя вы можете использовать dvgrab для этой цели). Она использует формат qt4linux mov по умолчанию для захвата.

Для более продвинутого монтажа cinelerra, пожалуй, пока единственный выбор под Linux. Это достаточно мощная, но достаточно долгая в освоении и пока что нестабильная (сохраняйтесь как можно чаще) программа. Вы можете просто импортировать тот же .mov полученный с помощью kino (dvgrab) в cinelerra (на

самом деле, она падает при работе с большинством других форматов). Затем вы можете использовать `cinelerra`, чтобы привести оконченный фильм обратно к формату `mov`.

Как только вы наконец получите тот `.mov` файл, какой вы хотели, загрузите его опять в `kino` и перейдите в закладку `export`. Здесь вы уже можете экспортировать `video` в формат `vcd`, `svcd` или `dvd`. Формат DVD потребует достаточно много времени для конвертирования (желателен мощный процессор).

Использование `replex` (подготовка DVB записей с цифрового телевидения (Digital Television))

"Эти инструкции были испробованы на записях формата DVB-C, сделанных в Финляндии"

Примечание: пожалуй, для России эта глава абсолютно бесполезна, но если есть большое желание, то я переведу и её. Мало ли, кто где живет ;)

Digital television streams are transmitted as MPEG2-TS in which the TS stands for Transport Stream. This means that extra information, such as multiple audio streams, could be transferred along with the actual video. Quite fortunately, the video stream is already suitable for DVD without re-encoding which is both time consuming and stupid as it lowers the quality. Enter `replex`. First use `czap` for tuning into some channel:

```
czap -r -n 3
```

This tunes the DVB card to channel number 3. Now, record some data by typing:
`cat /dev/dvb/adapater0/dvr0 > stream.ts`

This simply writes raw data from the first tuner card into a file. After awhile press CTRL-C to stop recording.

Now, you should have a blob of data which could be played with e.g. `mplayer`. Next, you should use `replex` to transform TS format suitable for DVD. Enter following command:
`replex -k -i TS -t DVD -o stream.dvd stream.ts`

This command ignores possible errors in original stream (`-k`), assumes that input file is in TS format (`-i TS`), outputs format suitable for DVD (`-t DVD`), writes to a file called `stream.dvd` (`-o stream.dvd`) and reads its input from the file entered as a last parameter (`stream.ts`).

After a moment a file called `stream.dvd` should appear in the same directory where above command was entered. Now just follow the instructions from the next chapter onwards to burn this baby on a DVD.

Создание структуры директорий (authoring) с использованием `dvdauthor`

Это самый сложный этап! Имея готовый MPEG2 файл, воспользуйтесь программой `dvdauthor` (или вашим любимым графическим фронтом к ней (например, `dvdstyler`)), чтобы создать структуру директорий для записи DVD

(VIDEO_TS и AUDIO_TS).

dvdauthor -o . showgirls-dvd.vob && dvdauthor -o . -T

Создание DVD образа для записи

Если директории AUDIO_TS и VIDEO_TS созданы у вас в директории bar, которая в свою очередь находится в текущей директории, то введите команду:

mkisofs -dvd-video -o ./bar.img ./bar/

и вы получите образ bar.img в текущей директории, который теперь следует записать на диск вашей любимой программой для записи дисков.

Или можете записать директории AUDIO_TS и VIDEO_TS прямо из каталога bar/ набрав:

growisofs -Z /dev/dvd -dvd-video -V название_диска .

Где /dev/dvd - ваш DVD-writer.

Запись

В Linux существует множество способов записи DVD. Из уже показанных можно использовать cdrecord или growisofs.

С помощью графического интерфейса

Формат ISO можно записать с помощью программы X-CDroast (она использует cdrecord-ProDVD, которая требует ключ, что доставляет определенное неудобство).

Также можно воспользоваться программой k3b, которая использует growisofs. Это, пожалуй, самый простой способ.

Из командной строки

Можно записать образ ещё и так:

cdrecord-ProDVD -dao -speed=4 -dev=/dev/dvd ./bar.img

Воспользуйтесь командой cdrecord -scanbus предварительно, чтобы определить какое устройство использовать для записи. Для устройств ATAPI используйте формат -dev=/dev/dvd или -dev=/dev/hdd (зависит от машины), для эмуляции SCSI устройств используйте -dev=0,0,0 или -dev=0,0,1 (зависит от машины).

Если у вас нет cdrecord-ProDVD можете использовать growisofs.

growisofs -dvd-compatible -Z /dev/dvd= ./bar.img

Подсказка: С помощью growisofs вы можете писать диски налету (без создания

четырёхгигабайтного файла .iso):

```
growisofs -dvd-compat -Z /dev/dvd -dvd-video ./bar
```

Можно также воспользоваться скриптом, который расположен по адресу <http://james.nontrivial.org/projdvd.htm>

Система Portage

Portage через NFS

Проблемы

Слишком долгое время выполнения **'emerge --sync'** на всех серверах в LAN. Согласно этикету gentoo, вы не можете синхронизироваться чаще одного раза в день

Решение

Общий NFS portage. Одна машина синхронизируется под управлением cron, остальные машины монтируют свою директорию /usr/portage с помощью NFS.

Общее представление

1. Обязательная поддержка nfs в ядре. В случае компиляции ее как модуля, добавить соответствующие строчки в modules.autoload
2. Если потребуется, перезагрузить или загрузить модуль nfs
3. Выполнить emerge **nfs-utils**
4. Настроить NFS/fstab для автоматического монтирования общего репозитория portage.
5. Установить локальный PORTAGE_TMPDIR на локальную файловую систему.

Выгоды

Только один сервер заботится о синхронизации portage (ночью по заданию cron'a). Не нужен локальный rsync сервер.

Общий для всех каталог /usr/portage/distfiles означает, что вам не нужно скачивать одно и то же каждый раз, чтобы синхронизировать репозиторий portage на ваших компьютерах.

И вообще, NFS - это классная штука ;)

Предупреждение: NFS является очень скудным протоколом по части безопасности. Убедитесь, что используете его только в пределах вашей локальной сети, где вы полностью доверяете вашим пользователям. Мы вас предупредили!

Подробный процесс

Поддержка ядра

У вас ДОЛЖНА быть включена поддержка NFS в вашем ядре, либо вкомпиленная в ядро (в таком случае необходимо будет перезагрузиться):

Linux Kernel Configuration: Встроенная поддержка

File systems --->

Network File Systems --->

<*> NFS file system support

[*] Provide NFSv3 client support

[] Provide NFSv4 client support (EXPERIMENTAL)

[] Allow direct I/O on NFS files (EXPERIMENTAL)

<*> NFS server support

[*] Provide NFSv3 server support

[] Provide NFSv4 server support (EXPERIMENTAL)

[] Provide NFS server over TCP support (EXPERIMENTAL)

либо в качестве модулей:

Linux Kernel Configuration: Включение модулей

File systems --->

Network File Systems --->

<M> NFS file system support

[*] Provide NFSv3 client support

[] Provide NFSv4 client support (EXPERIMENTAL)

[] Allow direct I/O on NFS files (EXPERIMENTAL)

<M> NFS server support

[*] Provide NFSv3 server support

[] Provide NFSv4 server support (EXPERIMENTAL)

[] Provide NFS server over TCP support (EXPERIMENTAL)

Добавляем NFS server support, так как на данной машине будет размещаться общий каталог portage. Поддержку клиента NFS можно добавить по желанию.

Сохраните конфиг вашего ядра и пересоберите его. Code: Пересборка ядра

gentoo # mount /boot //если это еще не сделано

gentoo # cd /usr/src/linux

Для ядер 2.4:

gentoo # make dep && make bzImage modules modules_install install

Для ядер 2.6:

gentoo # make && make modules_install

Включение поддержки

Перезагрузитесь, если включили поддержку NFS в ядро, или, если вы собрали ее как модули, просто запустите: `modprobe nfs`

Установка и настройка

Установите пакет

nfs-utils: emerge nfs-utils

Настраиваем автоматическое монтирование общего portage

rc-update add portmap default

rc-update add nfs default

Отредактируйте crontab на сервере (метод зависит от того, какой cron вы используете). Для `vi` и `dcron` команда выглядит так: `crontab -e` Добавьте код:

```
0 0 * * * emerge --sync > /dev/null 2>&1 || true --nospinner && emerge world -vup
```

Отредактируйте файл `/etc/exports` на сервере, чтобы в нем были следующие строки:

```
/usr/portage ip_range/subnet(sync,no_root_squash,rw)
```

Отредактируйте `/etc/fstab` на клиентах, добавив в него следующий код:

```
SERVER_IP:/usr/portage /usr/portage nfs bg,hard 0 0
```

Отредактируйте `/etc/modules.autoload.d/kernel-version`, добавив следующую строку: `nfs`

Примечание: Кеш портежей не расшарен через `nfs`. Если каждый раз вас докучает обновление кеша при команде `emerge` сразу после синхронизации, попробуйте добавить следующее в `crontab` клиентов. Это обновит кеш на компьютерах ваших клиентов спустя 10 минут после старта синхронизации на сервере. Установите время согласно скорости вашего подключения.

```
10 0 * * * emerge --metadata
```

PORTAGE_TMPDIR

`PORTAGE_TMPDIR` должен быть локальным каталогом (для скорости).

Переменная **PORTAGE_TMPDIR** устанавливается в файле `/etc/make.conf` и по умолчанию указывает на `/var/tmp`, то есть на локальную файловую систему, но просто для того, чтобы убедиться в том, что это так, а не иначе, сделайте: `emerge info` Если всё прошло успешно (а всё так и должно быть), вы спокойно можете очистить содержимое каталога `/usr/portage` на КЛИЕНТСКОЙ машине (например, чтобы освободить дополнительное свободное место на диске)

Пояснение кода

Запись crontab'a

```
0 0 * * * emerge --sync > /dev/null 2>&1 || true --nospinner && emerge world -vup
```

говорит о том, что раз в сутки, в полночь будут выполняться синхронизация с перенаправлением вывода в мусор и без бегунка и `emerge world -vup` и отправка письма root'у о том, что должно быть обновлено - настройте корректно почту).

/etc/exports:

"/usr/portage" - каталог для экспорта

"ip_range/subnet" - только компьютеры из этого диапазона могут подключать его по сети

"(sync,no_root_squash,rw)" - полезные параметры, всегда мною используемые

/etc/fstab:

"SERVER_IP:/usr/portage" - что вы хотите подключить

"/usr/portage" - куда вы хотите подключить

"nfs" - ФС ресурса

"bg,hard 0 0" - полезные параметры, всегда мною используемые

/etc/modules.autoload.d/kernel-version:

"nfs" сообщает какой модуль загружать при загрузке ядра версии kernel-version

Комментарии

Советы? Пожелания? Похвала? Дайте нам знать.

Если вы получаете ошибку "Error starting NFS daemon" попробуйте

```
# mount -t nfsd nfsd /proc/fs/nfsd
```

это работает для меня.

Настройка Portage

Portage Setup Howto

Целевая аудитория

Это руководство создается для людей, которые хотят использовать все возможности системы Portage.

Зачем нужно это howto?

Portage — это система управления пакетами Gentoo Linux. Ее возможности довольно велики — сборка пакетов из исходников, включая конфигурирование пакета на основе пользовательских USE-флагов; начальная настройка

приложений; установка предкомпилированных пакетов; удаление программ; автоматическое обновление, и т. д. Однако, как показывает опыт, несмотря на отличную документацию, большинство пользователей не знают всех возможностей этой замечательной системы, поэтому очень часто Gentoo Linux подвергается незаслуженной критике. Данный документ — это попытка восполнить этот пробел.

Профили Portage (Portage profiles)

Gentoo — многоплатформенная система, при этом количество поддерживаемых архитектур постоянно увеличивается. Поэтому возникает необходимость в так называемых профилях сборки. **Профиль** — это набор настроек по умолчанию для определенной архитектуры, или определенного класса задач (например, для систем с повышенной безопасностью существуют профили `hardened` и `selinux`). Наличие профилей, которые легко создавать, позволяет называть Gentoo не дистрибутивом, а метадистрибутивом, то есть дистрибутивом для создания других дистрибутивов ;). Но не об этом сейчас разговор.

Итак, для того, чтобы иметь работающую Gentoo-систему, необходимо выбрать себе нужный профиль. Профили по умолчанию находятся в каталоге **`/usr/portage/profiles`**. По названию профиля легко определить тот, который нужен вам. Например, **`'default-x86-2004.2'`** — это стандартный профиль для архитектуры x86 версии 2004.2. Если в каталоге с профилем есть файл `deprecated`, то это означает, что данный профиль больше не поддерживается и выбирать его не следует.

После того как вы определились с выбранным профилем необходимо создать симлинк **`/etc/make.profile`** на каталог с выбранным профилем. Например, `ln -s /usr/portage/profiles/default-x86-2004.2 /etc/make.profile` Когда выйдет новая версия Gentoo, вам не надо бежать в магазин, покупать новые диски. Достаточно просто поменять симлинк `/etc/make.profile` на симлинк, указывающий на каталог с новым профилем.

Примечание: Сейчас идет переход на новый вид профилей — Cascading

Profiles. Такие профили имеют вид:

`default-linux/x86/2004.2/`

То есть настройки, общие для нескольких профилей не надо записывать по несколько раз. Пока что это экспериментальная фишка, но если вы хотите помочь ее протестировать, то используйте на здоровье ;).

Примечание: Вообще-то переход на cascading profiles уже произошел, посему надо обновить то, что написано выше в этой секции "Portage profiles"

Настройки профилей

В профилях содержится следующая информация:

- use-флаги использующиеся по умолчанию;
- системные пакеты, то есть такие пакеты, которые устанавливаются при emerge system;
- маскированные пакеты;
- Настройки сборки пакетов по умолчанию. (CFLAGS, CHOST и другие);
- Какие пакеты надо устанавливать при наличии альтернативных. (например, хорг или xfree)

Конфигурацию профилей напрямую менять не следует, так как ваши настройки будут удалены после следующей синхронизации с rsync-сервером. О том как правильно настраивать portage будет рассказано в следующем разделе.

/etc/portage u /etc/make.conf

Для того, чтобы изменить конфигурацию выбранного профиля, нужно использовать конфигурационные файлы в /etc/portage. Если у вас нет этого каталога, то его необходимо создать. Вот список конфигурационных файлов, которые читает Portage во время своей работы:

/etc/portage/package.mask

Синтаксис: в каждой строке содержится DEPEND ATOM, то есть один из знаков >, <, >=, <=, =, категорию, название и версию пакета. Логические знаки обязательны если указана версия пакета.

Например: >=net-www/mozilla-1.7

Неправильные примеры:

net-www/mozilla-1.7 (нет логической операции)

>=mozilla-1.7 (нет категории)

Предназначение: Маскирует пакеты. Может использоваться для того, чтоб Portage не обновлял нужный вам пакет. Например, если вы хотите использовать apache-1.3 вместо apache-2.0

/etc/portage/package.unmask

Синтаксис: такой же, как и у package.mask

Предназначение: Демаскирует пакеты, которые маскированы в профиле. Используется для установки нестабильных пакетов. Рекомендуется использовать только тем, кто хочет протестировать какой-то нестабильный пакет.

/etc/portage/profile/package.provided

Синтаксис: В каждой строчке содержится полное имя пакета, то есть category/name-version

Например: dev-lang/ghc-6.2.1-r1

Предназначение: Замена 'emerge --inject'. При наличии пакета в package.provided Portage будет считать, что этот пакет уже установлен в системе.

Примечание: Иногда этого бывает недостаточно и пакет приходится вносить в /etc/make.profile/package.provided (который удаляется после каждого emerge sync)

/etc/portage/mirrors

Синтаксис: Такой же как и у файла `/usr/portage/profiles/thirdpartymirrors`.

Предназначение: Список зеркал, которые будут использоваться в первую очередь. Можно также указать зеркала sourceforge или gnu, которые будут использоваться в первую очередь.

Например: `sourceforge http://keihanna.dl.sourceforge.net/sourceforge`

Так же можно использовать тип зеркала 'local'. Такое зеркало будет проверяться, даже если в ebuild'е пакета есть `RESTRICT="NOMIRROR"`. Например:

`local ftp://gentoo.linux.kiev.ua/pub/Linux/Gentoo/distfiles/`

/etc/portage/package.use

Синтаксис: `DEPEND ATOM USE-флаги`

Например: `x11-libs/gtk+ doc`

Предназначение: Установка индивидуальных USE-флагов для отдельных пакетов

/etc/portage/package.keywords

Синтаксис: `DEPEND ATOM KEYWORD KEYWORD` — это идентификатор, который показывает, насколько стабилен пакет на данной архитектуре. Всего есть 4 вида KEYWORDS:

- `arch` — пакет стабилен на архитектуре `arch`
- `~arch` — пакет стабилен на архитектуре `arch`, но еще не прошло время тестирования
- `-arch` — у пакета наблюдаются проблемы на архитектуре `arch`.
- `-*` — пакет нестабилен на всех архитектурах

Например: `>=app-editors/emacs-cvs-21.3 ~x86`

Предназначение: Установка индивидуальных KEYWORDS для отдельных пакетов. Используйте если хотите, чтоб в вашей стабильной системе были отдельные нестабильные пакеты или наоборот :)

/etc/portage/categories

Синтаксис: В каждой строке содержится название категории. Предназначение: Используется для создания новых категорий для Portage.

Например: `app-vasia`

Теперь вы можете создать ebuild для пакетов категории `app-vasia`. `emacs $ {PORTDIR_OVERLAY}/app-vasia/pupkin/pupkin-0.0.1.ebuild`

/etc/make.conf

Файл `/etc/make.conf` служит для настройки самого процесса сборки пакетов. Никаких сложных опций там нет, читайте внимательно комментарии в `/etc/make.conf.example`.

\$PORTDIR_OVERLAY

Представьте себе такую ситуацию: вы хотите установить пакет, ebuild'а которого нет в официальном дереве, но кто-то уже его написал. Или вы собираетесь самостоятельно научиться писать ebuild'ы, и вы хотите, чтоб Portage мог устанавливать нужные вам программы. Если вы запишете свой ebuild в /usr/portage, то он сотрется при следующей синхронизации с rsync-сервером. Для того, чтобы third-party ebuild'ы сохранялись, необходимо создать каталог для их хранения (например, /usr/local/portage) и записать его имя в переменную \$PORTDIR_OVERLAY в файле /etc/make.conf. Теперь вы можете размещать свои ebuild'ы в этом каталоге, пример был показан в предыдущем разделе.

Portage Overlay

Введение

Это руководство расскажет о том как установить ebuild который не включен в официальное дерево портежей.

Установка Portage Overlay

Первым делом мы должны указать PORTAGE_OVERLAY директорию, для этого необходимо отредактировать файл /etc/make.conf

Файл: /etc/make.conf

```
PORTDIR_OVERLAY="/usr/local/portage"
```

Таким образом мы позволяем устанавливать дополнительные ebuild'ы при этом не нарушая процесс синхронизации основного дерева портежей.

Месторасположение ebuild

Проверьте наличие директории /usr/local/portage, и если ее нет создайте ее `install -d /usr/local/portage`

Когда помещаете новый ebuild в /usr/local/portage, вы должны использовать такую же схему как в /usr/portage (category/program/program.ebuild)

Создание digest-файлов

digest-файлы -- это файлы, которые содержат md5 суммы файлов, необходимых для установки пакетов. После того, как Portage скачает необходимые файлы, будет сделана проверка на соответствие файлов их md5-суммам (это дает возможность убедиться в целостности файлов). Для того, чтобы Portage мог устанавливать пакеты, необходим digest-файл. Digest-файл можно создать с помощью команды `ebuild /usr/local/portage/category/program/program.ebuild digest` Вручную редактировать созданные этой командой файлы не следует. При этом надо учитывать требования к имени ebuild-а. При переименовании ebuild-а в произвольное имя, команда

`ebuild /usr/local/portage/category/program/program.ebuild digest` выдаст ошибку:

!!! `/usr/local/portage` does not seem to have a valid PORTDIR structure.

Чтобы этого не произошло необходимо переименовывать `ebuild` в то же самое имя, но с приставкой номера релиза.

Пример:

В портежах имеется `ebuild`:

`/usr/portage/media-sound/ncmnc/ncmnc-0.11.1-r1.ebuild`

И архив исходных кодов:

`/usr/portage/distfiles/ncmnc-0.11.1.tar.gz` для этого `ebuild-a`.

Вы хотите наложить свой патч и добавить `ebuild` вашего пропатченного пакета в систему. Для этого вы должны скопировать исходный `ebuild` в ваш новый, но добавить к нему приставку `-rN`, где `N` - номер релиза. Если такая приставка уже есть - изменить её номер:

**`cp /usr/portage/media-sound/ncmnc/ncmnc-0.11.1-r1.ebuild`
`/usr/local/portage/media-sound/ncmnc/ncmnc-0.11.1-r2.ebuild`**

После этого пропатчите исходные коды пакета, на который ссылается исходный

`ebuild /usr/portage/distfiles/ncmnc-0.11.1.tar.gz`

и сохраните его в `/usr/portage/distfiles/ncmnc-0.11.1-r2.tar.gz`

Далее, нам нужно будет поправить наш новый `ebuild`:

`vi /usr/local/portage/media-sound/ncmnc/ncmnc-0.11.1-r2.ebuild`

И изменить строку:

`SRC_URI="http://mercury.chem.pitt.edu/~shank/${P}.tar.gz`
`mirror://sourceforge/musicpd/${P}.tar.gz"`

На строку:

`SRC_URI="http://mercury.chem.pitt.edu/~shank/${P}-r2.tar.gz`
`mirror://sourceforge/musicpd/${P}-r2.tar.gz"`

Т.е. добавив наш префикс `-r2`, чтобы система знала, какой именно файл с исходными кодами ей необходимо прописать. В данном случае скачать с сайта этот пакет конечно же не удастся, но нам этого и не требуется - нам необходимо установить нами поправленный пакет.

После этого даём команду

`ebuild /usr/local/portage/media-sound/ncmnc/ncmnc-0.11.1-r2.ebuild digest`,
после чего `emerge` будет знать о вашем пакете.

Установка нового ebuild

Когда вы будете устанавливать пакет используя новый ebuild, скорее всего он будет masked, поэтому надо добавить имя пакета в /etc/portage/package.keywords. Создайте директорию /etc/portage/, если она еще не создана, и затем выполните команду:

```
echo "<category>/<package> ~x86" >>/etc/portage/package.keywords
```

Теперь этот пакет можно установить как и любой другой:

```
emerge -p package
```

```
emerge package
```

Дополнительно

Unofficial ebuilds содержит список с коротким описанием сайтов где можно найти неофициальные пакеты. Многие, написанные юзерами ebuild'ы, которые еще не попали в официальное дерево, находятся в Bugzill'e Gentoo. Найти нужные ebuild'ы можно с помощью поиска

Экономия трафика с помощью deltup

Что это такое?

Как известно, программы в Gentoo Linux устанавливаются из исходников. Это чистый текст. Однако, такие программы как, например, gss или openoffice весят непростительно много. А у большинства из нас интернет не позволяет качать файлы больших объемов, а пользователи модемов вообще толком ничего обновить не могут. Для людей, которые экономят свои деньги, и была написана программа deltup. Вместо wget мы заставляем emerge использовать getdelta. Эта программа скачивает со специального сайта не весь архив программы, а только разницу между требуемой версией и тем, что у вас есть. Если у вас нет предыдущей версии программы, то придется качать весь архив.

Установка и настройка

Установка deltup

Теперь установим deltup:

```
emerge deltup и getdelta: emerge getdelta
```

Примечание: Если при выполнении emerge deltup выйдет ошибка "!!!All ebuilds that could satisfy "deltup" have been masked." перед командой добавьте ACCEPT_KEYWORDS="~x86"

Настройка emerge

Теперь нужно предупредить emerge о том что мы хотим использовать getdelta вместо wget. Для этого добавим в /etc/make.conf параметр FETCHCOMMAND:
Файл: File /etc/make.conf

FETCHCOMMAND="/usr/bin/getdelta.sh \${URI}"

Конфигурация getdelta

У **getdelta** есть свой файл конфигурации:/etc/deltup/getdelta.rc. Рассмотрим некоторые полезные параметры:

DELTUP_SERVER - указывает на используемый deltup сервер. Рекомендуется оставить без изменений, так как deltup-сервера сейчас собраны в сеть обращение к которой идёт через один сервер linux01.gwdg.de.

QUEUERETRY - количество секунд, которые будет ждать getdelta,пока сервер не сделает dtu-файл.

MAXIMUM_ACCEPTABLE_QUEUEPOS - максимальный номер в очереди ожидания. В связи с ростом загруженности deltup-сервера имеет смысл установить число побольше.

REMOVE_OLD - удалять старые версии файлов. Полезно, если вы не хотите складировать дистфайлы.

DO_NOT_REMOVE - путь к файлу, содержащему имена файлов, которые не следует удалять при включенной опции **REMOVE_OLD**

Дополнительно

Могут возникнуть проблемы если вы скачали часть файла нужного для установки. Дело в том, что для продолжения докачки emerge использует не FETCHCOMMAND, а RESUMECOMMAND. Однако не следует менять RESUMECOMMAND на getdelta, так как getdelta не поддерживает докачку.

Обновление портежей без доступа в интернет непосредственно из системы

Данное руководство может помочь вам если с доступом в интернет из системы Gentoo проблемы. Однако у вас есть свежий portage-xxxxxxxx.tar.bz2. Если portage-xxxxxxxx.tar.bz2.md5sum не старше 40 дней(возможно это ограничение можно убрать, но автор этих строк не знает как) тогда проще сделать так :

Создаем папку /var/tmp/emerge-webrsync/ командой:

mkdir /var/tmp/emerge-webrsync/

Копируем файлы portage-xxxxxxxx.tar.bz2 и portage-xxxxxxxx.tar.bz2.md5, которые находятся к примеру, в корневом каталоге на CD-ROM диске (/mnt/cdrom/portage-xxxxxxxx.tar.bz2 и /mnt/cdrom/portage-xxxxxxxx.tar.bz2.md5sum), командами:

```
# cp /mnt/cdrom/portage-xxxxxxx.tar.bz2 /var/tmp/merge-webrsync/  
#cp /mnt/cdrom/portage-xxxxxxx.tar.bz2.md5sum /var/tmp/merge-webrsync/  
и обновляем портежи при помощи  
merge -webrsync: # merge-webrsync
```

Теперь мы получили систему портежей по состоянию на xxxxxxxx.

Если же portage-xxxxxxx.tar.bz2 старше 40 дней тогда:

Первым делом переименовываем каталог /usr/portage например в /usr/portage2:

```
# mv /usr/portage /usr/portage2
```

затем создаем переименованный /usr/portage:

```
# mkdir /usr/portage
```

Перемещаем из /usr/portage2/distfiles исходники (если вам они нужны :)) обратно в /usr/portage/distfiles:

```
# cp /usr/portage2/distfiles /usr/portage/distfiles
```

Теперь можно распаковывать наш новый portage-xxxxxxx.tar.bz2, который находится, к примеру, в корневом каталоге на CD-ROM диске (/mnt/cdrom/portage-xxxxxxx.tar.bz2), командой:

```
# tar -xvjp /mnt/cdrom/portage-xxxxxxx.tar.bz2 -C /usr/portage
```

После достаточно продолжительной распаковки нужно обновить кэш системы портежей:

```
# emerge metadata
```

Теперь мы получили систему портежей по состоянию на xxxxxxxx, а в каталоге /usr/portage2 - её бэкап.

Дополнительно можно прочитать:

[HOWTO Обновление пакетов без доступа в интернет непосредственно из системы.](#)

Обновление пакетов без доступа в интернет непосредственно из системы

В моем случае это понадобилось в следующей (я полагаю довольно распространенной) ситуации: дома - нет интернета (дорогой, медленный - нужное подчеркнуть :)), а на работе - хороший и бесплатный. Итак для начала необходимо получить список нужных пакетов. В этом примере мы хотим установить/обновить glibc baselayout texinfo gettext zlib binutils gcc ncurses. Список мы сохраняем на устаревший магнитный носитель, но счастливые обладатели более надежных носителей естественно могут пользоваться ими.

Получение списка для скачивания:

(Не забываете '2' перед '>') Code:

```
# emerge -fp glibc baselayout gettext zlib binutils gcc ncurses 2> stage1.list
```

```
# mount -t vfat /dev/fd0 /mnt/floppy
```

```
# cp /mnt/gentoo/stage1.list /mnt/floppy
```

umount /mnt/floppy

Теперь вставляем дискету в компьютер с хорошим каналом. (В моем случае это рабочий компьютер.) Если заглянуть в файл stage1.list, можно увидеть, что в нем перечисляются несколько ссылок на каждую загрузку. К сожалению, это не совсем то, что нам надо. Сначала список надо почистить от лишнего:

Очистка от лишних ссылок:

Этот скрипт привязан к формату вывода emerge, который может измениться без предварительного предупреждения - используйте с осторожностью! # cut -f 1 -d ' ' stage1.list > stage1.download

Теперь используем wget для загрузки списка пакетов:

wget -N -i stage1.download

Получив все файлы, переносим их в наш компьютер в /mnt/gentoo/usr/portage/distfiles. И для проверки запустим

emerge -pv glibc baselayout gettext zlib binutils gcc ncurses

Если в строке Total Download: указана цифра 0 - значит все было сделано верно. Можно теперь запускать установку:

emerge glibc baselayout gettext zlib binutils gcc ncurses

У всего вышеописанного есть один недостаток. В файле stage1.download есть все необходимые для установки файлы. Даже те, которые у вас могут быть в distfiles. А как вырезать ссылки из stage1.download на имеющиеся в distfiles файлы я не знаю. Если кто знает - напишите.

Взято с Gentoo.org: <http://www.gentoo.org/doc/ru/altinstall.xml>

Полное обновление системы

Приведение в порядок /var/lib/portage/world

В world должен быть список программ, которые нужно доустановить к тем, которые уже входят в "system" (т.е. в текущий профайл). в world не должно быть никаких библиотек, и т.д., которые не нужны сами по себе, а нужны только для удовлетворения чьих-то зависимостей (чтобы не продолжать устанавливать/обновлять их, если они уже станут не нужны по какой-то причине) программ, которые уже входят в "system", не должно быть в world. В world нельзя указывать определенную версию софта, это лучше делать в /etc/portage/package.mask.

Скрипт **regenworld** может помочь восстановить world путем анализа /var/log/emerge.log и генерации на его базе файла world (он перезапишет текущий world!).

Скрипт **dep -p -w** поможет найти избыточные записи в world(которые всё-равно нужны другим записям в world или входят в system).

Перед серьёзными обновлениями желательно просмотреть `/etc/portage/*`, т.к. там могут быть уже не актуальные записи мешающие текущему обновлению.

Обновление profile

Не каждый Gentoo release включает в себя новый profile (например, 2004.1 был без profile). Даже если новый profile есть, то переходить на него не обязательно (если это будет обязательно, то старый профайл будет deprecated и emerge об этом должен будет громко кричать).

Инструкции по обновлению profile будут выкладываться здесь:

<http://www.gentoo.org/doc/en/gentoo-upgrading.xml>

и как правило сводится к изменению симлинка `/etc/make.profile`

USE-флаги

Запустить **emerge -uDpv --newuse world** и проверить что USE-флаги для всех пакетов выставлены корректно, и при необходимости скорректировать

USE-флаги выставляются в `/etc/make.conf` и `/etc/portage/package.use`

Запуск обновления системы (если не нужно обновлять toolchain)

Если

emerge -uDav --newuse world

показывает что будет обновляться пакет входящий в toolchain (linux-headers, glibc, binutils или gcc), то крайне рекомендуется полностью перекомпилировать всю систему - см. следующий пункт - а иначе можно вместо следующего пункта просто запустить:

emerge -uDav --newuse world

Некоторые причины не использовать emerge -U вместо -u

Причина 1: Проблемы со SLOT

Это, к примеру, происходит потому, что некоторые люди хотели gimp-2 вместо gimp-1.2. Представьте ситуацию, где gimp-1.2 помечен stable и находится в SLOT 1, gimp-2 помечен unstable и находится в SLOT 2. Теперь при выполнении `ACCEPT_KEYWORDS=~x86 emerge gimp` получите gimp-2.

Позже, когда вы посчитаете, что наступило время обновить свою систему чем-либо похожим на "emerge -U world", эта команда установит gimp-1.2, потому, что gimp находится в world-файле, и флаг "-U" не обрабатывает SLOT должным образом.

Причина 2: Проблемы, в случае удаления ebuild-ов с Portage-дерева.

Допустим, в Portage находятся 2 версии пакетов foo, foo-1.4 (помеченный как stable) и foo-1.6 (помеченный как unstable). Вы хотите вариант unstable и делаете emerge, как с вышеуказанным gimp. Позже обновляете world как было сказано выше, но в промежутке этого времени вышло критическое обновление для foo-1.6 - foo-1.6.1. Теперь появляется несколько возможностей обработки.

foo-1.6 был удален из Portage. Будет установлен foo-1.4, несмотря на "снижение"

версии вместо флага "-U"

Ситуация будет еще хуже, если foo-1.6 не был удалён из Portage по какой-либо причине: foo-1.6 (тот, что с критической уязвимостью) будет оставаться на вашей системе до тех пор, пока не будет помечено stable что-либо выше чем foo-1.6.

Обновление одного из пакетов входящих в toolchain

Если обновляется хотя-бы один из linux-headers, glibc, binutils или gcc, то рекомендуется пересобрать их дважды, после чего весь system, после чего весь world.

Примечание: Цель двойной компиляции **toolchain** - получить гарантированно стабильный и корректный **toolchain** не зависящий от предыдущего. Перекомпилировать system/world после этого жёсткой необходимости нет, по крайней мере если остальной софт продолжает работать (возможно даже используя библиотеки из старого toolchain - см. предыдущие пункты об апгрейде).

Цель перекомпиляции system/world - чтобы весь софт получил потенциальное преимущество от установки нового toolchain. system перекомпилируется перед world из тех-же соображений, т.к. при компиляции программ из world используются утилиты из system.

Если увеличивается первая или вторая цифра версии gcc, то перед второй сборкой нужно переключиться на новую версию через gcc-config - иначе новый gcc просто установится параллельно со старым в "новый слот", но по умолчанию использоваться будет старый.

При сборке system после двойной перекомпиляции toolchain нет необходимости опять компилировать toolchain как часть system. Аналогично при сборке world после system нет необходимости опять компилировать пакеты из system как часть world. Это можно попробовать обойти либо вручную, либо используя скрипты [1], либо через бинарные пакеты и `emerge -k` (я предпочитаю последний вариант).

Итак, рекомендованный набор команд:

для того, чтобы безопасно использовать `emerge -k` нужно очистить

каталог с текущими бинарными пакетами

(напр., переместить его в /tmp/portage-packages)

pkgdir=\$(portageq pkgdir)

mv \$pkgdir /tmp/portage-packages1

install -d -o portage -g portage \$pkgdir

первая сборка toolchain

emerge linux-headers glibc binutils gcc-config gcc

выбрать новый gcc если он установился в новый слот

gcc-config имя_или_номер_нового_gcc

см. `gcc-config -l`

source /etc/profile

компиляция toolchain с созданием бинарных пакетов

emerge -b glibc binutils gcc portage

не компилировать glibc, binutils и gcc

emerge -bke system

не компилировать предыдущие пакеты (включая system)

emerge -bke world

Примечание: Чисто теоретически существует пакет binutils-config, который когда-нибудь может потребоваться использовать аналогично gcc-config.

Обновление безопасности.

Примечание: Даже после `emerge -uDav --newuse world` в системе могут оставаться устаревшие пакеты с дырами в безопасности - в слотах!

glsa-check -l | grep '[N\']

emerge ... # если нужно

Удаление неиспользуемых пакетов.

После обновления системы в ней могут оказаться пакеты, которые никто не использует. Эти пакеты желательно удалить, т.к. они не будут в дальнейшем обновляться при `emerge -uDav --newuse world`.

emerge -a depclean # очень осторожно!!!

После обновления библиотек может потребоваться перекомпилировать программы, которые эти библиотеки используют:

Примечание: Для glsa-check, revdep-rebuild необходимо установить пакет gentoolkit

rm /root/.revdep-rebuild*.*_*

revdep-rebuild -p

revdep-rebuild

Обновление конфигов.

dispatch-conf

Если используется runit-init и обновлялся пакет baselayout, то нужно восстановить /sbin/init:

ls -l /sbin/*init*

if (/sbin/init это бинарник, а не симлинк) {

mv /sbin/init /sbin/init-sysv

ln -s runit-init /sbin/init

}

Отслеживание важных сообщений при установке пакетов.

В процессе emerge world выдаётся очень много сообщений, причём важные комментарии перемешаны с командами компиляции, и отследить их при сборке нескольких пакетов одновременно не возможно.

Но все эти сообщения можно получить из log-файлов после окончания установки **emerge world**. Для этого нужно использовать либо **enotice**, либо **portlog-info**.

Сборка на другом компьютере

Существует много методов добиться этого, но мы используем самый удобный - тот, что работает с портежами.

Другие методы могут быть более подходящими для других ситуаций, когда главная проблема не только медленный процессор и долгое время сборки, например, если надо перенести Gentoo на систему, которая не может загружаться с CD или сети.

Скомпилировать Gentoo для старых систем x86 (например, Pentium Pro 200MHz с небольшим количеством RAM) можно на x86_64-системе с работающей Gentoo. Обратный трюк, собрать 64-битную систему на 32-битном компьютере, невозможен.

Я собирал систему в chroot-окружении согласно Gentoo-handbook, используя мою 64bit-Gentoo вместо "живого" CD:

```
# mkdir /your/new/gentoo
```

```
# tar xjvpf stage3-x86-*.tar.bz2 -C /your/new/gentoo
```

Отредактируйте переменные CHOST и CFLAGS, соответствующие вашей целевой системе (не изменяйте CHOST, если используете стадию 2 или 3 - возьмите стадию для соответствующей платформы). У меня было:

```
# CHOST="i686-pc-linux-gnu"
```

```
# CFLAGS="-march=pentiumpro -Os -momit-frame-pointer -pipe"
```

```
# CXXFLAGS="$CFLAGS"
```

Далее переходим в chroot-окружение и выполняем все шаги, описанные в [Руководстве Gentoo по установке](#).

Чтобы собрать ядро для i386 на x86_64-машине добавьте ARCH=i386 при конфигурации и компиляции ядра.

```
# make menuconfig ARCH=i386
```

```
# make clean dep modules modules_install bzImage ARCH=i386
```

```
# cp arch/i386/boot/bzImage /boot/vmlinuz
```

Размечать диск и создавать файловые системы на старом компьютере вам

придется с помощью "живого" CD или другой системы Linux.

Переносим готовую систему на старый компьютер:

```
# cd /your/new/gentoo
# echo "/sys/*" >> tar_exclusions
# echo "/var/run/*" >> tar_exclusions
# echo "/tmp/*" >> tar_exclusions
# echo "/proc/*" >> tar_exclusions
#
# tar -C /your/new/gentoo -X tar_exclusions --preserve -cf ../gentoo.tar .
## можете добавить -v для подробного вывода или -j / -z для компрессии
# scp [-P ssh_port] ../gentoo.tar user@dest_pc:
```

На старой машине подготавливаем диски и распаковываем систему:

```
# tar -C /your/new/system --preserve -xf ~user/gentoo.tar
```

Теперь вы можете войти в chroot-окружение. Сконфигурируйте системный загрузчик и перегрузитесь. Теперь вы находитесь в новой, с иголки, Gentoo.

Система теперь должна работать, но я заметил проблему - я не могу ничего скомпилировать. Похоже, что не работает ни C-препроцессор, ни gcc (обычно configure говорит, что /lib/cpp fails sanity check). Не знаю почему, но копирование /usr/include из chroot-каталога с исходной машины на старый компьютер эту проблему решило.

Удачи!

Оригинал на http://gentoo-wiki.com/HOWTO_Compile_on_another_computer

Перевод Poor Fred

Прочее

Разделы MS Windows по-русски

Введение

Достаточно часто возникает необходимость подмонтировать раздел с установленным M\$ Windows.

Ядро

Для того, чтобы система могла работать с разделами FAT или NTFS, необходимо включить их поддержку в ядре.

Linux Kernel Configuration: Filesystems

File systems ->

DOS/FAT/NT Filesystems ->

- (M) DOS FAT fs support
- (M) MSDOS fs support
- (M) VFAT (Windows-95) fs support
- (M) NTFS file system support

Также надо включить поддержку следующих кодировок:

Linux Kernel Configuration: Native Language Support

File systems ->

Native Language Support ->

- <M> Windows CP1251 (Bulgarian, Belarusian)
- <M> Codepage 866 (Cyrillic/Russian)

Монтирование

Теперь можно примонтировать любой windows-раздел программой mount. `mount -t file_system /dev/device /mnt/dir` В нашем случае в качестве file_system необходимо указывать vfat или ntfs, в зависимости от файловой системы раздела с windows.

Монтирование FAT

Для монтирования разделов FAT нужно использовать следующую команду:

```
mount -t vfat -o codepage=866,icharset=koi8-r,quiet,umask=000 /dev/hdd2 /mnt/win1
```

Для украинских символов:

```
mount -t vfat -o codepage=866,icharset=koi8-u,quiet,umask=000 /dev/hdd2 /mnt/win1
```

Мы использовали дополнительные опции icharset, codepage, quiet и umask.

Рассмотрим их внимательнее:

- **codepage** - это кодовая страница используемая на монтируемой ФС. Так как мы из России, то нужно писать **codepage=866**. Эта опция нужна для перекодировки имен файлов.
- **icharset** - указывает какую кодировку использовать для ввода/вывода. Нужно указать свою системную локаль. Локаль можно узнать командой locale. (Здесь под вводом/выводом понимается ввод с и вывод на терминал. Именно поэтому нужно указывать кодировку локали, т.е. кодировку которую используют программы запущенные в терминале.)
- **quiet** - при копировании на файловую систему FAT программа попытается записать права доступа, владельца и т.д., но на FAT их сохранить нельзя и будет выведено сообщение об ошибке (хотя файл скопируется). Чтобы

ошибки не появлялись мы просим смонтировать в "молчаливом" режиме

- **umask** - в некоторых случаях у простого пользователя может не хватить прав на чтение или запись файлов из-за неправильно установленных прав на файлы и директории. Поскольку у файловой системы FAT нет прав доступа, они выставляются операционной системой (в данном случае Linux) на лету в соответствии с указанным значением umask. Про umask смотрите в umask(2)

Параметры **codepage** и **iocharset** можно задать прямо в ядре. Тогда не нужно будет писать их каждый раз:

Linux Kernel Configuration: FAT

File systems ->

DOS/FAT/NT Filesystems ->

(utf8) Default iocharset for FAT

(866) Default codepage for FAT

То есть вместо

mount -t vfat -o codepage=866,iocharset=utf8,quiet,umask=000 /dev/hdd2 /mnt/win1 можно писать

mount -t vfat -o quiet,umask=000 /dev/hdd2 /mnt/win1

Монтирование NTFS

Для монтирования NTFS разделов опции mount немного изменяться. Делаем это так:

mount -t ntfs -o nls=koi8-r,umask=0,ro /dev/hdd2 /mnt/win1

Для украинских символов:

mount -t ntfs -o nls=koi8-u,umask=0,ro /dev/hdd2 /mnt/win1

Вот что они значат:

nls - указывает на вашу системную локаль. Отсутствие опций codepage и iocharset связано с тем, что NTFS разделы используют юникод для хранения имен файла.

umask - указывает на то что читать файлы с этого раздела могут все. А вот писать не разрешает опция ro. Мы рекомендуем ее поставить, так как функция записи на разделы NTFS еще не отлажена.

В случае если вы в качестве системной кодировки используете utf-8, то опции mount должны быть следующими: **mount -t ntfs -o utf8,umask=0,ro /dev/hdd2 /mnt/win1**

utf8 - использовать UTF-8 для перекодировки имён файлов.

Автомонтирование

Если у вас все еще есть разделы ОС Windows на харде, то можно добавить монтирование этих разделов при загрузке. Для этого необходимо отредактировать файл /etc/fstab. Файл: /etc/fstab

...

```
/dev/hda9 /mnt/win1 vfat auto,codepage=866,ioccharset=koi8-r,quiet,umask=000 0 0
#/dev/hda9 /mnt/win1 vfat auto,codepage=866,ioccharset=koi8-u,quiet,umask=000 0 0
/dev/hda3 /mnt/win2 ntfs auto,nls=koi8-r,umask=0,user 0 0
```

еще к опциям можно добавить showexес, чтобы все файлы не выглядели исполняемыми, в случае с FAT32

Udev

Настройка системы

Установка необходимых программ

emerge udev hotplug coldplug

Конфигурация ядра

Для активизации udev нам необходимо ядро 2.6. Лично у меня 2.6.12-mm1.

General setup --->

☒ Support for hot-pluggable devices

File systems --->

Pseudo filesystems --->

☐ /dev file system support (OBSOLETE)

☐ Automatically mount at boot (NEW)

☒ Virtual memory file system support (former shm fs)

Примечание: devfs Может быть включен, но автоматическое монтирование обязано быть выключено. А с учетом того, что из 2.6.13 его вообще собираются убрать... ну смотрите сами.

Примечание: Во всяком случае, из 2.6.14 убрали

Ядро собираем и устанавливаем.

Конфигурация системы

Не буду рассматривать как udev работает из таррбола, рассматриваю вариант чистого udev.

/etc/conf.d/rc:

...

RC_DEVICE_TARBALL="no"

...

RC_DEVFSD_STARTUP="yes"

...

Примечание: Последнюю опцию можно поставить в "но", когда мы убедимся в нормальной работоспособности udev

rc-update add coldplug boot rc-update add hotplug default

Конфигурация процесса загрузки

GrUB

К строке kernel от старого ядра необходимо дописать:

kernel /boot/kernel-2.6.9-r4 root=/dev/hda3 gentoo=noudev

А в строке kernel для нового ядра прописать:

kernel /boot/kernel-2.6.10-r6 root=/dev/hda3 gentoo=nodev

Troubleshooting

Попытка номер раз

В принципе можно перегружаться, но нас ждет одно неприятное сообщение. Вот примерно такое:

WARNING: Unable to open an initial console

Для того чтобы все было хорошо, нам необходимо загрузиться с другого носителя (например любой LiveCD) подмонтировать наш корневой раздел, и в каталоге /dev (каталог по идее должен оказаться абсолютно пустым) проделать следующее: Code:

mknod -m 660 console c 5 1

mknod -m 660 null c 1 3

ПОМАН007: Зачем же так жестоко - заставлять грузиться с компактa? Согласно udev guide с [www.gentoo.org](http://www.gentoo.org/doc/en/udev-guide.xml) (<http://www.gentoo.org/doc/en/udev-guide.xml>), доступ к девственному /dev можно получить альтернативно: Code: Code Listing 3.1: Listing device nodes available at boot

mkdir test

mount --bind / test

cd test/dev

ls

Попытка номер два

С консолью вроде все в порядке, а вот иксы не хотят... грузимся в шелл и в файле конфигурации Xorg прописываем в разделе мыши:

```
Option "Device" "/dev/input/mice"
```

PS

Ну теперь вроде все так как было раньше. и старая конфигурация тоже работает. Если старая конфигурация не нужна, то убираем все из загрузчика, ставим в /etc/conf.d/rc - "no" и живем спокойно под udev.

PPS

Кстати тарболл может потребоваться пользователям нестандартных устройств, а так же некоторых сторонних модулей ядра

Использование CFLAGS для оптимизации собранных программ

Для чего все это надо?

Стремление выжать из своего компьютера максимум производительности есть в каждом, ну почти в каждом ;-). Особенно его много в русских линуксоидах Gentoo'шниках ;) Мы попытаемся путем изменения флагов оптимизации ускорить работу приложений нашей системы.

ВНИМАНИЕ: Некоторые флаги могут сделать приложения нестабильными, так что нужно быть аккуратным.

Оптимизация

Переменная окружения CFLAGS

Для указания параметров оптимизации компилятору GCC, используется переменная окружения CFLAGS. Эта переменная определена в /etc/make.conf, её можно изменить двумя способами:

Отредактировать эту переменную в /etc/make.conf;

Экспортировать ее в окружение (emerge будет использовать эти параметры, но каждый раз выполнять export неудобно):

```
export CFLAGS='параметры оптимизации'
```

Уровни оптимизации

Для gcc версий 3.x и выше существует только 5 уровней оптимизации: -O0 (без оптимизации), -O1, -O2 и -O3 (O3 - самый высокий уровень), а так же -Os.

Примечание: Если вы используете несколько -O опций, то только последняя объявленная будет оказывать влияние на процесс компиляции.

-O0

Отключает оптимизацию. Только переменные, объявленные register, сохраняются в регистрах.

-O(-O1)

Включает оптимизацию. Пытается уменьшить размер кода и ускорить работу программы. Соответственно увеличивается время компиляции. При указании -O активируются следующие флаги: -fthread-jumps, -fdefer-pop.

На машинах, у которых есть слоты задержки, включается опция -fdelayed-branch.

На тех машинах, которые способны поддерживать отладку даже без указателя на стек функции, также включается опция -fomit-frame-pointer.

На других машинах могут быть включены и другие флаги.

-O2

Оптимизирует еще больше. GCC выполняет почти все поддерживаемые оптимизации, которые не включают уменьшение времени исполнения за счет увеличения длины кода. Компилятор не выполняет раскрутку циклов или подстановку функций, когда вы указываете -O2. По сравнению с -O, эта опция увеличивает как время компиляции, так и эффективность сгенерированного кода.

-O2 включает все флаги оптимизации наследованные от -O. Также включает следующие флаги оптимизации:

- fforce-mem -foptimize-sibling-calls
- fstrength-reduce -fcse-follow-jumps -fcse-skip-blocks
- frerun-cse-after-loop -frerun-loop-opt -fgcse -fgcse-lm
- fgcse-sm -fgcse-las -fdelete-null-pointer-checks -fexpensive-optimizations
- fregmove -fschedule-insns -fschedule-insns2 -fsched-interblock
- fsched-spec -fcaller-saves -fpeephole2 -freorder-blocks
- fre-order-functions -fstrict-aliasing -funit-at-a-time -falign-functions
- falign-jumps -falign-loops -falign-labels -fcrossjumping

-O3

Оптимизирует еще немного. Включает все оптимизации -O2 и также включает флаг -finline-functions и -fweb.

-Os

Включает оптимизацию по размеру. -Os флаг активирует все флаги оптимизации

из -O2, в основном те, которые не увеличивают размер выходного файла. В дальнейшем выполняются оптимизации по уменьшению размера кода.

-Os включает следующие флаги оптимизации: -falign-functions, -falign-jumps, -falign-loop, -falign-labels, -freorder-blocks, -fprefetch-loop-arrays.

Примечание: Более полное описание флагов -Ox, -fflag смотрите в man gcc

Оптимизация под тип процессора

Не все процессоры одинаковы, поэтому давайте укажем компилятору на наш тип процессора. Для этого есть опции -mtune и -march. Отличие в том, что с опцией -mtune компилятор сделает код, который будет совместим с более младшими моделями процессора, в то время как с -march этого не происходит. Вот список возможных значений для данных опций:

i386

i486

i586

i686

pentium

pentium-mmx

pentiumpro

pentium2

pentium3

pentium4

pentium-m

prescott

nocona

k6

k8

k6-2 (не рекомендуется ставить, из-за багов в компиляторе, заменять на i686)

k6-3

athlon

athlon-tbird

athlon-4

athlon-xp

athlon-mp

athlon64

opteron

winchip-c6

winchip2

c3.

Внимание! pentium-m - это аналог для pentium3. Если процессор в вашем ноутбуке Mobile Intel Pentium 4 - M, то нужно ставить опцию pentium4 или pentium4m (они равнозначны)

Примечание(JohnBat26) (обновлено в 1.5): Если Вы используете компилятор версии 4.2.0 и выше, то вместо указания специфичного типа процессора, можно указывать одно из двух (в параметрах: **-march** и **-mtune**):

3. **generic:** если Вы хотите, чтобы Ваш скомпилированный код запускался на всех процессорах, архитектуры x86;
4. **native:** если Вы хотите оптимизировать код только для Вашего процессора. В этом случае компилятор будет брать сведения о процессоре путем вызова `cpuid` ! .

Выбор оптимальных параметров

Для этого есть очень интересная утилита. `emerge acovea`

Правда существующие профили рассчитаны только на pentium 3/4, и на gcc 3.3/3.4, Но в принципе добавить свою конфигурацию тоже не составляет труда. Также рекомендуется добавить в конфигурацию опции `-ftracer` и `-mfpmath=sse`. В некоторых случаях они дают значительный прирост производительности сгенерированного кода.

После чего вызываем утилиту `runacovea -config gcc33_pentium3.acovea -bench evobench.c` Ждем несколько часов и получаем оптимальные флаги компиляции.

Возможны различные тесты, которые хранятся в каталоге `/usr/share/acovea/benchmarks`, И различные конфигурации платформы `/usr/share/acovea/config`, к которым при желании можно добавить свою.

Редактор nano

nano - Nano's ANOther editor

Nano - достаточно простой и удобный текстовый редактор, клон редактора `pico`

Установка

emerge nano

Конфигурация

Файл: /etc/nanorc

Запуск

Запуск редактора
nano -w somefile

Параметр -w здесь означает запрет переноса строк.

Udev и автомонтирование носителей

Введение

В данной статье речь пойдет о том, как настроить udev для автоматического монтирования и отмонтирования usb-носителей. Ведь согласитесь, неудобно каждый раз монтировать флэшки из консоли. А когда вам нужно быстро что-то скинуть с одной флэшки на другую? Здесь я привожу одно из возможных решений.

Требования

У меня установлена следующая версия udev:
sys-fs/udev-069

Я умышленно не ставлю более новые версии, т.к. с ними возникают проблемы при загрузке firmware на моем компьютере. Возможно в более новых версиях что-то работает по-другому, но суть одна и та же.

Настраиваем udev

Создаем правила для flash-носителей

Правила udev, в соответствии с которыми устройствам назначаются имена, а также выполняются специфические действия, расположены в /etc/udev/rules.d/*. В этой директории уже есть файлы, содержащие правила, у меня их было 2:

```
# ls /etc/udev/rules.d
05-udev-early.rules
50-udev.rules
```

Такие названия им были даны неспроста. Дело в том, что udev обрабатывает

файлы в этой директории в алфавитном порядке по возрастанию. Вообще, при написании правил udev следует помнить 2 главные вещи:

Udev считывает файлы из **/etc/udev/rules.d** в алфавитном порядке.

Найдя первое подходящее правило для устройства, udev прекращает дальнейший поиск.

Давайте добавим свой файл с правилами так, чтобы он обрабатывался раньше, чем файл с правилами по умолчанию (50-udev.rules). Назовем его 10-udev-my.rules. Содержимое нового файла будет следующим:

Файл: /etc/udev/rules.d/10-udev-my.rules

First rule

```
SUBSYSTEM=="block", KERNEL=="sd*", ACTION=="add", NAME="%k",
GROUP="disk", RUN+="/etc/udev/scripts/udev-flash-mount add %k"
```

#Second rule

```
SUBSYSTEM=="block", KERNEL=="sd*", ACTION=="remove",
RUN+="/etc/udev/scripts/udev-flash-mount remove %k"
```

Теперь разберемся в том, что написали.

Первое правило (добавление устройства)

SUBSYSTEM=="block" При добавлении нового устройства в систему возникает множество событий разных классов. block - это класс для блочных устройств, который нам больше всего подходит для составления правила. Для просмотра значений всех переменных, возникающих событий используйте `udevmonitor --env T.O.` здесь мы реагируем только на событие, в окружении которого есть переменная SUBSYSTEM со значением block .

KERNEL=="sd*" Эта часть правила сравнивает имя устройства, которое было дано по умолчанию ядром, с шаблоном. В данном случае нам нужны такие имена как: sda, sda1, sdb, sdb1 и т.д.

ACTION=="add" Реагируем на событие добавления устройства. Например, usb флэшка была вставлена в разъем.

NAME="%k" Обратите внимание на то, что здесь стоит один знак равенства, а не два, как в предыдущих частях правила. Это говорит о присваивании, а не сравнении. Здесь мы присваиваем вставленному устройству имя, которое соответствует тому, что было дано по умолчанию ядром. %k это одна из переменных окружения. Подробнее о них можно прочитать в `man udev`.

GROUP="disk" Причисляем устройство к группе дисковых.

RUN+="/etc/udev/scripts/udev-flash-mount add %k" Это самая интересная часть скрипта. Здесь мы добавляем ("+=") одно из действий, которое произойдет после обработки правила. В данном случае мы запускаем скрипт (он будет рассмотрен далее) /etc/udev/scripts/udev-flash-mount и передаем ему 2 параметра. Первый указывает на то что нужно выполнить действия по добавлению и инициализации нового устройства, а второй содержит имя устройства, с которым нужно выполнять действия.

Второе правило (удаление устройства)

ACTION=="remove" Реагируем на удаление устройства.

RUN+=" /etc/udev/scripts/udev-flash-mount remove %k" Здесь мы запускаем скрипт, использовавшийся в первом правиле, но уже для выполнения действий после удаления устройства из системы. Этот скрипт будет выполнен как только ядро заметит, что устройство было удалено из системы. Это не самое лучшее решение, т.к. данные могут быть утеряны, но об этом в следующем разделе.

Пишем скрипт

Теперь напишем скрипт, который будет выполняться после добавления устройства и после его удаления. Допустим он будет размещен в /usr/bin/. Итак, от пользователя root:

```
# su
```

Создадим файл:

```
# touch /etc/udev/scripts/udev-flash-mount
```

Сделаем его исполняемым:

```
# chmod u+x /etc/udev/scripts/udev-flash-mount
```

Напишем в нем следующее: Файл: /etc/udev/scripts/udev-flash-mount
#!/bin/bash

```
LOG="/var/log/udev"
```

```
sleep 3
```

```
DEV=`echo $2 | sed -n '/^sd[a-z][1-9]\?/p`
```

```
if [ "$1" = "add" ]; then
```

```
  if [ "$DEV" != "" ]; then
```

```
    echo "--- `date` ---" >> $LOG
```

```
    echo "Mounting /dev/$DEV" >> $LOG
```

```
    mkdir /mnt/$DEV >> $LOG 2>&1
```

```
    chmod a+rwX /mnt/$DEV >> $LOG 2>&1
```

```
    mount /dev/$DEV /mnt/$DEV -o sync,umask=0000,ioccharset=cp1251 >> $LOG 2>&1
```

```
  fi
```

```
elif [ "$1" = "remove" ]; then
```

```
  if [ "$DEV" != "" ]; then
```

```
    echo "--- `date` ---" >> $LOG
```

```
    echo "Unmounting /dev/$DEV" >> $LOG
```

```
    umount /dev/$DEV >> $LOG 2>&1
```

```
    rm -rf /mnt/$DEV >> $LOG 2>&1
```

```
  fi
```

```
fi
```

Задержка в 3 или более секунд необходима для того, чтобы ядро успело выполнить свои действия по инициализации устройства. Если не лень, то поэкспериментируйте с задержкой. Если скрипту был передан параметр `add`, то он создает папку с именем устройства, затем пытается примонтировать устройство в эту папку. Весь вывод, в том числе и ошибки, выводятся в файл лога. Соответственно, при вызове скрипта с параметром `remove`, он пытается отмонтировать устройство и удалить ранее созданную папку. Предупреждение: Опция `sync` в параметрах `mount` является необходимой, т.к. форсирует мгновенную запись на устройство. Если ее не будет, то когда вы удалите устройство из системы, данные не будут записаны на него. Поэтому, чтобы избежать проблем такого рода, дождитесь пока индикатор на носителе перестанет мигать и после этого извлеките устройство. НО для файловой системы `fat` с версии ядра `linux 2.6.13` `sync` ставить нельзя т.к. функционирование данной опции сильно изменилось и максимальная скорость записи на `usb2.0` это 200 килобайт в секунду а для `usb1.0` это где то 10 - 20 килобайт в секунду. Кроме того, опция `sync` в параметрах `mount` убьет Вашу флешку с гарантией за 6 месяцев использования. См. <http://bugs.debian.org/309625>

Предупреждение: Внимание! Если какая-либо запущенная программа просматривает содержимое папки, в которую смонтировано устройство, то скрипт не сможет отмонтировать его и удалить папку. Поэтому сначала убедитесь в том, что ни одна из программ не использует эту папку.

Заключение

Последний скрипт можно немного модернизировать. Предположим, если вы монтируете свою камеру, чтобы просмотреть фотографии, то можно передать скрипту еще один параметр из `udev` правила, например, с производителем устройства. Либо можно проанализировать содержимое каталога и на основе этого выполнить какие-либо действия. Все в ваших руках, дерзайте

Оптимизация glibc

Введение

Glibc - свободная библиотека, предоставляющая системные вызовы и другие основные операторы для основных систем GNU/Linux. Если C является наиболее распространённым языком, используемым для программирования в Linux, такая же судьба постигла и `glibc` - она является продуманной частью ядра системы. `Glibc` может быть оптимизирована на вашей системе для ускорения операций различными способами. Оптимизация, естественно, зависит от вашей системы и её использования. Это также зависит от установки флагов `USE`, доступных при выполнении `emerge`. До применения рекомендаций рекомендуется узнать о

назначении и использовании флагов USE. Есть неплохая инструкция [здесь](#).

Оптимизация компиляции Glibc

Эффект CFLAGS

Компиляция Glibc с возможностями GCC используется в основном коде glibc. Glibc идентифицирует GCC как компилятор и использует установленный флаги CFLAGS. [Подробнее здесь](#). Используя правильные флаги CFLAGS для вашей системы, вы можете хорошо оптимизировать glibc.

Улучшение оптимизации

Если вы использовали `-fomit-frame-pointer`, вы можете также поместить `glibc-omitfp` во флаги USE. Повторно пересоберите glibc. Code: `Remerging glibc`

`emerge --newuse -v world`

При этом соберутся Glibc и её зависимости. При компиляции glibc этот флаг активизирует флаг `-enable-omitfp`, который, в свою очередь, указывает glibc на использование `--fomit-frame-pointer`. В результате более безопасно устанавливается указатель кадра. Этот флаг устанавливает максимальную оптимизацию glibc и предоставляет два типа библиотек - 'оптимизированная' и 'стандартная'. То есть, по умолчанию будут использоваться оптимизированные библиотеки, но при необходимости можно будет использовать 'стандартную' версию. Это увеличит размер glibc и приведет к обычному поведению при установленном `--fomit-frame-pointer`, т.е. сделает невозможным отладку (debug) программ, так что не используйте этот флаг если вы планируете заниматься разработкой программ. Теоретически этот флаг может вызвать некоторые ошибки компилятора, хотя на практике он достаточно безопасен, тем не менее вы были предупреждены.

Модели потоков

glibc поддерживает 2 различных модели потоков - старую `linuxthreads` и новую `nptl`. По умолчанию, если не был указан флаг `nptlonly`, кроме `nptl` собирается и версия с `linuxthreads`. В случае, если у вас нет устаревшего программного обеспечения, вы можете указать флаг `nptl`. Если у вас современная система без сторонних бинарных пакетов, будет лучше указать `nptlonly`, чтобы избежать сборки glibc дважды (с поддержкой `linuxthreads` и без неё).

Локализация

В основном это основано на еженедельном новостном бюллетене Gentoo от 8

ноября 2004 года, который можно найти здесь (en:TIP Specifying only needed locales). Вы можете выбрать, какие локали (включая раскладки и установки клавиатуры, времени и т.д.) будут собраны при установке. Если вы не ограничите локали, используемые в вашей системе, будут собраны все имеющиеся локали, начиная с aa_DJ (локаль Афар для Джибути) через en_GB (английская локаль для Великобритании) и заканчивая zu_ZA.utf8 (локаль Зулу для Южной Африки). При ограничении сборки локалей вы можете сохранить до 90% места, необходимого для Glibc в вашей системе, сохранить время, необходимое на сборку ненужных вам локалей и, как следствие, уменьшить общее время компиляции. Если вы действительно не нуждаетесь в них всех (а трудно представить, кому это могло бы понадобиться), вы можете ограничить их необходимым минимумом.

Используйте флаг USE userlocales для сборки только тех локалей, которые указаны в /etc/locales.build.

Отредактируйте /etc/make.conf в вашем любимом редакторе и поместите userlocales где-нибудь между "" в строке USE="".

Другой способ - изменить /etc/portage/package.use используя данную команду:
Code: Activating the userlocales USE flag for glibc
echo "sys-libs/glibc userlocales" >> /etc/portage/package.use

После этого вы можете указать локали, которые хотите использовать:

Файл: nano -w /etc/locales.build

#Читайте комментарии в начале файла для большей информации!

```
en_US/ISO-8859-1
en_US.UTF-8/UTF-8
#en_GB/ISO-8859-1
#en_GB.UTF-8/UTF-8
de_DE/ISO-8859-1
de_DE@euro/ISO-8859-15
ru_RU.UTF-8/UTF-8
```

В glibc-2.3.6-r4 или glibc-2.4-r2, был удалён флаг userlocales. Вы должны править файл /etc/locale.gen и удалить /etc/locales.build.

Вы можете воспользоваться для этого следующими командами: Code: Convert locales.build to locale.gen

```
cd /etc
grep '^[^#].*' locales.build | sed 's/: : ' > locale.gen
rm locales.build
nano -w locale.gen
Файл: nano -w /etc/locale.gen
```

#Читайте комментарии в начале файла для большей информации!

```

en_US.UTF-8 UTF-8
en_US ISO-8859-1
ru_RU.UTF-8 UTF-8
de_DE ISO-8859-1
de_DE@euro ISO-8859-15

```

Вы можете найти правильные значения локалей в файле `/usr/share/i18n/SUPPORTED`.

Помогите! Я не знаю, какими должны быть настройки моей локали!

Не волнуйтесь, это просто. Записи в этом файле должны быть в формате `<locale>/<charmap>`. `<locale>` это локаль из каталога `/usr/share/i18n/locales`, а `<charmap>` -- это имя одного из файлов каталога `/usr/share/i18n/charmaps/`. Нужно только понимать два момента: первое, локаль с новой валютой, такой как евро, требует изменения `@euro`, описанного в примере выше. Второе, локали UTF-8 требуют от вас добавления `.UTF-8` в конец определения локали (нет, я не знаю, зачем, но иначе оно не работает).

Если вы В САМОМ ДЕЛЕ хотите минимизировать ваши локали, вам всего лишь нужно перевести вашу систему на Юникод. Вообще-то пока есть причины этого не делать, но это уже становится стандартом *de facto*, так как несёт множество преимуществ перед старыми форматами ISO, ASCII и другими. Больше об этом можно прочесть [здесь](#) и [здесь](#), а также непосредственно в соответствующем руководстве с сайта Gentoo.

Вам просто нужно определить следующее: Файл: `nano -w /etc/locales.build`

#Как раньше!

```

en_US.UTF-8/UTF-8
ru_RU.UTF-8/UTF-8

```

Обратите внимание, я оставил строку `US UTF-8`. Её лучше не убирать, так как некоторые программы не соберутся без неё.

Сделайте это до установки Gentoo

Вы можете выполнить эти настройки до начальной сборки системы с нуля. Просто следуйте инструкциям из *Gentoo Handbook*. Это позволит меньше волноваться о возможных недоделках программ после сборки и наслаждаться быстрой сборкой.

Сделайте это после установки Gentoo

Если glibc уже установлена, вам потребуется пересобрать glibc:

Code: Пересборка glibc

emerge glibc

Вы также можете пересобрать world, если пакеты были скомпилированы с поддержкой множества локалей, но полученное ускорение работы пакетов и сэкономленное дисковое пространство не стоят затрат времени. Придётся затратить очень много времени на пересборку всей системы.

Возможно, вам стоит обратить внимание на пакет localepurge, который может очистить систему от любых установленных man страниц или info-файлов на языках не используемых в вашей системе. Вначале прочитайте man к localepurge, что бы определить локали, которые необходимо пропустить в /etc/locale.nopurge.

Для большей информации об управлении локалями читайте:

[Gentoo Linux Localization Guide](#).

Или на русском языке [Руководство по русской локализации Gentoo Linux](#).

Эта оптимизация полностью безопасна (Кроме случаев, если вы не можете читать на языке, который установили!).

Безопасность

Как безопасность относится к оптимизации? Итак, это оптимизация безопасности, поскольку ранее было сказано, как оптимизация зависит от вашей системы и как всё это работает. Если это сервер, вы, к примеру, можете пожертвовать оптимизацией в угоду безопасности, или выполнить обе оптимизации.

Стабилизация Glibc

Это предполагает (как вы, возможно, догадывались...) использование флага USE hardened. Это сделает вашу систему более устойчивой к нападениям разного вида.

Усиление стека

Используйте флаг erandom если вам необходимо использование модуля случайных чисел, установленного в вашем ядре.

Подключение USB-flash

Подключение USB-flash к компьютеру

Введение

В Linux я пользователь новый и с этим вопросом никогда не сталкивался, и более того - ядро по жизни собирал без поддержки USB устройств, т.к. ничего такого у меня просто нет :), но вот, как говорится - пришлось и срочно. И как оказалось - это довольно несложно.

Итак, по прочтении парочки-тройки вопросов связанных с USB, и небольшого количества теории - я без проблем подмонтировал USB флешку. Примечание: Что такое флешка и USB описывать не буду, т.к. если читатель этого не знает - то имхо ему следует сначала прочесть несколько других статей.

Итак, начнем

Конфигурация ядра

USB флешки видны в Linux как scsi устройства, ибо поддержка usb устройств в ядре Linux реализуется именно посредством эмуляции скази, т.е. типа /dev/sda1(2,3) и т.д, смотря сколько партиций.

Следовательно, чтобы была возможность подключения флешки, сначала необходимо добавить поддержку scsi в ядро

Linux Kernel Configuration: Включаем поддержку SCSI

Device Drivers --->

SCSI device support

<*> legacy /proc/scsi/ support

--- SCSI support type (disk, tape, CD-ROM)

<*> SCSI disk support

<*> SCSI generic support

--- Some SCSI devices (e.g. CD jukebox) support multiple LUNs

<*> Probe all LUNs on each SCSI device

Далее добавляем непосредственно поддержку USB (я не говорю о клавиатурах или других устройствах USB, а говорю только о флешке)

Linux Kernel Configuration: Поддержка USB

Device Drivers --->

USB support

<*> Support for Host-side USB

<*> USB device filesystem

<*> EHCI HCD (USB 2.0) support

< > OHCI HCD support

<*> UHCI HCD (most Intel and VIA) support

<*> USB Mass Storage support

Примечание: В разделе USB Mass Storage support есть куча других устройств - я там не отмечал ничего, у меня заработало и так. Подозреваю, что для специфических устройств (в смысле каких то навороченных флешек) нужно будет что нибудь еще там отметить.

Кроме того, для этих всяких сложных флешек скорее всего придется добавить MTD в ядро

Linux Kernel Configuration: MTD

Device Drivers --->

Memory Technology Devices (MTD) --->

ну и там смотрим, что нужно. Примечание: Для большинства современных флэшек, USB носителей и цифровых фотоаппаратов поддержка MTD не требуется.

Монтирование

После загрузки с новым ядром, нажимаем Alt+F12 и вставляем флешку - если с ядром все ок- то сразу появятся сообщения, что мол есть устройство USB , потом пишется, что есть /dev/uba ну и т.п.

Если этого не происходит, делаем

dmesg | grep usb

Если и здесь ничего нет, то советую перегрузиться и посмотреть включены ли USB контроллеры в BIOS :) как это не смешно звучит, но я сам себе это отрубил, когда ядро собирал, а потом включить забыл, и минут 15 гадал - чего же оно не работает :)

Если **dmesg | grep usb** выдает результат, а когда вставляем флешку - никакой реакции, то нужно проверить еще раз - все ли отмечено в ядре.

Вобщем - надеемся , что все заработало и теперь нужно смонтировать флешку.

Добавляем в /etc/fstab строку

Файл: /etc/fstab

```
/dev/uba1 /mnt/usbdisk auto user,noauto,rw 0 0
```

и можно монтировать! но есть одно но, в опциях я не указал кодировки, т.к. кодировка по умолчанию указана в разделе Native Languages конфигурации ядра.

Не будет лишним напомнить отмонтировать флешку перед тем как вытаскивать ее!

Если такой вариант вас не устраивает, то вы можете установить пакет submount и монтировать флешку вот так:

Файл: /etc/fstab

```
/dev/uba1 /mnt/usb subfs fs=vfat,auto,umask=0,quiet,sync 0 0
```

Параметр sync необходим чтобы отключить отложенную запись и избежать потерю данных при копировании файлов на флешку.

Вроде бы все. Удачи всем!

И естественно- за любые комментарии, дополнения, уточнения и исправления ошибок – буду только благодарен :)

надеюсь, кому-то помог tradakad

исправил касательно монтирования при помощи submount (dernik)

Монтирование под kde

Установите пакеты: **hal**, **dbus**, and **hotplug**

Добавьте флаг hal в /etc/make.conf

emerge -avt kdebases-kioslaves

Убедитесь, что hal используется всеми пакетами:

emerge -DNu world

rc-update add dbus default

/etc/init.d/dbus start

rc-update add hald default

/etc/init.d/hald start

так же можно монтировать устройства без изменения /etc/fstab с помощью

emerge -av pmount

Добавьте себя в групп plugdev

gpasswd -a USER plugdev

Включите "Storage media" ("Устройства хранения данных" в русском варианте) в systray (на панель). Наслаждайтесь :)

Автоматическое монтирование

Если лень замарачиваться каждый раз при втыкании флэшки или сидюка, а хочется, чтоб как в одной другой ОС, добавьте параметр managed в /etc/fstab к требуемому устройству, например

Файл: /etc/fstab

```
/dev/cdrw /mnt/cdrom iso9660 user,noauto,ro,managed 0 0
```

```
/dev/sda /mnt/flash vfat exec,user,noauto,sync,managed 0 0
```

Типа, все, теперь при появлении флэшки в системе будет появляться иконка на рабочем столе.

ps. распишите эту статью подробнее. Добавьте gpasswd в статью:

http://gentoo-wiki.com/HOWTO_D-BUS,_HAL,_KDE_media/

Чтение-запись NTFS с использованием драйвера ntfs-3g

Введение

"Драйвер ntfs-3g является открытым, лицензированным по GPL, драйвером NTFS для Linux, созданным в рамках проекта Linux-NTFS. Он предоставляет полный доступ к разделам NTFS (чтение-запись), кроме работы с зашифрованными файлами и записи сжатых файлов. Так же не поддерживается смена владельца файла и его прав доступа. Технически он является сильно усовершенствованным драйвером ntfsmount. Улучшена функциональность, качество и добавлены дополнительные возможности." (Szakacsits Szabolcs, автор ntfs-3g)

Автор создал драйвер, который гораздо лучше пригоден для использования, чем другие драйвера NTFS для Linux, и при этом не менее быстрый, чем драйвера родных файловых систем Linux - временами в два раза быстрее чем родной модуль файловой системы EXT3 в ядре.

Для получения более подробной информации посетите: [1]

Предупреждение: Драйвер имеет статус BETA, имеются некоторые не решённые вопросы, возможна потеря данных и/или зависания при попытке примонтировать раздел. Он НЕ работает на архитектурах, отличных от x86 и amd64.

СДЕЛАЙТЕ РЕЗЕРВНУЮ КОПИЮ ДАННЫХ ПЕРЕД ЛЮБЫМИ ПОПЫТКАМИ ИСПОЛЬЗОВАНИЯ !!! В ЛЮБОМ СЛУЧАЕ - ВЫ ПРЕДУПРЕЖДЕНЫ !!!

Установка

ebuild драйвера зависит от sys-fs/fuse (portage установит его автоматически).

Поскольку это программное обеспечение находится на стадии бета-тестирования, вы должны размаскировать его:

```
# echo "sys-fs/ntfs3g ~x86" >> /etc/portage/package.keywords
```

И установить его:

```
# emerge sys-fs/ntfs3g
```

Модуль ядра fuse должен быть загружен перед использованием драйвера

```
# modules-update
```

```
# modprobe fuse
```

Примеры использования

Монтирование раздела:

```
# ntfs-3g /dev/hda1 /mnt/windows
```

Полный доступ для всех пользователей, без ограничений прав доступа, и поддержкой заданной локали (В примере используется ru_RU.utf8, однако у вас она может отличаться):

```
# ntfs-3g /dev/hda1 /mnt/windows -o silent,umask=0,locale=ru_RU.utf8
```

Размонтирование

```
# fusermount -u /mnt/windows
```

Прочтите страницу руководства по ntfs-3g для получения более подробной информации.

```
$ man ntfs-3g
```

Монтирование при загрузке:

Для автоматической загрузки модуля fuse при старте системы выполните:

```
# echo "fuse" >> /etc/modules.autoload.d/kernel-2.6
```

Отредактируйте /etc/fstab:

```
# echo "/dev/hda1 /mnt/windows ntfs-3g silent,umask=0,locale=ru_RU.utf8 0 0" >> /etc/fstab
```

Вопрос обновления ядра

Если после обновления ядра драйвер перестал работать, сделайте следующее:

```
# emerge sys-fs/fuse
```

```
# modules-update
```

```
# modprobe fuse
```

Отказ от гарантий

Автор этого документа не отвечает за какие либо последствия, связанные с использованием этого драйвера. Всё что вы делаете - вы делаете на свой страх и риск.

Сетевые сервисы

Установка Apache2

Установка

Apache2 сейчас работает на большинстве системах без проблем, если у вас возникли проблемы прочтите раздел Common Problems это может помочь.

Начнем установку apache2: добавьте "apache2" к вашим USE флагу в файле `/etc/make.conf` и выполните:

emerge apache

Запуск Apache

Запустите Apache2:

/etc/init.d/apache2 start

Если вы хотите запускать apache2 при запуске системы выполните команду:

rc-update add apache2 default

Посмотрите init scripts section для дополнительной информации.

Теперь у вас есть работающий веб сервер Apache2. Перейдите в вашем браузере на страницу **<http://localhost/>** и вы должны увидеть страницу приветствия.

Вы найдете в `/var/www/localhost/htdocs/index.html` HTML код страницы приветствия которую вы видите на `http://localhost/`. Замечаем что Apache отображает `htdocs/index.html` когда вы пытаетесь открыть `htdocs/`. Это особенность протокола HTTP. Apache не может передать директорию но может показать содержимое. Список (index) файлов в директории может быть передан. Apache ищет страницу со специальным именем `index` либо генерирует список файлов в директории. Если у вас есть страница с именем `'index'` то будет отображена она; в этом случае вы не сможете посмотреть список файлов в директории.

Модули

Apache очень гибок. Он может сёрфить файлы используя HTTP либо сёрфить файлы и помощью FTP. Он может передать файл с жесткого диска, либо вывод PHP скрипта. Для реализации этого Apache использует модули. Другие приложения используют для этого плагины. При добавлении модули добавляют функциональность. Их также можно устанавливать, удалять, пересобирать (перекомпилировать).

Модули Apache обычно называются `mod_something`. Некоторые уже включены в Апахь а некоторые нужно добавлять отдельно. В портежах содержатся множество модулей. Процесс установки выглядит примерно так:

1. **emerge module ;**
2. отредактируйте /etc/conf.d/apache2 для активации добавьте -D MOD ;
3. опционально /etc/apache2/modules.d/xy_module ;
4. добавьте директивы конфигурации в httpd.conf либо .htaccess ;

emerge mod_perl

nano /etc/conf.d/apache2

change APACHE_OPTS="" to APACHE_OPTS="-D PERL"

Документацию по конкретным модулям вы найдете [Apache Index in this wiki](#). Вы также можете почитать the documentation для дополнительной информации о модулях Apache.

Конфигурация

В файле httpd.conf, (/etc/apache2/httpd.conf,) который поставляется с Gentoo хранятся большинство настроек Apache. However, it probably does both more and less than you need it to. Apache configuration files have a consistent syntax.

Любые строки начинающиеся с # игнорируются

Apache не анализирует написанное здесь

это комментарии

Некоторые строки начинаются с директивы и могут иметь один или несколько аргументов.

Директивы могут быть объединены в секции. Разделы обычно заключены в угловые скобки.

```
<Section>
```

```
# Will only apply when the section matches
```

```
AnotherDirective
```

```
</Section>
```

В разделе могут быть подразделы. Вот часть файла httpd.conf:

```
# If mod_alias is loaded
```

```
<IfModule mod_alias.c>
```

```
# Alias is a directive and it only applies if mod_alias is loaded
```

```
Alias /icons/ "/usr/share/httpd/icons/"
```

```
# If the file is in the directory
```

```
<Directory "/usr/share/httpd/icons">
```

```
# Options will only apply if:
```

```
# mod_alias is loaded AND
```

```
# the file is in the directory
```

```
Options Indexes MultiViews
```

```
</Directory>  
</IfModule>
```

Вы можете прочитать подробнее configuration files и sections в официальной документации Apache.

Common Problems

SSI Not Working

When configuring for SSI (Server Side Includes), an error may occur:

mod_include: Options +Includes (or IncludesNoExec) wasn't set, INCLUDES filter removed

The problem is that setting Options +Includes in either .htaccess or httpd.conf is overwritten by the additional configuration file as defined at the end of httpd.conf.

Include /etc/apache2/vhosts.d/*.conf

You need to edit this additional configuration file such that

AllowOverride None

Is replaced by

AllowOverride Options

Could Not Open Error Log

While starting Apache, it prints:

Error while starting apache: ***(2)No such file or directory: apache2: could not open error log file /usr/lib/apache2/logs/error_log.***

/usr/lib/apache2/logs should be a symlink pointing to /var/log/apache2 . Check it using:

ls -la /usr/lib/apache2/logs

(note the lack of a slash on the end). If /var/log/apache2 is missing, create it and make sure you give apache ownership:

mkdir /var/log/apache2

chown apache:apache /var/log/apache2

If the symlink /usr/lib/apache2/logs is missing, you can create it:

ln -s /var/log/apache2 /usr/lib/apache2/logs

You don't need to set permissions on the symlink.

Check the Logs

See `/var/log/apache2/error_log` for errors, especially towards the end of the file. You may find `tail` useful because it displays only the last few lines of a file:

`tail /var/log/apache2/error_log`

If you wish to keep an eye on the log the `-f` option for `tail` may be useful:

`tail -f /var/log/apache2/error_log`

Here's one error you might see:

Error: [alert] (EAI 2)Name or service not known: mod_unique_id: unable to find IPv4 address of ""

With the base installation "mod_unique_id" is turned on, this can cause problems, notably the server not starting. Simply comment out this module in `/etc/apache2/httpd.conf` and the problem will be solved.

(Your config file might be `/etc/apache2/conf/apache2.conf`)

Forbidden User Directories

If the server is returning "403 Forbidden" while accessing `http://server/~username/` Make sure Apache (usually user `apache` and group `apache`) has read access to `username's` home directory and `public_html` (or equivalent). You can grant everyone read access using:

`chmod 755 ~username/ ~username/public_html/`

Not Enough Entropy

If Apache2

accepts connections

does not respond to clients

creates exactly one process

is not stopped by

`/etc/init.d/apache2 stop`

Check to see how much entropy is available using:

`cat /proc/sys/kernel/random/entropy_avail`

If little entropy (less than 100) is available, Apache2 is probably waiting for more so it can generate the secret for digest authentication (`mod_auth_digest`). To generate more entropy, just do something else for a little while. Grepping the kernel or emerging a

package usually works well.

The video-entropyd and audio-entropyd supply /dev/random with entropy gathered from your video and audio devices, respectively. If you have a hardware random number generator (RNG), you can emerge rng-tools and run rngd.

If there's still a shortage of entropy, you can enable the urandom USE flag and re-emerge APR and Apache2. This makes APR use /dev/urandom, which falls back to a pseudorandom number generator when there isn't enough entropy. The program gets a number immediately, but it is cryptographically weaker. This is okay for some things (e.g. solitaire), but completely unacceptable for others (like PGP key generation).

Confusing config files

If you start the Apache2 server with the startup script /etc/init.d/apache2 check to see if the line

```
local myconf="/etc/apache2/httpd.conf"
```

from /etc/init.d/apache2 points to your configuration script. If it points to apache.conf and you use httpd.conf, make the necessary adjustments.

See Also

Configure LAMP (Linux, Apache, MySQL, and Python/PHP/Perl) - A popular web server combination

Apache Installation & Configuration

How to install mod_security for Apache

Настройка iptables для начинающих

Введение

В сети мало документации по [iptables](#) рассчитанной на новичков. Мы же попытаемся восполнить этот пробел. Рассмотрим основы составления правил, а также некоторые дополнительные модули которые помогут сделать жизнь легче.

Прежде чем двигаться дальше - убедитесь, что ...

1. Всё ваше аппаратное обеспечение работоспособно. То есть Вы подключили все оборудование, модули грузятся, устройства видны в системе. Полезно в

начале проверить, что соединение с интернетом возможно и без всяких там [iptables](#). Нет ничего хуже, чем в течении многих часов возиться с упрямой программой, ругая её почём зря, а потом понять, что сетевая карта не работает или модем сконфигурирован неправильно.

2. Вы имеете представление о сетевых технологиях и Вы знакомы с администрированием Linux и Gentoo Linux в частности. То есть необходимы навыки работы с такими базовыми вещами как [ifconfig](#), [rc-update](#), [/etc/conf.d/net](#), и так далее. Если для Вас это пока пустые звуки, то, пожалуйста предварительно изучите [Настольную книгу Gentoo](#) и [Linux Help's Networking Basics 101](#)

Конфигурация ядра ОС Linux

Все что вам нужно - это включить поддержку [iptables](#).

```
Networking --->
Networking Options---->
Network Packet Filtering (replace Ipchains)--->
Netfilter Configuration
```

Включим все опции как модули (хотя с точки зрения безопасности модули следует вообще отключить, монолитное ядро надежнее, хотя и медленнее).

Установка iptables

Далее вы должны установить пакет [iptables](#): `emerge iptables`

Проверка сети

Предположим, что у нас есть 2 сетевых интерфейса: `eth0` - локальная сеть и `ppp0` - интернет соединение.

Проверим работоспособность сети командой [ping](#):

Code: [ping](#)

```
ping www.google.com
ping 192.168.1.78
ping 192.168.2.77
```

Запуск iptables

Запустим [iptables](#): `/etc/init.d/iptables start`

Эта команда загрузит основные модули и создаст цепочки в ядре Linux. Теперь добавим [iptables](#) в автозагрузку: `rc-update add iptables default`

Использование /etc/init.d/iptables

Скрипт `/etc/init.d/iptables` понимает несколько команд (`/etc/init.d/iptables <команда>`), некоторые из них:

- **start** - запуск [iptables](#). Восстанавливает все правила (правила хранятся в `/var/lib/iptables/rules-save`);

- **stop** - сброс всех цепочек;
- **save** - сохранение всех правил.

Создание правил

Практически все правила можно привести к виду: `iptables -A ЦЕПОЧКА ПАРАМЕТРЫ_ПАКЕТА -j ДЕЙСТВИЕ`

Цепочки

Все изменения будем проводить над таблицей *filter*, именно она отвечает за фильтрацию пакетов. В таблице *filter* существует 3 цепочки: **INPUT**, **OUTPUT** и **FORWARD**. В каждой цепочки свой "тип" пакетов:

- **INPUT** - пакеты пришедшие к Вам. То есть входящий трафик.
- **FORWARD** - пакеты которые предназначены для другого узла, то есть транзитный трафик.
- **OUTPUT** - пакеты, которые уходят от нас, или исходящий трафик.

Работают с цепочками так: `iptables <опция> <цепочка>`

Для работы с цепочками предусмотрены следующие опции:

- **-A** - добавление нового правила в цепочку. Правило будет добавлено в конец цепочки.
- **-I** - добавление правила не в конец, а туда куда вы укажете. Например команда:

`iptables -I INPUT 2 bla-bla-bla` - сделает наше правило вторым.

- **-D** - удаление правила. Например для удаления пятого правила введите:

`iptables -D INPUT 5`

- **-F** - сброс всех правил цепочки. Нужно, например, при удалении ненужной цепочки.
- **-N** - создание пользовательской цепочки. Если не хотите создавать кашу в каждой цепочке, то создайте несколько дополнительных цепочек. Синтаксис такой: `iptables -N ЦЕПОЧКА`. Только русские буквы, конечно, использовать нельзя.
- **-X** - удаление пользовательской цепочки.

ПРИМЕЧАНИЕ: Удалить цепочки **INPUT**, **OUTPUT** и **FORWARD** нельзя.

- **-P** - установка политики для цепочки. Например:

`iptables -P ЦЕПОЧКА ПОЛИТИКА`

Параметры пакетов

Итак по каким параметрам можно фильтровать пакеты? Рассмотрим самые основные.

Источник пакета

Для фильтрации по источнику используется опция **-s**. Например запретим все входящие пакеты с узла 192.168.133.133: `iptables -A INPUT -s 192.168.133.133 -j DROP`

Можно использовать доменное имя для указания адреса хоста. То есть: `iptables -A INPUT -s test.host.jp -j DROP`

Также можно указать целую подсеть: `iptables -A INPUT -s 192.168.133.0/24 -j DROP`

Также вы можете использовать отрицание (знак !). Например так - все пакеты с хостов отличных от 192.168.133.156 будут уничтожаться: `iptables -A INPUT -s ! 192.168.133.156 -j DROP`

Адрес назначения

Для этого нужно использовать опцию **-d**. Например запретим все исходящие пакеты на хост 192.168.156.156: `iptables -A OUTPUT -d 192.168.156.156 -j DROP`

Как и в случае с источником пакета можно использовать адреса подсети и доменные имена. Отрицание также работает.

Протокол

Опция **-p** указывает на протокол. Можно использовать *all*, *icmp*, *tcp*, *udp*.

Порт источника

Указывает на порт с которого был прислан пакет. Вот синтаксис: `iptables -A INPUT -p tcp --sport 80 -j ACCEPT`

Для указания порта необходимо указать протокол (**tcp** или **udp**). Можно использовать отрицание.

Порт назначения

Порт назначения. Синтаксис: `iptables -A INPUT -p tcp --dport 80 -j ACCEPT`

Как и в случае с портом источника нужно указать протокол. Можно использовать отрицание.

Действия над пакетами

Проку от того, что мы укажем параметры пакета нет. Нужно указать, что надо с ним делать. Для этого служит опция **-j**. Рассмотрим основные действия:

- **ACCEPT** - разрешить пакет.
- **DROP** - уничтожить пакет.
- **REJECT** - будет отправлено ICMP сообщение, что порт недоступен.
- **LOG** - информация об этом пакете будет добавлена в системный журнал ([syslog](#)).

В качестве действия можно указать и имя пользовательской цепочки. Например перекинем все пакеты с локальной сети в цепочку, где будет производиться дополнительная проверка: `iptables -A INPUT -s 192.168.200.0/24 -j LOCAL_NET`

Пример правил

В большинстве случаев пользователю достаточно выполнить такую последовательность команд: `iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT` `iptables -A INPUT -i lo -j ACCEPT` `iptables -P INPUT DROP`

Вот собственно и вся настройка. На первый взгляд непонятно, что мы тут вообще сделали. Поэтому ознакомимся с основами составления правил.

Модули

Может сложиться впечатление, что возможностей у [iptables](#) маловато. Однако с использованием модулей [iptables](#) получит просто безграничные возможности. Для указания модуля используется опция **-m**. Например: `iptables -A INPUT -m модуль bla-bla`

-m owner

Добавляет следующие опции (опции только для цепочки OUTPUT):

- **--uid-owner UID** - UID программы пославшей пакет.
- **--gid-owner GID** - GID программы пославшей пакет.

Следующие опции доступны только в версиях ядра **ниже 2.6.14**:

- **--pid-owner PID** - PID программы пославшей пакет.
- **--sid-owner SID** - SID (идентификатор сессии) производится проверка SID пакета, значение SID наследуются дочерними процессами от "родителя".
- **--cmd-owner NAME** - имя программы пославшей пакет.

-m multiport

Позволяет указывать не по одному порту, а сразу несколько:

- **--source-ports порт1,порт2** - список портов, с которых пришел пакет;
- **--sports порт1,порт2** - укороченный аналог **--source-ports**;
- **--destination-ports порт1,порт2** - список портов назначения;
- **--dports порт1,порт2** - укороченный аналог **--destination-ports**;
- **--ports порт1,порт2** - проверяет как исходящий так и входящий порт пакета.

-m state

Предназначен для указания состояния пакета с помощью опции **--state**. Доступны следующие типы пакетов:

- **NEW** - пакет устанавливающий новое соединение.
- **ESTABLISHED** - пакет от уже установленного соединения.
- **RELATED** - новый пакет уже установленном соединении.

-m mac

Проверяет соответствие MAC-адреса в пакете с помощью опции **--mac-source**, например:

```
iptables -A INPUT -s 192.168.0.1 -m mac --mac-source 00:65:3F:ED:12:98 -j DROP
```

Получено с http://ru.gentoo-wiki.com/Настройка_iptables_для_начинающих

Подробная настройка iptables

Введение

Документация по **iptables** в сети не рассчитана на новичков. В этой статье будет сжато и быстро описаны команды, затем, возможно, будут правки и добавления для расширенного объяснения. Так что это будет минимальная установка которую мы в дальнейшем расширим и упрочим с помощью правил.

Так же примите во внимание, что будет использовано рррое соединение и 2.6.x ядро. Для настройки сетевой карты надо будет заменить **ppp0** на **eth0** (или подходящий по смыслу ваш сетевой интерфейс глядящий в интернет)

Прежде чем двигаться дальше - убедитесь, что ...

- 1.Всё ваше аппаратное обеспечение работоспособно. То есть вы все подключили, модули грузятся, устройства видны в системе. Полезно вначале проверить, что соединение с интернетом возможно и без всяких там iptables. Нет ничего хуже, чем в течение многих часов возиться с упрямой программой, ругая её почём зря, а потом понять, что сетевая плата не работает или модем сконфигурирован неправильно.
- 2.Вы читали ману по теме. Предполагается, что, пока вы изучаете это руководство, man iptables постоянно открыт в соседнем терминале для точного понимания и уточнения, что же та или иная команда означает на самом деле.
 - 3.Вы имеете представление о сетевых технологиях и Вы знакомы с администрированием Linux и Gentoo Linux в частности. То есть необходимы навыки работы с такими базовыми вещами как [ifconfig](#), [rc-update](#), [/etc/conf.d/net](#), и так далее. Если для вас это пока пустые звуки, то, пожалуйста предварительно изучите [The Gentoo Handbook](#) и [Linux Help's Networking Basics 101](#)

Конфигурация ядра

Все что вам нужно - это включить поддержку iptables.

Linux Kernel Configuration: Включение IPTables

```
Device Drivers--->
Networking Support--->
Networking Options----->
Network Packet Filtering (replace Ipchains)--->
Netfilter Configuration
```

Я включил все опции как модули (с тем расчетом, что я захочу попробовать другие опции позже) и добавил ip_tables в modules.autoload. Это загрузит еще несколько модулей в качестве зависимостей. Модуль ip_conntrack необходим для "statefull" фильтрования, то есть для отслеживания соединений. Для запуска

скриптов выполните команду :

```
# modprobe ip_tables
```

Необходимые утилиты

Далее вы должны установить пакет iptables:

```
# emerge iptables
```

Конфигурация интерфейсов

Сетевые протоколы

Прикладной уровень	HTTP , SMTP , SSH , SNMP , FTP , NNTP , NTP , LDAP , ...
Сеансовый уровень	TLS , SSL , RPC , WSP ...
Транспортный уровень	TCP , UDP , SCTP , ICMP , OSPF , RSVP , VRRP , RTP , DCCP ...
Сетевой уровень	IPv4 , IPv6 , ARP , RARP , MPLS , IPX ...
Канальный уровень	Ethernet , 802.11 , xDSL , Fibre Channel , FDDI , ATM , ISDN ...

В моем случае имеется 3 сетевых адаптера. Один подключен к [WAN](#) через рррое. Другие два - к моей внутренней сети. Для того, чтобы не было проблем с iptables и маскардингом ([NAT](#)'ом), они должны быть сконфигурированы для различных подсетей. Для примера, 2 сетевых адаптера подключены к моим внутренним компьютерам (внутренние сетевые интерфейсы). Им присвоены IP-адреса: 192.168.1.1 и 192.168.2.1.

Следует заметить, что будет лучше если подключать эти внутренние адаптеры в любое сетевое устройство, такие как свитч и хаб. Для рррое подключений мы должны убедиться, что сетевой адаптер подключен к внешнему миру, то есть внешним интерфейсам не присвоены никакие IP-адреса. Его запись в [/etc/conf.d/net](#) должна оставаться пустой. Это делается потому, что рррое выступает в качестве виртуального устройства, которое включается вслед за сетевым интерфейсом. Мы также должны присвоить правильные сетевые маски и широковещательные адреса для этих интерфейсов. Ваш [conf.d/](#) должна выглядеть примерно так:

Сервер

Файл: [/etc/conf.d/net](#)

```
# Для рррое подключений вы не должны указывать значения для [[eth0]],
# просто добавьте net.ppp0 или rc-pppoe в default уровень загрузки.
iface_eth0="192.168.1.1 broadcast 192.168.1.255 netmask 255.255.255.0"
iface_eth1="192.168.2.1 broadcast 192.168.2.255 netmask 255.255.255.0"
```

Заметьте, что не было указано никаких шлюзов.

Клиент1

Файл: [/etc/conf.d/net](#)


```
iface_eth0="192.168.1.77 broadcast 192.168.1.255 netmask 255.255.255.0"
gateway="eth0/192.168.1.1"
```

Клиент2

Файл: /etc/conf.d/net

```
iface_eth0="192.168.2.77 broadcast 192.168.2.255 netmask 255.255.255.0"
gateway="eth0/192.168.2.1"
```

Шлюз для клиентов установлен на внутренний IP сетевого интерфейса сервера, что и логично. Теперь добавьте все интерфейсы в default уровень загрузки и перезапустите подключения:

```
# rc-update add net.eth1 default && rc-update add net.eth2 default && rc-update add net.ppp0 default
```

и

```
# /etc/init.d/net.eth1 start && /etc/init.d/net.eth2 start && /etc/init.d/net.ppp0 start
```

- Для клиентов:

```
# /etc/init.d/net.eth0 restart
```

Проверка настроек

Теперь убедитесь в том, что ваш сервер подключен к интернету, а также все интерфейсы могут пинговать друг друга. Для сервера:

Code: ping

```
ping www.google.com;
ping 192.168.1.78
ping 192.168.2.78
ping 192.168.1.77
ping 192.168.2.77
```

Убедитесь что у клиентов правильно указаны [DNS](#)-сервера в [/etc/resolv.conf](#)

Scripting

Теперь интересная часть... iptables и NAT(трансляция адресов). Для начала сделаем простое перенаправление адресов с минимальными правилами, чтобы убедиться что можем выходить в сеть.

Предупреждение: Если вы параноик, то это не самое секретное, что можно сделать... мы открываемся в сеть с мизерной защитой. Однако будем считать, что сеть настраиваем для дома или для игрового класса.

Файл: /var/lib/iptables/rules-save

```
#!/bin/bash
```

```
IPTABLES='/sbin/iptables'
```

```
# Определяем интерфейсы
```

```

EXTIF='ppp0'
INTIF1='eth1'
INTIF2='eth2'

# Включаем форвардинг ip в ядре.
/bin/echo 1 > /proc/sys/net/ipv4/ip_forward

# Сбросить правила и удалить цепочки
$IPTABLES -F
$IPTABLES -t nat -F
$IPTABLES -t mangle -F
$IPTABLES -X
$IPTABLES -t nat -X
$IPTABLES -t mangle -X

# Включаем маскардинг для разрешения доступа в интернет
$IPTABLES -t nat -A POSTROUTING -o $EXTIF -j MASQUERADE

# Форвардить сетевой трафик с $INTIF1 на интернетовский интерфейс $EXTIF
$IPTABLES -A FORWARD -i $INTIF1 -o $EXTIF -m state --state NEW,ESTABLISHED -j ACCEPT

# Форвардить сетевой трафик с $INTIF2 на интернетовский интерфейс $EXTIF
$IPTABLES -A FORWARD -i $INTIF2 -o $EXTIF -m state --state NEW,ESTABLISHED -j ACCEPT

#echo -e "          - Разрешаем доступ к SSH серверу"
$IPTABLES -A INPUT --protocol tcp --dport 22 -j ACCEPT
#echo -e "          - Разрешаем доступ к HTTP серверу"
$IPTABLES -A INPUT --protocol tcp --dport 80 -j ACCEPT
# Блокируем все прочие попытки доступа на $EXTIF
$IPTABLES -A INPUT -i $EXTIF -m state --state NEW,INVALID -j DROP
$IPTABLES -A FORWARD -i $EXTIF -m state --state NEW,INVALID -j DROP

```

Проверка на работоспособность

Примечание: Этот скрипт написан кем-то в сетевом форуме... Адрес автора утерян, имя... В общем, спасибо ему, пусть и без имени. (Надеюсь, он не обидется).

Теперь проверяем могут ли наши пользователи выйти в интернет или подключиться к серверу по ssh. Если все нормально, то можно переходить ниже по тексту. Если нет - проверьте синтаксические ошибки и прочее... Удостоверьтесь, что IP-адреса и маски клиентов и сервера введены правильно... Ну или...

Если все работает, как задумали, сохраняем конфигурацию:

```
# /etc/init.d/iptables save
```

И бэкапим вашу рабочую конфигурацию для возможного восстановления "как было":

```
# cp /var/lib/iptables/rules-save /var/lib/iptables/rules.working
```

Проверим iptables start-up скрипт перед тем как добавить iptables в default runlevel:

Code: Проверка скрипта

1. /etc/init.d/iptables start
2. /etc/init.d/iptables stop
3. /etc/init.d/iptables start

Смысл в запуске-остановке-запуске в том, что у нас нет скрипта запуска iptables... поэтому нужно “инициализировать” статус перед остановкой. Остановка, по существу, обнуляет настройки и возвращает все к исходному. Перезапуск покажет нам работает ли наша сеть после перезагрузки. Если все в порядке, то добавляем iptables в default runlevel:

```
rc-update add iptables default
```

Не забудем также установить в /etc/sysctl.conf:

```
net.ipv4.ip_forward = 1
```

Оборона Firewall

Далее мы сделаем наш, уже работающий фаервол безопасным, т.е. защищающим нашу систему от проникновений извне. На самом деле нам придется настроить фаервол таким образом, чтобы он не только защищал нас, но и защищал внешнюю сеть от нас. :) Защита такого рода является обязательной, она нужна для того, чтобы, в том случае, если наша система все-таки была взломана, взломщик не смог воспользоваться нашими ресурсами для дальнейшей атаки любого рода. Этот аспект является важнейшим в сетях класса SOHO, т.е. небольших офисах. Обыкновенно вирусы не оказывают никакого влияния на малые сети и заражение ими редко приводит к потере данных. Для нас, пользователей *nix, этой проблемы практически не существует. В любом случае, т.к. небольшие сети, как правило, защищены гораздо хуже больших, кракеры стараются использовать их в качестве “опорной базы” для DoS атак, или другой своей вредоносной активности.

В последующем я опишу всю конфигурацию по кусочкам, дабы мы смогли бы проверить пошагово каждый фрагмент. Каждый шаг может потребовать от вас вставки чего-либо до, после или в середину указываемого скрипта. Все действия производятся так, чтобы (я надеюсь) ваша сеть не работала только лишь короткий промежуток времени во время настройки. Это сделано мною потому как я предположил что у многих из вас (таких как я) выделен сервер под Firewall. И так как я предпочитаю настраивать свой сервер через SSH, отключение сети может иметь пренебрежительные последствия, такие как ползания под столами или хуже того. Если же вы смелы, то вы можете скопировать скрипт в конце этого документа и запустить его на своей машине. Этот скрипт должен быть работоспособен на все сто, но тестировал я его только на своей машине, так что [ymmv](#).

Установка переменных окружения

Вы можете установить необходимые переменные окружения следующим скриптом:

```
Code: '
#!/bin/sh
#
# Внешний интерфейс
EXTIF="ppp0"

# Внутренний интерфейс
INTIF="eth1"

# Loop device/localhost
LPDIF="lo"
LPDIP="127.0.0.1"
LPDSK="255.0.0.0"
LPDNET="$LPDIP/$LPDSK"

# Необходимые утилиты
IPT="/sbin/iptables"
IFC="/sbin/ifconfig"
G="/bin/grep"
SED="/bin/sed"
AWK="/usr/bin/awk"
ECHO="/bin/echo"

# Последующие команды могут работать некорректно при локализации.

# Установка переменных окружения внешнего интерфейса
EXTIP="$IFC $EXTIF | $AWK /$EXTIF/{next}/{split($0,a,":");split(a[2],a,"
");print a[1];exit}`"
EXTBC="255.255.255.255"
#EXTMSK="$IFC $EXTIF | $G Mask:|SED 's/. *Mask:\([^\ ]*\)/\1/'"
EXTMSK="$IFC $EXTIF | $AWK /$EXTIF/{next}/{split($0,a,":");split(a[4],a,"
");print a[1];exit}`"
EXTNET="$EXTIP/$EXTMSK"
$ECHO "EXTIP=$EXTIP EXTBC=$EXTBC EXTMSK=$EXTMSK EXTNET=$EXTNET"

# Due to absence of EXTBC I manually set it to 255.255.255.255
# this (hopefully) will serve the same purpose

# Установка переменных окружения внутреннего интерфейса
INTIP="$IFC $INTIF | $AWK /$INTIF/{next}/{split($0,a,":");split(a[2],a,"
");print a[1];exit}`"
INTBC="$IFC $INTIF | $AWK /$INTIF/{next}/{split($0,a,":");split(a[3],a,"
");print a[1];exit}`"
INTMSK="$IFC $INTIF | $AWK /$INTIF/{next}/{split($0,a,":");split(a[4],a,"
");print a[1];exit}`"
INTNET="$INTIP/$INTMSK"
$ECHO "INTIP=$INTIP INTBC=$INTBC INTMSK=$INTMSK INTNET=$INTNET"
```

iptables ACCEPTS

Теперь мы должны установить АССЕРТЫ, так, чтобы мы могли соединяться с

нашим сервером. На самом деле это очень большой вопрос. Правила для надежного маршрутизатора должны по умолчанию запрещать нежели разрешать. Однако, если вы сделаете это, то потеряете все соединения. Продолжайте тестирование пока не будете уверены что ваши АССЕРТЫ работают как надо. Однако думаю, что сперва мы введем следующее и это будет предпоследним правилом в окончательном скрипте.

Code: '

```
$IPT -t nat -A PREROUTING -j ACCEPT
# $IPT -t nat -A POSTROUTING -o $EXTIF -s $INTNET -j SNAT --to $EXTIP

# Закомментируйте последующие строки (которые содержат "MASQUERADE")
# для сетей без трансляции адресов (NAT)
$IPT -t nat -A POSTROUTING -o $EXTIF -s $INTNET1 -j MASQUERADE
$IPT -t nat -A POSTROUTING -o $EXTIF -s $INTNET2 -j MASQUERADE
$IPT -t nat -A POSTROUTING -j ACCEPT
$IPT -t nat -A OUTPUT -j ACCEPT
$IPT -A INPUT -p tcp --dport auth --syn -m state --state NEW -j ACCEPT
$IPT -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
```

iptables DROP & REJECT

Теперь мы определим пару цепочек (chains) которые будут фиксировать события DROP и REJECT. Таким образом нам не придется вводить отдельные строки для каждой введенной команды. Сообщения о событиях будут отправлены сервису syslog, (обычно они фиксируются в /var/log/messages). Позже я (переводчик не имеет к этому никакого отношения) собираюсь написать скрипт на sed/grep по разбору событий для облегченного просмотра и установлю его как ежедневную работу для сервиса cron.

Эти строки следует вставить сразу после текста выше, в тот же скрипт. Когда вы это сделаете, запустите скрипт снова. Это не окажет влияния на вашу сеть, вы пока просто устанавливаете правила. Но это поможет убедиться что мы не сделали ошибок на данном этапе.

Code: '

```
# ***** Цепочки журналирования событий *****
#
# Теперь мы определяем несколько цепочек которые служат для записи
# событий о сбрасываемых пакетах. Это позволит нам избежать ввода
# команд для каждого правила. Сперва мы фиксируем DROP, а потом REJECT.
# Не жалуйтесь, если цепочки уже существуют (однако это не приведет к
# ошибкам???)
$IPT -N DROP1 2> /dev/null
$IPT -A DROP1 -m limit --limit 3/minute --limit-burst 10 -j LOG --log-prefix
'FIREWALL DROP BLOCKED:'
$IPT -A DROP1 -j DROP
$IPT -N REJECT1 2> /dev/null
$IPT -A REJECT1 -m limit --limit 3/minute --limit-burst 10 -j LOG --log-
prefix 'FIREWALL REJECT BLOCKED:'
$IPT -A REJECT1 -j REJECT
```

```

$IPT -N DROP2 2> /dev/null
$IPT -A DROP2 -m limit --limit 3/second --limit-burst 10 -j LOG --log-prefix
'FIREWALL DROP UNKNOWN:'
$IPT -A DROP2 -j DROP
$IPT -N REJECT2 2> /dev/null
$IPT -A REJECT2 -m limit --limit 3/second --limit-burst 10 -j LOG --log-
prefix 'FIREWALL REJECT UNKNOWN:'
$IPT -A REJECT2 -j REJECT

# Для тестирования фиксируем события ACCEPT
$IPT -N ACCEPT1 2> /dev/null
$IPT -A ACCEPT1 -m limit --limit 10/second --limit-burst 50 -j LOG --log-
prefix 'FIREWALL ACCEPT:'
$IPT -A ACCEPT1 -j ACCEPT

```

Сброс правил

Теперь, когда мы видим наши устройства, правильно определенные, вставляем команду сброса правил. Однако все уже назначенные правила будут сброшены. Эти строки должны быть вставлены после определения утилит, которые заканчиваются строкой: `ECHO='/bin/echo'`

Code: '

```

# Сброс всех существующих и очистка персональных цепочек.
CHAINS=`cat /proc/net/ip_tables_names 2>/dev/null`

for i in $CHAINS
do
    $IPT -t $i -F
done

for i in $CHAINS
do
    $IPT -t $i -X
done

```

Локальные интерфейсы

Теперь мы готовы для того, чтобы описать некоторые правила. Для начала мы разрешим все пакеты с loopback интерфейса, имеющие в качестве адреса назначения один из адресов наших интерфейсов.

Code: '

```

$IPT -A INPUT -i $LPDIF -s $LPDIP -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $EXTIP -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $INTIP1 -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $INTIP2 -j ACCEPT

```

Блокировка широковещательных пакетов

Теперь мы должны заблокировать все входящие и исходящие широковещательные пакеты. Это предотвратит DoS атаки против нас, и не позволит нашим клиентам проводить DoS атаки против кого либо другого. Если бы все системные администраторы следовали этим правилам, тогда много суровых и дорогих DoS атак не состоялись или были максимально ограниченными.

Code: Блокировка широковещательных пакетов

```
$IPT -A INPUT -i $EXTIF -d $EXTBC -j DROP1
$IPT -A INPUT -i $INTIF1 -d $INTBC1 -j DROP1
$IPT -A INPUT -i $INTIF2 -d $INTBC2 -j DROP1
$IPT -A OUTPUT -o $EXTIF -d $EXTBC -j DROP1
$IPT -A OUTPUT -o $INTIF1 -d $INTBC1 -j DROP1
$IPT -A OUTPUT -o $INTIF2 -d $INTBC2 -j DROP1
$IPT -A FORWARD -o $EXTIF -d $EXTBC -j DROP1
$IPT -A FORWARD -o $INTIF1 -d $INTBC1 -j DROP1
$IPT -A FORWARD -o $INTIF2 -d $INTBC2 -j DROP1
```

Теперь проверим скрипт еще раз, чтобы убедиться в том, что мы не наделали синтаксических ошибок. Также отмечу, что мы используем определенные нами DROP1 цепочки (chains). Это означает, что отбрасываемые пакеты будут отмечены в журнале событий (log file).

Блокировка доступа в локальную сеть из глобальной

Теперь мы блокируем доступ из глобальной сети в нашу локальную сеть, если мы не хотим что бы интернет провайдер назначал IP адреса для нашей внутренней сети.

Code: '

```
# Блокировать внешний доступ к локальной сети
# Это позволит остановить боевых кракеров от использования
# нашей сети как стартовой точки для других атак.
#
# Нижеприведенная строка на человеческом языке будет выглядеть как
# "если входящий пакет, пришедший на наш внешний интерфейс,
# имеет адрес назначения, отличный от адреса нашего внешнего интерфейса,
# то этот пакет не будет пропущен."
$IPT -A INPUT -i $EXTIF -d ! $EXTIP -j DROP1
```

Изолирование локальных сетей

Теперь мы предпримем некоторые действия для наших локальных сетей. Другими словами - все пакеты не относящиеся к локальным сетям должны быть заблокированы.

Code: '

```
# Теперь мы должны заблокировать все пакеты не относящиеся к
# адресному пространству наших локальных сетей.
# Запомните, если вы подключите свой ноутбук к другому разъему,
```

```
# вам надо убедиться, что ваш сетевой адрес соответствует адресам этой сети.
#
# Первая локальная сеть
$IPT -A INPUT -i $INTIF1 -s ! $INTNET1 -j DROP1
$IPT -A OUTPUT -o $INTIF1 -d ! $INTNET1 -j DROP1
$IPT -A FORWARD -i $INTIF1 -s ! $INTNET1 -j DROP1
$IPT -A FORWARD -o $INTIF1 -d ! $INTNET1 -j DROP1
# Вторая локальная сеть
$IPT -A INPUT -i $INTIF2 -s ! $INTNET2 -j DROP1
$IPT -A OUTPUT -o $INTIF2 -d ! $INTNET2 -j DROP1
$IPT -A FORWARD -i $INTIF2 -s ! $INTNET2 -j DROP1
$IPT -A FORWARD -o $INTIF2 -d ! $INTNET2 -j DROP1
```

Дальше мы сделаем некоторые дополнительные проверки исходящих пакетов и остановим все icmp пакеты кроме ping.

Code: '

```
# Дополнительная проверка
$IPT -A OUTPUT -o $EXTIF -s ! $EXTNET -j DROP1
# Блокируем исходящие ICMP (кроме PING)
$IPT -A OUTPUT -o $EXTIF -p icmp --icmp-type ! 8 -j DROP1
$IPT -A FORWARD -o $EXTIF -p icmp --icmp-type ! 8 -j DROP1
```

Замечательно. Двигаемся дальше и проверяем скрипт на ошибки.

Ports

Предполагая что у нас все сработало мы заткнем еще несколько портов, доступ по которым может представлять для нас серьезную опасность:

```
# COMmon ports:
# 0 is tcpmux; SGI had vulnerability, 1 is common attack
# 13 is daytime
# 98 is Linuxconf
# 111 is sunrpc (portmap)
# 137:139, 445 is Microsoft
# SNMP: 161,2
# Squid flotilla: 3128, 8000, 8008, 8080
# 1214 is Morpheus or KaZaA
# 2049 is NFS
# 3049 is very virulent Linux Trojan, mistakable for NFS
# Common attacks: 1999, 4329, 6346
# Common Trojans 12345 65535
COMBLOCK="0:1 13 98 111 137:139 161:162 445 1214 1999 2049 3049 4329 6346
3128 8000 8008 8080 12345 65535"
# TCP ports:
# 98 is Linuxconf
# 512-515 is rexec, rlogin, rsh, printer(lpd)
# [very serious vulnerabilities; attacks continue daily]
# 1080 is Socks proxy server
# 6000 is X (NOTE X over SSH is secure and runs on TCP 22)
# Block 6112 (Sun's/HP's CDE)
TCPBLOCK="$COMBLOCK 98 512:515 1080 6000:6009 6112"

# UDP ports:
```



```
# 161:162 is SNMP
# 520=RIP, 9000 is Sangoma
# 517:518 are talk and ntalk (more annoying than anything)
UDPBLOCK="$COMBLOCK 161:162 520 123 517:518 1427 9000 9 6346 3128 8000 8008
8080 12345 65535"
```

После определения переменных окружения нам останется только пробежаться по ним циклом:

```
echo -n "FW: Blocking attacks to TCP port"
for i in $TCPBLOCK;
do
    echo -n "$i "
    $IPT -A INPUT    -p tcp --dport $i    -j DROP
    $IPT -A OUTPUT    -p tcp --dport $i    -j DROP
    $IPT -A FORWARD  -p tcp --dport $i    -j DROP
done
echo ""

echo -n "FW: Blocking attacks to UDP port "
for i in $UDPBLOCK;
do
    echo -n "$i "
    $IPT -A INPUT    -p udp --dport $i    -j DROP
    $IPT -A OUTPUT    -p udp --dport $i    -j DROP
    $IPT -A FORWARD  -p udp --dport $i    -j DROP
done
echo ""
```

Ну что ж, теперь каждый раз, когда мы запускаем скрипт, эти строки просто добавляются к уже существующим... что создаёт небольшой бардак. По этой причине мы собираемся перепрыгнуть в начало скрипта... сразу после переменных окружения для `sed` и `grep`, но перед переменными для `EXTIP` и `EXTBC` - там мы добавляем цикл, который производит очистку. Так мы будем уверены, что работаем в чистой среде. Нам не приходилось задумываться об этом ранее, потому что мы не имели возможности оттестить скрипт без обрыва соединения либо закрытия файрвола. Этот скрипт сначала выставляет всю политику в `DROP`, после чего очищает и удаляет наши цепочки. Чтобы убедиться, что мы по-прежнему можем залогиниться по `ssh` на наш сервер после того, как скрипт перезапустится, мы добавим цепочку `INPUT` для `ssh`. Пока что поставим её в конец скрипта. Это делается для того чтобы избежать открытия дыры в момент установки новых правил, что является довольно распространённой ошибкой:

```
# Отказываем, потом принимаем: это уберёт дыру в момент,
# когда мы закрываем порты
$IPT -P INPUT      DROP
$IPT -P OUTPUT     DROP
$IPT -P FORWARD   DROP

# Очищаем все существующие цепочки и стираем дополнительные
CHAINS=`cat /proc/net/ip_tables_names 2>/dev/null`
for i in $CHAINS;
do
```

```

    $IPT -t $i -F
done
for i in $CHAINS;
do
    $IPT -t $i -X
done
$IPT -A INPUT -i $INTIF1 -p tcp --dport 22 --syn -m state --state NEW -j
ACCEPT

```

Sysctl'ы

Сразу после этого мы запустим sysctl'ы для tcp_syncookies, icmp_echo_ignore_broadcasts, rp_filter и accept_source_route. До этого момента многие правила, которые мы "проверяли", фактически не выполнялись. По сути, мы просто делали проверку синтаксических ошибок. Теперь наши правила будут выполняться в полной мере:

```

echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# Проверка адреса источника
for f in /proc/sys/net/ipv4/conf/*/rp_filter;
do
    echo 1 > $f
done
# Отключаем роутинг источника IP и ICMP-редиректы
for f in /proc/sys/net/ipv4/conf/*/accept_source_route;
do
    echo 0 > $f
done
for f in /proc/sys/net/ipv4/conf/*/accept_redirects;
do
    echo 0 > $f
done
echo 1 > /proc/sys/net/ipv4/ip_forward

```

Теперь мы собираемся добавить трекинг ftp-соединения, так что нам не доведётся наблюдать ошибки PASV при установке паков:

```

# Запускаем трекинг ftp-соединения
MODULES="ip_nat_ftp ip_conntrack_ftp"
for i in $MODULES;
do
    echo "Добавляем модуль $i"
    modprobe $i
done

```

Базовая конфигурация NAT

А сейчас возвратимся к концу нашего скрипта, поскольку мы собираемся открыть сервисы для хостов, находящихся за файерволом. Я включил следующие сервисы: IRC, MSN, ICQ, and NFS, FTP, domain,time и некоторые другие. Самое главное здесь то, что эти сервисы могут использоваться ТОЛЬКО ЗА

файерволом. Таким образом никто не сможет их использовать по ftp внутри Вашей локальной сетки:

```
IRC='ircd'
MSN=1863
ICQ=5190
NFS='sunrpc'
# Мы пользуемся sync!!
PORTAGE='rsync'
OpenPGP_HTTP_Keyserver=11371
# Все порты сервисов считываются из /etc/services
TCPSESV="domain ssh http https ftp ftp-data mail pop3 pop3s imap3 imaps imap2
time $PORTAGE $IRC $MSN $ICQ $OpenPGP_HTTP_Keyserver"
UDPSERV="domain time"
echo -n "FW: Allowing inside systems to use service:"
for i in $TCPSESV;
do
    echo -n "$i "
    $IPT -A OUTPUT -o $EXTIF -p tcp -s $EXTIP --dport $i --syn -m state --state
NEW -j ACCEPT
    $IPT -A FORWARD -i $INTIF1 -p tcp -s $INTNET1 --dport $i --syn -m state
--state NEW -j ACCEPT
    $IPT -A FORWARD -i $INTIF2 -p tcp -s $INTNET2 --dport $i --syn -m state
--state NEW -j ACCEPT
done
echo ""

echo -n "FW: Allowing inside systems to use service:"
for i in $UDPSERV;
do
    echo -n "$i "
    $IPT -A OUTPUT -o $EXTIF -p udp -s $EXTIP --dport $i -m state --state NEW -
j ACCEPT
    $IPT -A FORWARD -i $INTIF1 -p udp -s $INTNET1 --dport $i -m state --state
NEW -j ACCEPT
    $IPT -A FORWARD -i $INTIF2 -p udp -s $INTNET2 --dport $i -m state --state
NEW -j ACCEPT
done
echo ""
```

Теперь сделаем так, чтобы файервол позволил нам пинговать внешний мир. Для этого разрешим прохождение icmp-пакетов через внешний интерфейс:

```
# Разрешаем внешнее пингование
$IPT -A OUTPUT -o $EXTIF -p icmp -s $EXTIP --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A FORWARD -i $INTIF1 -p icmp -s $INTNET1 --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A FORWARD -i $INTIF2 -p icmp -s $INTNET2 --icmp-type 8 -m state --state
NEW -j ACCEPT

# Разрешим файерволу пинговать наши внутренние сетки:
$IPT -A OUTPUT -o $INTIF1 -p icmp -s $INTNET1 --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A OUTPUT -o $INTIF2 -p icmp -s $INTNET2 --icmp-type 8 -m state --state
NEW -j ACCEPT
```

Теперь мы по умолчанию будем записывать в журнал все прочие запросы, но никак не будем на них реагировать. Все что нам надо было принять мы описали в начале наших правил. Так что заканчиваем правила блокировкой всего, что специально не разрешено:

```
# Заблокируем все, что осталось:
$IPT -A INPUT -j DROP1
$IPT -A OUTPUT -j REJECT1
$IPT -A FORWARD -j DROP1
```

Итак, все сделано. я имею дружественные nmap и nessus для моих соединений с вышеописанным набором правил и ничего не препятствует использованию IRC, MSN, ICQ, и emerge sync.

The full script

А сейчас, полноценный скрипт во всей своей красе (Заодно я поместил форвардинг ssh в более подходящее для него место):

```
# Внешний интерфейс
EXTIF=ppp0
# Внутренний интерфейс
INTIF1=eth1
INTIF2=eth2
# Loop-устройство/localhost
LPDIF=lo
LPDIP=127.0.0.1
LPDMASK=255.0.0.0
LPDNET="$LPDIP/$LPDMASK"
# Переменные текстовых инструментов
IPT='/sbin/iptables'
IFC='/sbin/ifconfig'
G='/bin/grep'
SED='/bin/sed'
# Последнее (но немаловажное) - пользователи
JAMES=192.168.1.77
TERESA=192.168.2.77
# Deny вместо accept: предотвращает открытие "дыр"
# в то время, как мы закрываем порты и все такое
$IPT -P INPUT DROP
$IPT -P OUTPUT DROP
$IPT -P FORWARD DROP
# Сброс всех существующих цепочек и стирание персональных цепочек
CHAINS=`cat /proc/net/ip_tables_names 2>/dev/null`
for i in $CHAINS
do
    $IPT -t $i -F
    $IPT -t $i -X
done
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Проверка адреса источника
for f in /proc/sys/net/ipv4/conf/*/rp_filter;
do
    echo 1 > $f
```

```

done
# Запрет маршрутизации IP от источника и редиректов ICMP
for f in /proc/sys/net/ipv4/conf/*/accept_source_route;
do
    echo 0 > $f
done
for f in /proc/sys/net/ipv4/conf/*/accept_redirects;
do
    echo 0 > $f
done
echo 1 > /proc/sys/net/ipv4/ip_forward
# Установка переменных среды для внешнего интерфейса
EXTIP="\$IFC \$EXTIF|$G addr:|$SED 's/. *addr:\([^\ ]*\) .*/\1/'"
EXTBC="\$IFC \$EXTIF|$G Bcast:|$SED 's/. *Bcast:\([^\ ]*\) .*/\1/'"
EXTBC="255.255.255.255"
EXTMSK="\$IFC \$EXTIF|$G Mask:|$SED 's/. *Mask:\([^\ ]*\) /\1/'"
EXTNET="\$EXTIP/\$EXTMSK"
#echo "EXTIP=\$EXTIP EXTBC=\$EXTBC EXTMSK=\$EXTMSK EXTNET=\$EXTNET"
echo "EXTIP=\$EXTIP EXTBC=\$EXTBC EXTMSK=\$EXTMSK EXTNET=\$EXTNET"
# Так как EXTBC отсутствует, я устанавливаю ее вручную как it to
255.255.255.255
# Это (надеюсь) послужит тем же целям
# Установка переменных среды для первого внутреннего интерфейса
INTIP1="\$IFC \$INTIF1|$G addr:|$SED 's/. *addr:\([^\ ]*\) .*/\1/'"
INTBC1="\$IFC \$INTIF1|$G Bcast:|$SED 's/. *Bcast:\([^\ ]*\) .*/\1/'"
INTMSK1="\$IFC \$INTIF1|$G Mask:|$SED 's/. *Mask:\([^\ ]*\) /\1/'"
INTNET1="\$INTIP1/\$INTMSK1"
echo "INTIP1=\$INTIP1 INTBC1=\$INTBC1 INTMSK1=\$INTMSK1 INTNET1=\$INTNET1"
# Установка переменных среды для второго внутреннего интерфейса
INTIP2="\$IFC \$INTIF2|$G addr:|$SED 's/. *addr:\([^\ ]*\) .*/\1/'"
INTBC2="\$IFC \$INTIF2|$G Bcast:|$SED 's/. *Bcast:\([^\ ]*\) .*/\1/'"
INTMSK2="\$IFC \$INTIF2|$G Mask:|$SED 's/. *Mask:\([^\ ]*\) /\1/'"
INTNET2="\$INTIP2/\$INTMSK2"
echo "INTIP2=\$INTIP2 INTBC2=\$INTBC2 INTMSK2=\$INTMSK2 INTNET2=\$INTNET2"
# Сейчас мы собираемся создать несколько собственных цепочек, результатом
работы
# которых будет логгинг отброшенных пакетов. Это поможет нам избежать
необходимости
# вводить команду log перед каждым отбрасыванием пакета, что мы хотим
запротоколировать.
# Первыми идут лог отброшенных пакетов и собственно отброс, затем лог пакетов
с отказами
# и собственно отказы.
# Отключаем сообщения о том, что цепочки уже существуют (чтобы перезапуск был
без мусора)
$IPT -N DROP1 2> /dev/null
$IPT -A DROP1 -j LOG --log-prefix 'DROP1:'
$IPT -A DROP1 -j DROP
$IPT -N REJECT1 2> /dev/null
$IPT -A REJECT1 -j LOG --log-prefix 'REJECT1:'
$IPT -A REJECT1 -j REJECT
# Весь трафик от устройства loopback принимается
# если IP совпадает с любым из наших интерфейсов.
$IPT -A INPUT -i $LPDIF -s $LPDIP -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $EXTIP -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $INTIP1 -j ACCEPT
$IPT -A INPUT -i $LPDIF -s $INTIP2 -j ACCEPT
# Широковещательные пакеты блокируем
$IPT -A INPUT -i $EXTIF -d $EXTBC -j DROP1

```

```

$IPT -A INPUT -i $INTIF1 -d $INTBC1 -j DROP1
$IPT -A INPUT -i $INTIF2 -d $INTBC2 -j DROP1
$IPT -A OUTPUT -o $EXTIF -d $EXTBC -j DROP1
$IPT -A OUTPUT -o $INTIF1 -d $INTBC1 -j DROP1
$IPT -A OUTPUT -o $INTIF2 -d $INTBC2 -j DROP1
$IPT -A FORWARD -o $EXTIF -d $EXTBC -j DROP1
$IPT -A FORWARD -o $INTIF1 -d $INTBC1 -j DROP1
$IPT -A FORWARD -o $INTIF2 -d $INTBC2 -j DROP1
# Блокируем доступ к внутренней сети из WAN
# Это также призвано не дать нечестивым крякерам использовать нашу сетку
# в качестве отправной точки для атак на других людей
# Перевод с языка iptables:
# "если пришедшие на наружный интерфейс пакеты были отправлены не с выданного
# nefarious адреса, выкинуть их как горячую картошку"
$IPT -A INPUT -i $EXTIF -d ! $EXTIP -j DROP1
# А сейчас мы блокируем внутренние адреса, кроме двух, присвоенных нашим двум
# внутренним интерфейсам.....только помните, что если вы воткнете свой лэптоп
или
# какой другой pc в напрямую в одну из этих сетевых карт, то нужно
удостовериться,
# что они имеют именно эти IP-адреса или добавить соответствующий адрес
отдельно.
# Первый интерфейс/первая внутренняя сеть
$IPT -A INPUT -i $INTIF1 -s ! $INTNET1 -j DROP1
$IPT -A OUTPUT -o $INTIF1 -d ! $INTNET1 -j DROP1
$IPT -A FORWARD -i $INTIF1 -s ! $INTNET1 -j DROP1
$IPT -A FORWARD -o $INTIF1 -d ! $INTNET1 -j DROP1
# Второй интерфейс/вторая внутренняя сеть
$IPT -A INPUT -i $INTIF2 -s ! $INTNET2 -j DROP1
$IPT -A OUTPUT -o $INTIF2 -d ! $INTNET2 -j DROP1
$IPT -A FORWARD -i $INTIF2 -s ! $INTNET2 -j DROP1
$IPT -A FORWARD -o $INTIF2 -d ! $INTNET2 -j DROP1
# Дополнительная Egress-проверка
$IPT -A OUTPUT -o $EXTIF -s ! $EXTNET -j DROP1
# Блокируем исходящие пакеты ICMP (за исключением PING)
$IPT -A OUTPUT -o $EXTIF -p icmp --icmp-type ! 8 -j DROP1
$IPT -A FORWARD -o $EXTIF -p icmp --icmp-type ! 8 -j DROP1
# печально известные порты:
# 0 - tcprtx; у SGI есть уязвимость, через которую можно атаковать
# 13 - daytime
# 98 - Linuxconf
# 111 - sunrpc (portmap)
# 137:139, 445 - Microsoft
# SNMP: 161,2
# Флотилия Squid: 3128, 8000, 8008, 8080
# 1214 - Morpheus или KaZaA
# 2049 - NFS
# 3049 - очень заразный троян для Linux, часто путаемый с NFS
# Часто атакуемые: 1999, 4329, 6346
# Частые трояны 12345 65535
COMBLOCK="0:1 13 98 111 137:139 161:162 445 1214 1999 2049 3049 4329 6346
3128 8000 8008 8080 12345 65535"
# Порты TCP:
# 98 - Linuxconf
# 512-515 - rexec, rlogin, rsh, printer(lpd)
# [очень серьезные уязвимости; продолжаются ежедневные атаки]
# 1080 - прокси-серверы Socks
# 6000 - X (ЗАМЕЧАНИЕ. X через SSH - безопасен, и работает на порту TCP 22)
# Блокировка 6112 (CDE у Sun и HP)

```

```

TCPBLOCK="$COMBLOCK 98 512:515 1080 6000:6009 6112"
# Порты UDP:
# 161:162 - SNMP
# 520=RIP, 9000 - Sangoma
# 517:518 - talk и ntalk (самые надоедливые)
UDPBLOCK="$COMBLOCK 161:162 520 123 517:518 1427 9000"
echo -n "FW: Blocking attacks to TCP port"
for i in $TCPBLOCK;
do
    echo -n "$i "
    $IPT -A INPUT -p tcp --dport $i -j DROP1
    $IPT -A OUTPUT -p tcp --dport $i -j DROP1
    $IPT -A FORWARD -p tcp --dport $i -j DROP1
done
echo ""
echo -n "FW: Blocking attacks to UDP port "
for i in $UDPBLOCK;
do
    echo -n "$i "
    $IPT -A INPUT -p udp --dport $i -j DROP1
    $IPT -A OUTPUT -p udp --dport $i -j DROP1
    $IPT -A FORWARD -p udp --dport $i -j DROP1
done
echo ""
# Открываем отслеживание соединений по ftp
MODULES="ip_nat_ftp ip_conntrack_ftp"
for i in $MODULES;
do
    echo "Inserting module $i"
    modprobe $i
done
# Защищаем некоторые распространенные клиенты для чата.
# Уберите из списка допустимых для пущей безопасности.
IRC='ircd'
MSN=1863
ICQ=5190
NFS='sunrpc'
# Нам нужно синхронизировать данные!!
PORTAGE='rsync'
OpenPGP_HTTP_Keyserver=11371
# Все порты сервисов читаются из /etc/services
TCPSERV="domain ssh http https ftp ftp-data mail pop3 pop3s imap3 imaps
imap2 time $PORTAGE $IRC $MSN $ICQ $OpenPGP_HTTP_Keyserver" UDPSERV="domain
time"
echo -n "FW: Allowing inside systems to use service:"
for i in $TCPSERV;
do
    echo -n "$i "
    $IPT -A OUTPUT -o $EXTIF -p tcp -s $EXTIP --dport $i --syn -m state
--state NEW -j ACCEPT
    $IPT -A FORWARD -i $INTIF1 -p tcp -s $INTNET1 --dport $i --syn -m state
--state NEW -j ACCEPT
    $IPT -A FORWARD -i $INTIF2 -p tcp -s $INTNET2 --dport $i --syn -m state
--state NEW -j ACCEPT
done
echo ""
echo -n "FW: Allowing inside systems to use service:"
for i in $UDPSERV;
do

```

```

echo -n "$i "
$IPT -A OUTPUT -o $EXTIF -p udp -s $EXTIP --dport $i -m state --state NEW
-j ACCEPT
$IPT -A FORWARD -i $INTIF1 -p udp -s $INTNET1 --dport $i -m state --state
NEW -j ACCEPT
$IPT -A FORWARD -i $INTIF2 -p udp -s $INTNET2 --dport $i -m state --state
NEW -j ACCEPT
done
echo ""
# Разрешается ping наружу
$IPT -A OUTPUT -o $EXTIF -p icmp -s $EXTIP --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A FORWARD -i $INTIF1 -p icmp -s $INTNET1 --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A FORWARD -i $INTIF2 -p icmp -s $INTNET2 --icmp-type 8 -m state --state
NEW -j ACCEPT
# Файерволу разрешается ping внутренних систем
$IPT -A OUTPUT -o $INTIF1 -p icmp -s $INTNET1 --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A OUTPUT -o $INTIF2 -p icmp -s $INTNET2 --icmp-type 8 -m state --state
NEW -j ACCEPT
$IPT -A INPUT -i $INTIF1 -p tcp --dport 22 --syn -m state --state NEW -j
ACCEPT
$IPT -t nat -A PREROUTING -j ACCEPT
$IPT -t nat -A POSTROUTING -o $EXTIF -s $INTNET1 -j MASQUERADE
$IPT -t nat -A POSTROUTING -o $EXTIF -s $INTNET2 -j MASQUERADE
$IPT -t nat -A POSTROUTING -j ACCEPT
$IPT -t nat -A OUTPUT -j ACCEPT
$IPT -A INPUT -p tcp --dport auth --syn -m state --state NEW -j ACCEPT
$IPT -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPT -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
# Заблокировать и запротоколировать все, что мы могли забыть.
$IPT -A INPUT -j DROP1
$IPT -A OUTPUT -j REJECT1
$IPT -A FORWARD -j DROP1

```

Установка почтовой системы

Установка почтовой системы на Gentoo linux для небольшой конторы

Используется postfix + cyrus-imap, авторизация пользователей через cyrus-sasl без прикручивания какой-либо БД (подходит для контор с несколькими десятками почтовых ящиков - хранить что-либо в mysql или postgres в данном случае нет смысла). Пользователи хранятся в базе sasldb, поэтому нет нужды заводить в системе реальных пользователей.

Возможно при установке указанных пакетов понадобится установить все зависимости, которые они за собой тянут. Предоставим системе самой

разобраться, что же ей не хватает - она с этим вполне успешно справляется.
Предпочитаю использовать для /var reiserfs, но это сугубо личные предпочтения.

emerge -pv cyrus-sasl

```
[ebuild N ] dev-libs/cyrus-sasl-2.1.20 -authdaemond +berkdb -debug +gdbm +java
-kerberos -ldap +mysql +pam -postgres +ssl -static 0 kB
```

На всякий случай проверяем флаги - все ОК

emerge cyrus-sasl

Этот пакет поставили, едем дальше

emerge -pv postfix

These are the packages that I would merge, in order:

Calculating dependencies ...done!

```
[ebuild N ] mail-mta/postfix-2.1.5-r1 +ipv6 -ldap -mailwrapper -mbox +mysql +pam -postgres -sasl*(-
selinux) +ssl -vda 0 kB
```

Здесь нам может пригодиться флаг sasl, хотя в моем случае пока нет нужды авторизоваться на smtp, возможно в дальнейшем здесь будет описана авторизация через cyrus-sasl на smtp - вообще выставляйте флаги по вкусу кому что надо

USE="sasl" emerge postfix

Постфикс собрался

emerge -pv cyrus-imapd

These are the packages that I would merge, in order:

Calculating dependencies ...done!

```
[ebuild N ] net-mail/cyrus-imapd-2.2.10 -afs -drac -idled -kerberos +pam -snmp +ssl +tcpd 0 kB
```

После проверки и установки нужных нам флагов пересобираем openssl и imap-сервер

emerge cyrus-imapd

Программа для администрирования имаповских ящиков

emerge cyrus-imap-admin

После того как все нужные нам пакеты поставлены, можно приступить к настройке

сначала cyrus-sasl

passwd cyrus

pwconv

chown -R cyrus:mail /etc/sasl2 - доступ cyrus к базе /etc/sasl2/sasl2db2

saslpasswd2 cyrus - заводим в saslpasswd2
sasldblistusers2 - для проверки

Разбираемся с postfix

Содержимое /etc/postfix/main.cf, ниже приведена примерная конфигурация почтового сервера подключенного напрямую к инету (не через relayhost), естественно для этого случая должна быть MX-запись в ДНС. Подчеркиваю, что умышленно опускаю многие параметры в main.cf, чтобы не раздувать описание. Добавьте все остальное руководствуясь документацией postfix.

```
queue_directory = /var/spool/postfix
command_directory = /usr/sbin
daemon_directory = /usr/lib/postfix
mail_owner = postfix
myhostname = mail.domain.tld
mydomain = mail.domain.tld
myorigin = $mydomain
inet_interfaces = all
mydestination = $myhostname, localhost.$mydomain, localhost, $mydomain
local_recipient_maps =
unknown_local_recipient_reject_code = 550
mynetworks_style = subnet
mynetworks = 192.168.1.0/24, 127.0.0.0/8
relay_domains = $mydestination
НА ЭТУ СТРОЧКУ ОБРАТИТЕ ВНИМАНИЕ!!!
mailbox_transport = lmtp:unix:/var/imap/socket/lmtp
Если надо подключить procmail то вместо нее пишем
mailbox_transport = procmail
```

Когда добавили все, что нужно в main.cf, идем в /etc/postfix/master.cf

Ищем такую строку

```
# Also specify in main.cf: cyrus_destination_recipient_limit=1
cyrus    unix    -    n    n    -    -    pipe
user=cyrus argv=/cyrus/bin/deliver -e -r ${sender} -m ${extension} ${user}
```

И заменяем путь на

```
# Also specify in main.cf: cyrus_destination_recipient_limit=1
cyrus    unix    -    n    n    -    -    pipe
user=cyrus argv=/cyrus/bin/deliver -e -r ${sender} -m ${extension} ${user}
```

При использовании procmail, добавляем

```
procmail unix    -    n    n    -    -    pipe
flags=R user=cyrus argv=/usr/bin/procmail -p /etc/procmailrc USER=${user}
```

Где файл /etc/procmailrc :

```
DELIVERMAIL=/usr/lib/cyrus/deliver
LOGFILE=/var/log/procmaillog
IMAP="$DELIVERMAIL -e -a $USER -m user.$USER"
(если надо подключить spamassassin)
:0fw : spamassassin.lock
* < 90000
| /usr/bin/spamassassin
и далее обязательно
:0
| $IMAP
:0w
{
EXITCODE=$?
HOST
}
```

Все, с постфиксом разобрались, теперь cyrus-imapd

Редактируем /etc/cyrus.conf

```
# $Header: /var/cvsroot/gentoo-x86/net-mail/cyrus-imapd/files/cyrus.conf,v 1.4 2004/07/18 04:02:23
dragonheart Exp $
# Standard standalone server configuration.
START {
# Do not delete this entry!
recover      cmd="ctl_cyrusdb -r"
# This is only necessary if using idled for IMAP IDLE.
#idled       cmd="idled"
}
# UNIX sockets start with a slash and are put into /var/imap/socket.
SERVICES {
# Add or remove based on preferences.
imap         cmd="imapd" listen="imap2" prefork=0
pop3         cmd="pop3d" listen="pop-3" prefork=0
# Don't forget to generate the needed keys for SSL or TLS
# (see doc/html/install-configure.html).
imaps        cmd="imapd -s" listen="imaps" prefork=0
pop3s        cmd="pop3d -s" listen="pop3s" prefork=0
sieve        cmd="timsieved" listen="sieve" prefork=0
# at least one LMTP is required for delivery
#lmtp        cmd="lmtpd" listen="lmtp" prefork=0
##ОБРАТИТЕ ВНИМАНИЕ НА ЭТУ СТРОКУ
lmtpunix     cmd="lmtpd" listen="/var/imap/socket/lmtp" prefork=0
# this is only necessary if using notifications
#notifycmd="notifyd" listen="/var/imap/socket/notify" proto="udp" prefork=1
}
```

```
EVENTS {  
    # This is required.  
    checkpoint    cmd="ctl_cyrusdb -c" period=30  
    # This is only necessary if using duplicate delivery suppression.  
    delprune      cmd="ctl_deliver -E 3" period=1440  
    # This is only necessary if caching TLS sessions.  
    tlsprune      cmd="tls_prune" period=1440  
}
```

Вот так делаем сертификаты

```
openssl req -new -nodes -out req.pem -keyout key.pem  
openssl rsa -in key.pem -out new.key.pem  
openssl x509 -in req.pem -out ca-cert -req \  
-signkey new.key.pem -days 999  
cp new.key.pem /etc/ssl/cyrus/server.pem  
rm new.key.pem  
cat ca-cert >> /etc/ssl/cyrus/server.pem  
chown cyrus:mail /etc/ssl/cyrus/server.pem  
chmod 600 /etc/ssl/cyrus/server.pem # Your key should be protected
```

Проверяем все-ли директории созданы

```
/var/imap  
cd /var  
mkdir imap  
chown cyrus:mail imap  
chmod 750 imap  
/var/spool/imap  
cd /var/spool  
mkdir imap  
chown cyrus:mail imap  
chmod 750 imap  
/usr/sieve  
cd /usr  
mkdir sieve  
chown cyrus:mail sieve  
chmod 750 sieve
```

Далее редактируем /etc/imapd.conf

```
configdirectory: /var/imap  
partition-default: /var/spool/imap  
auto_transition: yes  
tls_ca_path: /etc/ssl/cyrus  
tls_cert_file: /etc/ssl/cyrus/server.pem  
tls_key_file: /etc/ssl/cyrus/server.pem  
admins: cyrus  
hashimapspool: yes  
allowanonymouslogin: no
```

```
allowplaintext:      yes
sasl_pwcheck_method: auxprop
sasl_auxprop_plugin: sasl_db
sasldb_path:         /etc/sasl2/sasldb2
sasl_mech_list:      LOGIN PLAIN
```

Авторизация через базу sasldb, механизмы авторизации LOGIN,PLAIN Если мы будем использовать обычные открытые пароли, без шифрования (например внутри организации) то изменим в /etc/imapd.conf

```
allowanonymouslogin: yes
```

вроде бы с настройкой cyrus покончено

Теперь займемся почтовыми ящиками юзеров, для этого используем программу **cyradm**

```
/etc/init.d/cyrus start
```

```
cyradm -user cyrus -server localhost
```

```
localhost> cm user.testuser
```

```
localhost> help - выводит все команды для руления юзерскими ящиками
```

```
localhost> exit
```

Не забудьте забить в sasldb2 нового юзера:

```
saslpasswd2 testuser
```

Поднимаем сервисы

```
/etc/init.d/postfix start
```

```
/etc/init.d/cyrus start
```

Настраиваем почтового клиента (проверено с thunderbird 1.0 и с KMail 1.7.1, с Outlook Express тоже не должно быть проблем).

Пробуем отправить письмо, если что-то не работает - смотрим логи, прежде всего проверьте права на доступ в /var/imap/socket/lmtp для пользователей postfix и cyrus, чаще всего проблемы связаны именно с этим. Также проверьте права на доступ к /etc/sasl2/sasldb2 для пользователя cyrus, который должен быть в группе mail.

Желательно поставить антивирус например clamav используем для этого связку clamav + clamsmtp хотя возможны и другие варианты например amavisd-new. Установка прекрасно описана здесь http://www.nixp.ru/articles/clamav_postfix, необходимо руководствоваться ей, естественно с адаптацией под реалии Gentoo

```
emerge clamav
```

```
ACCEPT_KEYWORDS=""~x86" emerge clamsmtp (пока замаскирован в портаджах)
```

Затем в файле `/etc/conf.d/clamd` `START_CLAMD = yes` (для версии старше 0.85 уже не актуально - все настройки только в `clamd.conf` и `clamsmtpd.conf`)

Редактируем файлы `/etc/clamd.conf` и `/etc/clamsmtpd.conf` в соответствии с рекомендациями приведенными выше по ссылке Обратите внимание на параметры `LocalSocket: /var/run/clamav/clamd.sock` в `/etc/clamav.conf` и `ClamAddress: /var/run/clamav/clamd.sock` в `/etc/clamsmtpd.conf` - путь и имя файла должен одинаковым для обоих конфигов

В файл `main.cf` необходимо добавить две строчки:

```
content_filter = scan:127.0.0.1:10025
```

```
receive_override_options = no_address_mappings
```

Первая говорит postfix'у о том, что необходимо пересылать всю почту через сервис (фильтр) 'scan' на 10025-ый порт, который, как раз, открыт `clamsmtpd`. Вторая строчка говорит о том, чтобы postfix не делал никаких манипуляций с адресами до того, как они дойдут до `content_filter`. Так что получается, что фильтр работает с реальными почтовыми адресами, а не с результатами перевода в виртуальные псевдонимы, маскарadingом и т.п.

В файл `master.cf` необходимо добавить следующие строки:

```
# AV scan filter (used by content_filter)
```

```
scan    unix    -    -    n    -    16    smtp
```

```
    -o smtp_send_xforward_command=yes
```

```
# For injecting mail back into postfix from the filter
```

```
127.0.0.1:10026 inet n    -    n    -    16    smtpd
```

```
    -o content_filter=
```

```
-o
```

```
receive_override_options=no_unknown_recipient_checks,no_header_body_checks
```

```
    -o smtpd_helo_restrictions=
```

```
    -o smtpd_client_restrictions=
```

```
    -o smtpd_sender_restrictions=
```

```
    -o smtpd_recipient_restrictions=permit_mynetworks,reject
```

```
    -o mynetworks_style=host
```

```
    -o smtpd_authorized_xforward_hosts=127.0.0.0/8
```

Примечание: вокруг знака '=' пробелы не ставить. Значение `127.0.0.1:10026` открывает 10026-порт для возвращения почты обратно от `clamsmtpd`.

rc-update add clamd default

rc-update add clamsmtpd default

/etc/init.d/postfix reload

все - антивирус должен проверять всю почту проходящую через Ваш почтовик

P.S. Все вышеописанное проверялось на моей рабочей машине и на серваке небольшой конторы. Автор (или авторы) не несут ответственности за безграмотное и необдуманное применение данного руководства. Ваша безопасность в ваших руках и в голове! Данное описание не претендует на полное и развернутое и не является единственно возможным вариантом конфигурации. Также выражаю признательность разработчикам вышеописанных замечательных программ за прекрасное исполнение и хорошую документацию. Практически все вышенаписанное подчерпнуто из документации к самим пакетам, а также некоторые мысли родились из отрывочных данных, блуждающих в рунете - большое спасибо за это их авторам.

PS. Некоторая путаница возникает с переадресацией почты - самый простой вариант использование файла `aliases` - но до его использования в отличие от `sendmail`, ящик который должен быть переадресован должен быть все-таки создан в системе через `cyradm -user cyrus -auth login -server localhost` команда `cm` и не забыть что `etc/postfix/main.cf`

`alias_maps = hash:/usr/local/etc/postfix/aliases`

`alias_database = hash:/usr/local/etc/postfix/aliases`

Настройка vsftpd

Введение

Это перевод статьи `en:HOWTO vsftpd`. Кое-что изменено и добавлено от себя для более понятного, как мне кажется, изложения

Взято из `en:HOWTO Plan, setup and run a high school Gentoo Club` и модифицировано.

Это руководство описывает процесс установки VSFTP и настройки его для анонимного доступа с правами только для чтения. FTP (File Transfer Protocol) - старый, но надежный протокол, используемый для быстрой передачи файлов в сети.

Установка VSFTPD

Зайдите в систему под пользователем `root` и введите команду: `Code: Установка emerge vsftpd`

Настройка

Это очень легко. Откройте файл `/etc/vsftpd/vsftpd.conf` вашим любимым текстовым редактором и внесите в него вот эти изменения:

Общие настройки

Файл: /etc/vsftpd/vsftpd.conf

```
dirmessage_enable=YES
# banner_file=/etc/vsftpd/vsftpd.banner # edit banner first
chown_uploads=NO
xferlog_enable=YES
idle_session_timeout=600
data_connection_timeout=120
ascii_upload_enable=NO
ascii_download_enable=NO
chroot_list_enable=YES
background=YES
listen=YES
ls_recurse_enable=NO
```

Анонимный пользователь (Anonymous), только для чтения

Файл: /etc/vsftpd/vsftpd.conf

```
anonymous_enable=YES
anon_upload_enable=NO
anon_mkdir_write_enable=NO
```

Отключаем локальных пользователей

Это плохая идея разрешать локальным пользователям доступ через ftp (проще sftp/ssh), потому мы отключаем им доступ. Файл: /etc/vsftpd/vsftpd.conf

```
local_enable=NO
write_enable=NO
```

Запуск службы

Gentoo предоставляет централизованное место для служб, которые называются init scripts или скрипты инициализации. Эти скрипты находятся в каталоге /etc/init.d/ и имеют имя сервиса, который они стартуют. Инит скрипты используются для управления сервисами. Для запуска VSFTPD сервера наберите Code: Запуск vsftpd как службы

/etc/init.d/vsftpd start

Вы можете узнать больше о том что вы можете сделать этим скриптом инициализации набрав "/etc/init.d/vsftpd" без аргументов ("start" это аргумент, еще есть "stop" и "restart"))

Настройка скриптов конфигурации

Вы не хотите набирать `/etc/init.d/vsftpd start` каждый раз когда вы включаете свой компьютер? Gentoo предоставляет простой путь запуска скриптов инициализации при включении вашего компьютера. Чтобы настроить VSFTPD для запуска при каждой загрузке, наберите: Code: запуск vsftpd как службы загрузки

rc-update add vsftpd default

Если вы хотите узнать больше о rc-update наберите "man rc-update".

Также можно и через chkconfig: Code: запуск vsftpd как службы загрузки

chkconfig vsftpd on

Где размещать файлы

Файлы анонимного пользователя находятся в домашнем каталоге специального пользователя ftp (пользователя в системе). По умолчанию это - `/home/ftp`. Если вы хотите сменить его например на `/var/ftp`, вам нужно сделать следующие действия: `/var/ftp` Code: Смена размещения

rmdir /home/ftp

mkdir /var/ftp

chown ftp:ftp /var/ftp

ln -s /var/ftp /home/

Теперь можно что-то разместить в этом каталоге. Например, если вы хотите сделать доступными для других каталоги `disfiles` и `packages`, можно сделать следующие действия: Code: '

mv /usr/portage/disfiles /var/ftp/

ln -s /var/ftp/disfiles /usr/portage/

mv /usr/portage/packages /var/ftp/

ln -s /var/ftp/packages /usr/portage/

Примечание: VSFTPD автоматически запускается в chroot окружении, поэтому символические ссылки не работают. Именно для этого пришлось перенести каталоги `/usr/portage/disfiles` и `usr/portage/packages` и создать символические ссылки в тех местах где они должны находиться.

Но можно например монтировать нужные каталоги в `.../ftp` при помощи `fstab`. Для этого нужно создать каталоги: `/var/ftp/distfiles` и `/var/ftp/packages` Code: '

mkdir /var/ftp/distfiles

mkdir /var/ftp/packages

После чего пропишите в /etc/fstab следующее: Code: '
/usr/portage/distfiles /var/ftp/distfiles none ro,bind 0 0
/usr/portage/packages /var/ftp/packages none ro,bind 0 0

Клиенты

Есть множество программ, которые позволяют получать доступ к ftp серверу. Вот некоторые из них:

ftp
lftp - лучший консольный клиент
mozilla-firefox
nautilus
gftp

Jabber Server

Введение

Jabber - это открытый протокол, использующий XML, для быстрого обмена сообщениями и информацией о присутствии между любыми двумя абонентами в интернете. Благодаря своей уникальной расширяемости и гибкости, jabber способен поддерживать множество протоколов - ICQ, IRQ, MSN, RSS, Yahoo и др.

Установка пакетов

В качестве платформы для развертывания протокола jabber возьмем пакет jabberd. Он поддерживает множество возможностей - основные IM-протоколы ICQ, MSN, Yahoo, поддержку SSL-соединений и даже IPv6.

Сначала посмотрим, какие USE-флаги требует jabber: Code: Зависимости

emerge -pv jabberd

These are the packages that I would merge, in order:

Calculating dependencies ...done!

[ebuild N] net-im/jabberd-1.4.3-r5 -icq -ipv6 -ldap -msn -oscar -ssl -yahoo 0 kB

Total size of downloads: 0 kB

Включаем соответствующие флаги в установку (неважно, через переменную USE или через файл make.conf) и собираем: USE="icq ssl" emerge jabberd В качестве зависимости вытянется пакет jit - Jabber ICQ Transport, необходимый для поддержки протокола ICQ.

Конфигурация

Сначала добавьте пользователей, ответственных за администрирование сервера jabber, в группу jabber:

gpasswd -а имя_пользователя jabber

Все конфигурационные файлы, связанные с jabber, имеют формат XML и находятся в каталоге /etc/jabber.

Обязательно надо прописать имя хоста и РАЗРЕШИТЬ регистрации.

Запуск демона

Добавляем запуск сервера при старте системы:

rc-update add jabber default

Стартуем наш сервер: **/etc/init.d/jabber start**

Все должно работать :)

Выделенный сервер Counter-Strike

Рано или поздно у большинства админов локальных сетей возникает необходимость настраивать игровые сервера для пользователей. Одной из самых популярных сетевых игр на сегодняшний день является Counter-Strike:Source. Которая, к сожалению пока не приобрела достаточной популярности у нас в России. Мы рассмотрим установку сервера для игры CS 1.6. Более популярной в среде российских геймеров.

Необходимое программное обеспечение

Для установки CS-сервера нам понадобится только одна программа:halflife-steam
Code: emerge -pv halflife-steam

Calculating dependencies ...done!

[ebuild N] games-server/halflife-steam-2.0 3,431 kB

Total size of downloads: 3,431 kB

Steam - это утилита от Valve,позволяющая обновлять любую их игру через steam-network.

Установка и настройка

Ставим steam

Установка сводится к простому **emerge halflife-steam**.

Настройка установка hl

Steam установится в каталог /opt/halflife

ВАЖНО: в ноябре 2006 года в пакете была бага. Что характерно для всех продуктов получаемых через стим, они иногда забывают unix специфику. Непосредственно у меня, на steam не стоял бит исполняемости. И в cs было имя типа xxx.WAD (нужно wad, различается регистр) Code: команды

```
cd /opt/halflife  
chmod +x stream
```

Code: команды

```
./steam -command update -game cstrike -dir ./
```

Наблюдаем лог. Если никаких слов типа error нет - значит сервер установился.
Code: output

HLDS installation up to date

Предупреждение: Этот сервер работает только в LAN (класс C) и только со steam клиентами. Все остальное Варез

Файл сервер Samba в домене Win2k

Задача:

Организация файлового сервера

Авторизация пользователей через домен контроллер

Дано:

Red Hat 9.0

Samba 3.0.13

DC win 2003 server

Вступление

Взяться за написание данной статьи меня побудило то, что когда возникла задача о которой написано выше , то статьи описывающей решение данной проблемы полностью я не нашел вот и решил написать такую статью в которой было бы готовое решение.

Основная часть

Первым делом устанавливаем самбу. Решено было использовать последнюю версию 3.0.13. Установлено все это было из стандартного RPM пакета. Думаю с этим сложностей ни у кого не возникает, док по данному вопросу полно и потому начнем сразу править конфиг самбы. Ниже приведен окончательный работающий конфиг, может там и есть что лишнее, но после того как все заработало я убирать оттуда ничего не стал.

```
[global]
```

```
realm = bryusov.iasnet.ru
```

```
# Workgroup = имя NT-домена (или рабочей группы):
```

```
workgroup = DOMAIN
```

```
# NetBIOS-имя, под которым будет виден сервер остальным машинам сети.
```

```
netbios name = NAU
```

```
# Комментарий, появляющийся рядом с именем машины в "Сетевом окружении" Windows.
```

```
server string = Samba Server
```

```
# Следующий параметр влияет на безопасность. Hosts allow разрешает машинам с
```

```
# указанными IP-адресами присоединяться к Samba-серверу.
```

```
hosts allow = 172.18. 172.17. 127.
```

```
# если подставить %m то для каждой машины подключенной к Samba-серверу будет
```

```
# использоваться свой log-файл.
```

```
log file = /var/log/samba/log.smbd
```

```
# это кому скока не жалко
```

```
max log size = 500
```

```
# определяет, каким образом будет осуществляться проверка пароля (нам надо через DC)
```

```
security = domain
```

```
# Параметр Password server используется только совместно с опцией security = domain
```

```
password server = <IP домен контролера>
```

```
# для репликации всех доменов входящих в траст с вашим доменом
```

```
allow trusted domains = yes
```

```
# включаем поддержку шифрованных паролей.
```

```
encrypt passwords = yes
```

```
# Используя следующий параметр можно создать отдельную конфигурацию для каждой машины домена.
```

```
# Вместо пары символов %m при входе подставляется NetBIOS-имя машины.
```

```
# Я Такого не делал хотя поэкспериментировать можно.
```

```
# include = /usr/local/samba/lib/smb.conf.%m

# данные строчки можно не включать в работающий конфиг они определяют место
# хранения , порядок обновления Unix паролей и какой программой все это производится
smb passwd file = /etc/samba/smbpasswd
unix password sync = Yes
passwd program = /usr/bin/passwd %u
    passwd chat = *New*UNIX*password* %n\n *ReType*new*UNIX*password* %n\n
*passwd:*all*authentication*tokens*updated*successfully*

# В документации говорится, что с помощью этого параметра
# можно повысить производительность Samba-сервера.
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

# по дефолту слушаются все интерфейсы, здесь можно указать конкретно
interfaces = <ip или название интерфейса>

# строчки которых не было в стандартном конфиге и были добавлены руками для репликации NT-
юзеров:

winbind uid = 10000-20000
winbind gid = 10000-20000
winbind enum groups = yes
winbind enum users = yes

# Если вам нужны русские буквы в именах файлов, то раскомментируйте
# следующие 3 строки (заменяв KOI8-R на свою локаль)
# dos charset = CP866
# unix charset = KOI8-R
# display charset = KOI8-R

# описываем шары
[FILES]
comment = share
path = /share/FILES
public = no
writable = yes
valid users = DOMAIN\users
create mask = 0744

# настройка кириллицы по желанию

Все с конфигом самбы закончили, далее привожу конфиг /etc/krb5.conf
[logging]
default = FILE:/var/log/krb5libs.log
kdc = FILE:/var/log/krb5kdc.log
```

```
admin_server = FILE:/var/log/kadmind.log
```

```
[libdefaults]
```

```
ticket_lifetime = 24000
default_realm = PDC.DOMAIN.NAME.RU
dns_lookup_realm = false
dns_lookup_kdc = false
```

```
[realms]
```

```
DOMAIN.NAME.RU = {
    kdc = pdc.domain.name.ru:88
    admin_server = kerberos.domain.name.ru:749
    default_domain = domain.name.ru
}
```

```
[domain_realm]
```

```
.domain.name.ru = DOMAIN.NAME.RU
domain.name.ru = DOMAIN.NAME.RU
```

```
[kdc]
```

```
profile = /var/kerberos/krb5kdc/kdc.conf
```

```
[appdefaults]
```

```
pam = {
    debug = false
    ticket_lifetime = 36000
    renew_lifetime = 36000
    forwardable = true
    krb4_convert = false
}
```

Все остальное оставлено без изменений.

Теперь заводим самба сервер в домен, делаем со стороны домен контролера двусторонние доверительные отношения, запускаем `getent group` и видим что все работает.

Вот и все надеюсь полезной инфы достаточно для быстрой и эффективной настройки файл сервера.

Огромное спасибо всем кто писал статьи про самбу почти все их прочитал и почерпнул много полезного , в том числе не только для решения своих задач.

Руководство по настройке Samba в режиме PDC с использованием LDAP

Устанавливаемые пакеты

Установим нужные пакеты:

```
# USE="ldap acl ldapsam pam gdbm samba ssl tcpd winbind" emerge samba  
openldap acl nss_ldap pam_ldap
```

ВАЖНО: Версия nss_ldap должна быть не ниже 250-r1 (>=sys-auth/nss_ldap-250-r1)

OpenLDAP

Конфигурирование OpenLDAP

Для примера используется домен amber.global.com который является дочерним доменом домена global.com, который работает под управлением Win2003 Server.

Примечание: Не решенной задачей осталось создание доверительных отношений между доменами и включение домена amber в лес global.com

Файл: /etc/openldap/slapd.conf

```
include /etc/openldap/schema/core.schema  
include /etc/openldap/schema/cosine.schema  
include /etc/openldap/schema/inetorgperson.schema  
include /etc/openldap/schema/misc.schema  
include /etc/openldap/schema/nis.schema  
include /etc/openldap/schema/openldap.schema  
include /etc/openldap/schema/samba.schema  
pidfile /var/run/openldap/slapd.pid  
argsfile /var/run/openldap/slapd.args
```

```
access to dn.base=""  
by self write  
by * auth  
access to attr=userPassword  
by self write  
by * auth  
access to attr=shadowLastChange
```



```

        by self write
        by * read
access to *
        by * read
        by anonymous auth
#loglevel 1
database      ldbm
suffix        "dc=amber,dc=global,dc=com"
rootdn        "cn=Manager,dc=amber,dc=global,dc=com"
# Пароль rootpw лучше всего указывать в зашифрованном виде.
# Для генерации шифрованного пароля используйте утилиту slappasswd
# Например: slappasswd -h {MD5}
# rootpw = secret
rootpw        {MD5}Xr4ilOzQ4PCOq3aQ0qbuaQ==
directory     /var/lib/openldap-ldbm

index objectClass      eq
index cn                eq,subinitial
index sn                eq,subinitial
index uid               eq,subinitial
index displayName      eq,subinitial
index uidNumber         eq
index gidNumber         eq
index memberUID         eq
index sambaSID          eq
index sambaPrimaryGroupSID eq
index sambaDomainName   eq
Файл: /etc/openldap/ldap.conf
HOST 127.0.0.1
BASE dc=sanaa,dc=global,dc=com

```

Проверяем разрешения на каталоги /var/lib/openldap-*

```
ls -la /var/lib/
```

ВАЖНО:

```

drwx----- 2 ldap  ldap   104 Июн  8 18:31 openldap-data
drwx----- 2 ldap  ldap    72 Июн  8 18:31 openldap-ldbm
drwx----- 2 ldap  ldap    72 Июн  8 18:31 openldap-slurp

```

Запуск сервера OpenLDAP

пока работаем без шифрования трафика, так как сервер LDAP и SAMBA работают

на одном сервере, подправляем конфиг чтобы LDAP слушал 389 порт только на localhost.

Файл: /etc/conf.d/sldap

```
# conf.d file for the openldap-2.1 series
```

```
#
```

```
# To enable both the standard unciphered server and the ssl encrypted
```

```
# one uncomment this line or set any other server starting options
```

```
# you may desire.
```

```
#
```

```
# OPTS="-h 'ldaps:// ldap:// ldapi://%2fvar%2frun%2fopenldap%2fslapd.sock'"
```

```
OPTS="-h 'ldap://127.0.0.1'"
```

и запускаем OpenLDAP

/etc/init.d/slapd start

SAMBA

Миграция OpenLDAP

Существует 2 пути создания учетных записей пользователей: домен уже существует и создание нового домена. В первом случае самбу нужно настроить в режиме BDC, перенести все учетные записи, и потом изолировав PDC, перезапустить самбу в режиме PDC. Во втором случае все еще проще, самбу сразу запускаем в режиме PDC и создаем стандартные учетные записи при помощи замечательного пакета smbldap-tools.

emerge smbldap-tools

/etc/init.d/samba start

Code: configure.pl

```
-----
smbldap-tools script configuration
-----
```

Before starting, check

- . if your samba controller is up and running.
- . if the domain SID is defined (you can get it with the 'net getlocalsid')

- . you can leave the configuration using the Ctrl-c key combination
- . empty value can be set with the "." character

```
-----
```

Looking for configuration files...

Samba Configuration File Path [/etc/samba/smb.conf] >

The default directory in which the smbldap configuration files are stored is shown.

If you need to change this, enter the full directory path, then press enter to continue.

Smbldap-tools Configuration Directory Path [/etc/smbldap-tools/] >

Let's start configuring the smbldap-tools scripts ...

```
. workgroup name: name of the domain Samba act as a PDC
  workgroup name [amber] >
. netbios name: netbios name of the samba controler
  netbios name [neptun] >
. logon drive: local path to which the home directory will be connected
  (for NT Workstations). Ex: 'H:'
  logon drive [U:] >
. logon home: home directory location (for Win95/98 or NT Workstation).
  (use %U as username) Ex: '\\neptun\%U'
  logon home (press the "." character if you don't want homeDirectory) [\\%L\users\%U] >
. logon path: directory where roaming profiles are stored. Ex: '\\neptun\profiles\%U'
  logon path (press the "." character if you don't
  want roaming profile) [\\%L\Profiles\%a\%U] >
. home directory prefix (use %U as username) [/home/%U] >
. default users' homeDirectory mode [700] >
. default user netlogon script (use %U as username) [] >
  default password validation time (time in days) [45] > 900
. ldap suffix [dc=amber,dc=global,dc=com] >
. ldap group suffix [ou=Groups] >
. ldap user suffix [ou=Users] >
. ldap machine suffix [ou=Users] >
. ldap suffix [ou=ldap] >
. sambaUnixIdPoolDn: object where you want to store the next uidNumber
  and gidNumber available for new users and groups
  sambaUnixIdPoolDn object (relative to ${suffix}) [sambaDomainName=amber] >
. ldap master server: IP adress or DNS name of the master (writable) ldap server
  ldap master server [127.0.0.1] >
. ldap master port [389] >
. ldap master bind dn [cn=Manager,dc=amber,dc=global,dc=com] >
. ldap master bind password [] >
. ldap slave server: IP adress or DNS name of the slave ldap server: can also
  be the master one
  ldap slave server [127.0.0.1] >
. ldap slave port [389] >
. ldap slave bind dn [cn=Manager,dc=amber,dc=global,dc=com] >
. ldap slave bind password [] >
. ldap tls support (1/0) [0] >
. SID for domain amber: SID of the domain (can be obtained with 'net getlocalsid neptun')
  SID for domain amber [S-1-5-21-1918777035-593721947-2697221154] >
```

```
. unix password encryption: encryption used for unix passwords
  unix password encryption (CRYPT, MD5, SMD5, SSHA, SHA) [SSHA] > MD5
. default user gidNumber [513] >
. default computer gidNumber [515] >
. default login shell [/bin/bash] >
. default skeleton directory [/etc/skel] >
. default domain name to append to mail adress [] >
```

backup old configuration files:

```
/etc/smbldap-tools/smbldap.conf->/etc/smbldap-tools/smbldap.conf.old
```

```
/etc/smbldap-tools/smbldap_bind.conf->/etc/smbldap-tools/smbldap_bind.conf.old
```

writing new configuration file:

```
/etc/smbldap-tools/smbldap.conf done.
```

```
/etc/smbldap-tools/smbldap_bind.conf done.
```

Если мы не хотим, что бы у всех пользователей профили были перемещаемыми, то в файле /etc/smbldap-tools/smbldap.conf установим следующее значение: Файл: /etc/smbldap-tools/smbldap.conf

```
...
```

```
userProfile=""
```

```
...
```

Инициализируем каталоги самбы в LDAP:

smbldap-populate -a Administrator -k 0 -m 0

Конфигурация SAMBA

Файл: /etc/samba/smb.conf

[global]

```
workgroup = amber
```

```
netbios name = neptun
```

```
realm = amber.global.com
```

```
nt acl support = yes
```

```
acl compatibility = win2k
```

```
map acl inherit = yes
```

```
server string = Samba Server %v
```

```
interfaces = eth0
```

```
bind interfaces only = yes
```

```
hosts allow = 192.168.7. 127.
```

```
log file = /var/log/samba/log.%m
```

```
debug level = 9
```

```
max log size = 500
```

```
socket options = TCP_NODELAY SO_SNDBUF=8192 SO_RCVBUF=8192
```

```
security = user
```

```
os level = 250
```

```

passdb backend = ldapsam:"ldap://127.0.0.1/"
enable privileges = yes

passwd program = /usr/sbin/smbldap-passwd "%u"
passwd chat = *new*password* %n\n *new*password* %n\n *successfully*
passdb expand explicit = no
unix password sync = no
ldap passwd sync = no

ldap suffix = dc=amber,dc=global,dc=com
ldap admin dn = cn=Manager,dc=amber,dc=global,dc=com
ldap user suffix = ou=Users
ldap group suffix = ou=Groups
# Т.к. для самбы компьютеры и пользователи - одно и то же,
# и искать она в дальнейшем записи компьютеров будет в пользователях,
# то для избежания дальнейших проблем при добавлении рабочих станций
# к домену мы вместо следующей строки
#
#      ldap machine suffix = ou=Computers
#
# напомним другую:
      ldap machine suffix = ou=Users
      ldap idmap suffix = ou=Idmap
      idmap backend = ldapsam:ldap://127.0.0.1/
      idmap uid = 10000-20000
      idmap gid = 10000-20000

ldap delete dn = Yes
ldap ssl = no

add user script = /usr/sbin/smbldap-useradd -n -a "%u"
delete user script = /usr/sbin/smbldap-userdel "%u"

add group script = /usr/sbin/smbldap-groupadd -p "%g"
delete group script = /usr/sbin/smbldap-userdel "%g"

add user to group script = /usr/sbin/smbldap-groupmod -m "%u" "%g"
delete user from group script = /usr/sbin/smbldap-groupmod -x "%u" "%g"
set primary group script = /usr/sbin/smbldap-usermod -g "%g" "%u"

add machine script = /usr/sbin/smbldap-useradd -w "%u"

#PDC
domain master = yes
preferred master = yes
#BDC

```

```
# domain master = no
# preferred master = no
domain logons = Yes
```

```
logon script =
# Если хотите, что бы профили всех пользователей были перемещаемыми
# и хранились на сервере (со всеми гигабайтами фильмов и личных фотографий)
# то укажите такое значение следующего параметра:
#
# logon path = \\%L\Profiles\%a\%U
#
# Если вы не хотите гонять профили по сети, оставьте значение пустым,
# (но ни в коем случае не комментируйте эту строку, она просто получит
# значение по умолчанию), вот так:
logon path =
logon drive = U:
logon home = \\%L\users\%U
```

```
#===== Share Definitions =====
```

```
[netlogon]
```

```
comment = Network Logon Service
path = /var/lib/samba/netlogon
browseable = yes
guest ok = yes
writable = no
share modes = no
```

```
[Profiles]
```

```
admin users = admin
create mode = 600
directory mode = 700
path = /var/lib/samba/profiles
browseable = yes
guest ok = yes
writable = yes
```

```
[homes]
```

```
comment = Home Directories
browseable = no
read only = no
```

```
[public]
```

```
path = /pub
guest ok = yes
read only = no
```

```
[users]
path = /home/users
writable = yes
printable = no
```

Добавим запуск winbind с самбой (если нужно): Файл: /etc/conf.d/samba

```
...
daemon_list="smbd nmbd winbind"
...
```

Введём пароль рабочей станции:

smbpasswd -w secret

Введём контроллер домена, собственно в домен

```
net rpc join -S neptun -U Administrator
```

Настройка системы на авторизацию в LDAP

--ladserg 14:05, 28 июля 2006 (UTC) У меня честно говоря не получилось сделать авторизацию пользователей samba через LDAP без настройки поддержки авторизации системных пользователей в LDAP, пришлось настраивать и это.

Сначала поправим файл /etc/ldap.conf, приведя его примерно к следующему виду:

Файл: /etc/ldap.conf

```
host 127.0.0.1
base dc=amber,dc=global,dc=com
ldap_version 3
rootbinddn cn=Manager,dc=amber,dc=global,dc=com
bind_timelimit 10
bind_policy soft
pam_filter objectClass=posixAccount
pam_password exop
nss_base_passwd ou=Users,dc=tty,dc=perm,dc=ru?one
nss_base_shadow ou=Users,dc=tty,dc=perm,dc=ru?one
nss_base_group ou=Groups,dc=tty,dc=perm,dc=ru?one
nss_base_hosts ou=Hosts,dc=tty,dc=perm,dc=ru?one
nss_base_services ou=Services,dc=tty,dc=perm,dc=ru?one
nss_base_networks ou=Networks,dc=tty,dc=perm,dc=ru?one
```

```
nss_base_protocols    ou=Protocols,dc=tty,dc=perm,dc=ru?one
nss_base_rpc          ou=Rpc,dc=tty,dc=perm,dc=ru?one
nss_base_ethers       ou=Ethers,dc=tty,dc=perm,dc=ru?one
nss_base_netmasks     ou=Networks,dc=tty,dc=perm,dc=ru?one
nss_base_bootparams   ou=Ethers,dc=tty,dc=perm,dc=ru?one
nss_base_aliases      ou=Aliases,dc=tty,dc=perm,dc=ru?one
nss_base_netgroup     ou=Netgroup,dc=tty,dc=perm,dc=ru?one
ssl off
nss_reconnect_tries 4
nss_reconnect_sleeptime 1
nss_reconnect_maxsleeptime 16
nss_reconnect_maxconntries 2
```

Сим мы скажем `nss_ldap` где и как искать записи пользователей и групп.

Теперь создадим файл `/etc/ldap.secret` и при помощи любого текстового редактора в `plain/text` виде занесём туда пароль пользователя, который выше у нас указан в опции `rootbinddn`, например пароль `secret`:

Файл: `/etc/ldap.secret`
`secret`

Затем непременно установим на него нужные права:

Code: Установка прав на файл `/etc/ldap.secret`
#chmod 600 /etc/ldap.secret
#chown root:root /etc/ldap.secret

Далее приведём файл `/etc/pam.d/system-auth` к следующему виду:

Файл: `/etc/pam.d/system-auth`

```
auth    required    pam_env.so
auth    sufficient   pam_unix.so likeauth nullok
auth    sufficient   pam_ldap.so use_first_pass
auth    required     pam_deny.so

account  sufficient   pam_ldap.so
account  required     pam_unix.so

password required     pam_cracklib.so difok=2 minlen=8 dcredit=2 ocredit=2 retry=3
password sufficient   pam_unix.so nullok md5 shadow use_authok
password sufficient   pam_ldap.so use_authok
password required     pam_deny.so

session  required     pam_limits.so
session  required     pam_unix.so
session  required     pam_mkhomedir.so skel=/etc/skel/ umask=077
session  optional     pam_ldap.so
```


Обратите внимание на строку:

```
session required pam_mkhomedir.so skel=/etc/skel/ umask=077
```

Она заставляет систему создавать домашние каталоги для тех пользователей у которых они ещё не созданы, при этом в новый каталог помещается содержимое директории /etc/skel/ и задаётся маска каталога 0x700

После чего правим файл /etc/nsswitch.conf, приводя его к следующему виду:

Файл: /etc/nsswitch.conf

```
passwd:    files ldap
shadow:    files ldap
group:     files ldap

hosts:     files dns
networks:  files dns

services:  db files
protocols: db files
rpc:       ldap [NOTFOUND=return] db files
ethers:    ldap [NOTFOUND=return] db files
netmasks:  files
netgroup:  ldap [NOTFOUND=return] files
bootparams: files

automount: files
aliases:   files
```

Предупреждение: Ни в коем случае не добавляйте значение ldap к следующим базам: hosts, networks, protocols, services. Иначе вы рискуете не дождаться следующей загрузки системы.

Всё, теперь мы указали системе брать пользователей как из системных файлов, так и из LDAP.

Перезагрузим наш компьютер, дабы убедиться что система грузится нормально. Если система останавливается на загрузке udev, то смотрите ошибки в файле /etc/nsswitch.conf, может вы указали использовать ldap не в той базе.

Управление пользователями

Ранее мы установили пакет smbldap-tools, теперь рассмотрим возможность управления пользователями с его помощью.

Создание пользователя

smbldap-useradd [-o] [-a] [-b] [-w] [-i] [-u uid] [-g gid] [-G groups,,,]
[-n] [-d home] [-s shell] [-c geccos] [-m [-k]] [-t] [-P] [-A 0|1] [-B 0|1]
[-C sambaHomePath] [-D sambaHomeDrive] [-E sambaLogonScript] [-F
sambaProfilePath]
[-H sambaAcctFlags] [-N surname] [-S family name] [-M local mailAddress,,,]
[-T mailToAddress] [-?] user

Где:

- user - системное имя создаваемого пользователя
- o - add the user in the organizational unit (relative to the user suffix)
- a - is a Windows User (otherwise, Posix stuff only)
- b - is a AIX User
- w - is a Windows Workstation (otherwise, Posix stuff only)
- i - is a trust account (Windows Workstation)
- u - uid
- g - gid
- G - список групп пользователя, разделённых запятой.
- n - do not create a group
- d - домашний каталог пользователя (по умолчанию /home/имя_пользователя)
- s - оболочка пользователя (по умолчанию /bin/false)
- c - отображаемое в Windows имя пользователя
- m - создать домашний каталог и скопировать в него файлы из /etc/skel
- k - указать иной каталог, из которого будут копироваться файлы при создании домашнего каталога пользователя (используется с ключём -m)
- t - time. Wait 'time' seconds before exiting (when adding Windows Workstation)
- P - ends by invoking smbldap-passwd
- A - возможность менять пароль пользователем, значение 0 если нет, 1 если да
- B - пользователь должен поменять пароль, значение 0 если нет, 1 если да
- C - домашний каталог samba (например '\\PDC-SRV\homes')
- D - буква диска для монтирования домашнего каталога samba (например 'H:')
- E - скрипт, выполняемый при входе в систему
- F - каталог профиля пользователя (например '\\PDC-SRV\profiles\foo')
- H - sambaAcctFlags (samba account control bits like '[NDHTUMWSLKI]')
- N - настоящее имя пользователя (для русских ещё и отчество)
- S - фамилия пользователя
- M - local mailAddress (comma seperated)
- T - mailToAddress (forward address) (comma seperated)
- ? - отобразить помощь

Например создание пользователя ladserg:

smbldap-useradd -a -c 'Serg Alex Lad' -N 'Serg Alex' -S 'Lad' -s /bin/bash ladserg

К сожалению подружить smbldap-tools с русским мне не удалось, даже при использовании кодировки UTF-8.

Итак, в приведённом выше примере будет создан пользователь с системным именем ladserg, фамилией Lad, именем Serg Alex, оболочкой /bin/bash, домашним каталогом /home/ladserg. Флаг -a укажет, что пользователь также будет являться пользователем домена.

Изменение пароля

smbldap-passwd [-s] [-u] [-h] username

Где:

- username - имя пользователя
- h, -?, --help - показать помощь
- s - обновить только samba пароль
- u - обновить только UNIX пароль

Например:

smbldap-passwd ladserg

После чего дважды будет запрошен пароль.

Теперь можно попробовать зайти в систему под учётной записью только что созданного пользователя.

Модификация пользователя

**smbldap-usermod [-a] [-c comment] [-d home_dir] [-e expiration_date]
 [-g initial_group] [-r new_login_name] [-p passwd] [-s shell] [-u uid [-o]] [-x]
 [-A canchange] [-B mustchange] [-C smbhome] [-D homedrive] [-E scriptpath]
 [-F profilepath] [-G group,...] [-H acct-flags] [-N canonical_name]
 [-S surname] [-P] login**

Где:

- c - Полное имя
- d - Домашний каталог
- r - новое имя пользователя (cn, sn и dn будут обновлены)

- u - uid
- o - uid может быть не уникальным
- g - gid
- G - список групп пользователя, разделённых запятой.
- s - оболочка
- N - настоящее имя пользователя (для русских ещё и отчество)
- S - фамилия пользователя
- P - ends by invoking smbldap-passwd

For samba users:

- a - add sambaSAMAccount objectclass
- e - expire date ("YYYY-MM-DD HH:MM:SS")
- A - возможность менять пароль пользователем, значение 0 если нет, 1 если да
- B - пользователь должен поменять пароль, значение 0 если нет, 1 если да
- C - домашний каталог samba (например "\\PDC-SRV\homes")
- D - буква диска для монтирования домашнего каталога samba (например 'H:')
- E - скрипт, выполняемый при входе в систему
- F - каталог профиля пользователя (например "\\PDC-SRV\profiles\foo")
- H - sambaAcctFlags (samba account control bits like '[NDHTUMWSLKI]')
- I - disable an user. Can't be used with -H or -J
- J - enable an user. Can't be used with -H or -I
- M - mailAddresses (comma seperated)
- T - mailToAddress (forward address) (comma seperated)
- ?|-h - отобразить помощь

Например команда:

smbldap-usermod -A 1 ladserg

Позволит пользователю ladserg менять пароль. А команда:

smbldap-usermod -a slad-adm

Добавит к учётной записи пользователя slad-adm класс sambaSAMAccount, что сделает его пользователем samba.

Удаление пользователя

smbldap-userdel [-r|-R|-?] username

Где:

- r удалить домашний каталог
- R удалить домашний каталог с запросом на удаление каждого файла
- ? отобразить помощь

Например команда:

smbldap-userdel -r slad-adm

удалит пользователя slad-adm, и его домашний каталог.

Управление пользователями в оффтопике

Если вы вводите в домен компьютеры под управлением Windows, то вам пригодятся пара утилит, архив которых можно скачать отсюда:

<ftp://ftp.microsoft.com/Softlib/MSLFILES/SRVTOOLS.EXE>

Мониторинг работы системы

Введение

В данной статье описывается, как осуществить мониторинг разных системных и физических параметров компьютеров, работающих под GNU/Linux или другими UNIX.

Постановка задачи

Нужно осуществлять постоянный мониторинг основных параметров работы компьютеров (физических и системных) через сеть, собирать и выдавать эту информацию в удобной форме.

Для примера см: <http://monitoring.sourceforge.net/example.html>

К основным системным параметрам относят: загрузка процессоров (пользователями, системой, привилегированными программами, свободное процессорное время), использования оперативной памяти (и многие другие, всего больше 100), дисков, сетевого трафика. Для получения необходимых нам значений параметров работы системы используем программы df, sar и iptables или ipfw.

К физическим параметрам относят: температуру процессора и материнской платы, скорость вращения вентиляторов, напряжения питания. Для получения показателей датчиков материнских плат используется модуль ядра I2C и

программа `lm_sensors`.

Для передачи информации через сеть используем клиент-серверную модель и программу Net-Telnet. Базу данных строим пакетом `rrdtool`, информацию предоставляем через сервер `apache` и/или по почте. С помощью скриптов постараемся максимально упростить и автоматизировать поставленную задачу, это избавит вас от рутинной работы.

Программа работает по клиент-серверной технологии. На компьютерах, которые нужно мониторить, размещается часть программы, которая снимает различные показатели работы компьютера и выдает их по запросу через сеть. На одном административном компьютере осуществляется периодический опрос всех компьютеров, за которыми ведется наблюдение, данные записываются в специально созданную базу, и на её основе формируется `html` страница, которая экспортируется через `apache` для просмотра обычным браузером.

Теперь подробно о том, что нам нужно сделать:

Загружаем последнюю версию программы для мониторинга с сайта <http://monitoring.sourceforge.net/> Файл `monitoring-*.tar.bz2` размещаем в доступном месте, например в папке локального `ftp`: `/var/ftp/pub/linux`.

"Серверная" сторона

Разархивируйте `monitoring-*.tar.bz2`.

У Gentoo есть ебилд, для установки программы `monitoring` со всеми зависимостями. С сайта необходимо скачать `monitoring-*.ebuild.tar.bz2` и распаковать его в `/usr/local`. Далее выполните:

```
# emerge -pv monitoring
```

```
[ebuild N ] net-analyzer/monitoring-0.11 USE="-admin -apache2 -lm_sensors -server"
```

Для серверной стороны необходимо добавить `USE="server"`, а для поддержки мониторинга датчиков `USE="lm_sensors"`

```
# export "USE=lm_sensors server"
```

```
# emerge monitoring
```

Компиляция и установка `lm_sensors`, `sysstat`, `iptables`, `coreutils`, `xinetd` в случае использования Gentoo можно пропустить, потому что эти пакеты уже установлены как зависимости, и перейти сразу к их настройке.

Мониторинг физических параметров

Определяем на базе какого чипсета сделаны платы: материнская, видеокарта (некоторые имеют датчики) и прочие, заходим на страницы:

<http://secure.netroedge.com/~lm78/supported.html> и <http://www.lm-sensors.nu/~lm78/newdrivers.html> и ищем, есть ли драйверы для мониторинга этих плат под Линуксом. Если определили, что ваши чипсеты поддерживаются и драйвер стабильный, можете продолжить выполнение этого пункта.

Внимание, если версия ядра меньше за 2.4.9, нужно обновить ядро или воспользоваться старой версией программы! смотрите: i2c. Если ядро версии 2.4.*, вам дополнительно нужен СООТВЕТСТВУЮЩИЙ! модуль i2c-*.tar.gz, родной Линуксовый не подходит. В версии ядра 2.6.* все хорошо - отлично работает родной модуль ядра i2c.

Компиляция ядра

Подробную инструкцию по сборке нового ядра можно прочесть по ссылке "Ставим ядро 2.6, или Ядерная физика для домохозяйки. Версия 2.0"

Обратите особое внимание на пункты, которые добавляют к ядру GNU/Linux поддержку сенсоров:

Для поддержания ACPI:

Power management options (ACPI, APM) ---> ACPI (Advanced Configuration and Power Interface) Support --->

Для поддержки IPMI стандарта управления сенсорами.

Device Drivers ---> Character devices ---> IPMI --->

Включаем поддержку сенсоров в ядре

Device Drivers ---> I2C support --->

Выбираем алгоритмы, использующие ваши чипы

Device Drivers ---> I2C support ---> I2C Algorithms --->

Выбираем ваш чипсет (внимание, здесь нужно указать чипсет именно вашей системной платы, или видеокарты, см. ниже):

Device Drivers ---> I2C support ---> I2C Hardware Bus support --->

Device Drivers ---> I2C support ---> Miscellaneous I2C Chip support --->

Выбираем драйверы ко всем вашим сенсорам (внимание, именно тех, соответствующих вашим чипам, предыдущим пунктам):

Device Drivers ---> Hardware Monitoring support --->

Можно собрать необходимые драйверы как модули, хотя это несколько добавит работы при настройке (нужно будет запустить программу sensors-detect). Для мониторинга сети можно использовать netfilter (iptables) и прочие опции...

Если Ваше аппаратное обеспечения поддерживается, установите lm_sensors

Загружаем с: lm_sensors, свеженькую версию lm_sensors-*.tar.gz.
Разархивируем lm_sensors-*.tar.gz Заходим в директорию, которая создалась и выполняем:

make user

make user_install

depmod -a

Проверяем, есть ли строка /usr/local/lib в файле /etc/ld.so.conf, если нет, добавляем и выполняем команду:

ldconfig

Настройка lm_sensors

Выполняем команду

prog/mkdev/mkdev.sh

Чтобы правильно определить ваше оборудование и знать, какие модули нужно загрузить, выполним:

prog/detect/sensors-detect

именно этот скрипт и указывает, какие у вас чипсеты, и какие в них сенсоры!

Добавляем строки, созданные программой для modules.conf в /etc/modules.conf

Копируем prog/init/lm_sensors.init в каталог /etc/rc.d/init.d/lm_sensors (назначить права для исполнения).

Добавляем строки, которые создала программа prog/detect/sensors-detect с modprobe и sensors -s например, в конец файла /etc/conf.d/local.start - чтобы они загружались при старте системы

или можно запускать как сервис:

для Gentoo выполняем:

rc-update -a lm_sensors default

Только для клонов RedHat выполняем:

chkconfig --add lm_sensors

копируем prog/init/lm_sensors.sysconfig в /etc/sysconfig/lm_sensors

копируем etc/sensors.conf.eg в /etc/sensors.conf

копируем prog/sensors/sensors в каталог /usr/bin/sensors (предоставить права на выполнение)

Все теперь выполняют modprobe с параметрами, которые указала программа sensors-detect для загрузки модулей, потом выполняем:

sensors -s

Корректируем /etc/sensors.conf таким образом, чтобы названия напряжений, скорости вентиляторов и температур, именно Ваших чипов содержали соответственно: "Volt", "Fan", "Temp". При этом, названия меток допускают только малые и большие латинские буквы, цифры, знак "_", и не должны содержать знаков ",", "/" или "+"! Выполнение этих условий обязательно!!!

Пример /etc/sensors.conf можно найти в server/etc/sensors.conf

Например, для lm85*:

cat /etc/sensors.conf

```
...
# Voltage inputs
label in0  "VoltA1_5"   # AGP on Intel S845WD1-E
label in1  "Volt1_5"
label in2  "Volt3_3"
label in3  "Volt5"
label in4  "Volt12"
# Temperature inputs
label temp1 "TempCPU"
label temp2 "TempMB1"
label temp3 "TempMB2"
# Fan inputs
label fan1  "FanCPU"
label fan2  "FanSys1"
label fan3  "FanSys2"
label fan4  "FanSys3"
# PWM Outputs
label pwm1  "CPUF_PWM"
label pwm2  "SysF1_PWM"
label pwm3  "SysF2_PWM"
...
```

Теперь выполним команду sensors, и если у нас все получилось, то увидим примерно следующее:

sensors

```
lm85b-i2c-0-2e
Adapter: SMBus I801 adapter at c800
VoltA1_5: +1.48 V (min = +1.42 V, max = +1.58 V)
Volt1_5:  +1.50 V (min = +1.45 V, max = +1.60 V)
Volt3_3:  +3.33 V (min = +3.13 V, max = +3.47 V)
Volt5:    +5.10 V (min = +4.74 V, max = +5.26 V)
Volt12:   +12.31 V (min = +11.38 V, max = +12.62 V)
FanCPU:   3360 RPM (min = 3000 RPM)
```

TempCPU: +36C (low = +10C, high = +60C)
TempMB1: +33C (low = +10C, high = +45C)
TempMB2: +33C (low = +10C, high = +45C)
CPUF_PWM: 255
SysF1_PWM: 255
SysF2_PWM: 77
vid: +1.525 V (VRM Version 9.1)

Если у вас названия напряжений, скорости вентиляторов и температуры не содержат соответственно: "Volt", "Fan", "Temp" - обязательно отредактируйте файл /etc/sensors.conf как показано выше.

Мониторинг системных параметров

Установка sysstat

Загружаем последнюю версию пакета sysstat с сайта <http://perso.wanadoo.fr/sebastien.godard/> или <ftp://ibiblio.org/pub/Linux/system/status/>

Розархивируем файл sysstat-*.tar.bz2 в /usr/src

Заходим в созданный каталог и выполняем:

```
# make config          #Отвечаем на вопросы только 'y' или 'n'
# make
# make install
```

Теперь из командной строки пробуем:

```
# sar -A 1 1 |grep Average:
```

```
Average:  proc/s
Average:    0.00
Average: cswch/s
Average: 258.00
Average:  CPU  %user  %nice  %system  %iowait  %idle
Average:  all  1.90   0.00   0.40   0.00  97.70
.....
Average:  runq-sz  plist-sz  ldavg-1  ldavg-5  ldavg-15
Average:    0     83   0.06   0.15   0.16
```

Проверка df

```
# df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda3	2007996	411772	1596224	71%	/
/dev/sda1	132206	10981	114399	43%	/boot
/dev/sda5	248895	4256	231789	2%	/tmp
/dev/sda6	5863496	2556028	3307468	64%	/var

```

/dev/sda7      6843432  5128048  1715384  75% /usr
/dev/sda8      1976492  1048740   827348  56% /home
/dev/sda9      5863496  4479752  1383744  77% /usr/portage/distfiles
none          254752    0   254752  0% /dev/shm
/usr/portage/distfiles 5863496  4479752  1383744  77% /var/ftp/pub/linux/distfiles
/usr/portage/packages 6843432  5128048  1715384  75% /var/ftp/pub/linux/packages

```

Если Вам выдало:

bash: df: command not found

установите df из пакета coreutils.

Мониторинг сети

Мониторинг сети основывается на подсчете байт, попадающих в правила сетевых экранов - iptables или ipfw, и поэтому они достаточно гибкие, но является зависимым от конфигурации сетевого экрана.

Если Вы хотите осуществлять мониторинг сети: просмотрите server/etc/show_network.conf (/etc/monitoring/show_network.conf) – это список графиков, которые Вы получите. Он должен отвечать названиям графиков в admin/etc/network (/etc/monitoring/network).

Linux iptables

Нужно откорректировать server/etc/show_iptables_acc.conf (/etc/monitoring/show_network.conf)

1 колонка – название графика (должна отвечать show_network.conf, но с “in_*.N” или “out_*.N”, где N – некоторый идентификатор, например число)

2 колонка – название таблицы

3 колонка – название цепочки с таблицы

4 колонка – порядковый номер правила с цепочки

Это всё должно отвечать Вашим правилам!

Простые примеры экранов находятся в server/firewall/* и если подходит Вам, можете использовать их.

Проверьте работу /usr/bin/show_iptables.sh. Результат найдёте в /home/monitoring

FreeBSD ipfw

Для BSD, надо откорректировать server/bin/show_ipf_acc.conf:

1 колонка – название графика

2 колонка – номер правила

3 колонка – порядковый номер правила в ipfw show

Это должно отвечать Вашим правилам!

Проверьте работу /usr/bin/show_ipfw.sh Результат в /home/monitoring

Некоторые важные системные настройки

Если нет xinetd – установите его. Загляните в

server/etc/host.allow (/etc/host.allow):

```
# BEGIN allow services for monitoring
show_hdisk.sh:127.0.0.1 # IP ком п'ютера что мониторит
show_sensors.sh:127.0.0.1 # IP ком п'ютера что мониторит
show_info.sh:127.0.0.1 # IP ком п'ютера что мониторит
show_system.sh:127.0.0.1 # IP ком п'ютера что мониторит
show_network.sh:127.0.0.1 # IP ком п'ютера что мониторит
# END allow services for monitoring
```

Также некоторые из них вы можете изменить в server/bin/show_* и строки "only_from", "bind" и "disable" в server/xinetd.d/show_*

```
$ cat server/xinetd.d/show_hdisk
# default: on
# description: The showdisk server show disk useg on the server.
#   It dont uses authentication !!!!!!!!!!!!!!!
service df
{
    socket_type      = stream
    wait            = no
    user            = monitoring
    bind            = 127.0.0.1          # IP сетевого интерфейса, на котором мониторят
    server          = /usr/bin/show_hdisk.sh
    only_from       = 127.0.0.1          # IP комп'ютера, который мониторит
    disable         = no                 # Вкл./Выкл.
}
```

Откройте доступ к следующим портам: 9045/tcp 9046/tcp 9047/tcp 9048/tcp 9049/tcp с административного компьютера в случае наличия сетевых экранов. Если Вы хотите осуществить установку вручную, пропустите следующий пункт.

Автоматическая инсталляция

Для автоматической установки отредактируйте следующие строки в **monitoringinstall.sh (/usr/sbin/monitoringinstall.sh):**

```
# Edit this first:
# all command will execute user:
runuser=monitoring
rungroup=monitoring
```

```
# cron cfg directory
cronpath=/etc/cron.d
# init script locations:
crond=/etc/init.d/crond
# Edit only for server side:
# init script location:
xinetd=/etc/init.d/xinetd
# program location:
dfpath=/bin/df
sarpath=/usr/bin/sar
sensorspath=/usr/bin/sensors
```

Теперь с правами суперпользователя выполните:

```
# ./monitoringinstall.sh --server
```

В Gentoo просто:

```
# monitoringinstall.sh --server
```

Вы можете пропустить пункт следующий пункт и перейти к „административной” части системы.

Ручная установка

Копируем следующие файлы:

```
cp server/etc/show_iptables_acc.conf /etc/monitoring #for GNU/Linux
cp server/etc/show_ipfw_acc.conf /etc/monitoring      #for BSD
cp server/etc/show_network.conf /etc/monitoring
cp server/xinetd.d/show_hdisk /etc/xinetd.d/
cp server/xinetd.d/show_network /etc/xinetd.d/
cp server/xinetd.d/show_sensors /etc/xinetd.d/
cp server/xinetd.d/show_system /etc/xinetd.d/
cp server/bin/show_hdisk.sh /usr/bin
cp server/bin/show_iptables.sh /usr/bin               #for GNU/Linux
cp server/bin/show_ipfw.sh /usr/bin                   #for BSD
cp server/bin/show_network.sh /usr/bin
cp server/bin/show_sar.sh /usr/bin
cp server/bin/show_sensors.sh /usr/bin
cp server/bin/show_system.sh /usr/bin
```

Добавим следующие строки к /etc/service:

```
network      9045/tcp      # show network info
info         9046/tcp      # show server info
```

df	9047/tcp	# show disk info
sys	9048/tcp	# show system info
sensors	9049/tcp	# show sensors info

Создадим специального пользователя:

```
groupadd monitoring
useradd -g monitoring -d /home/monitoring -s /bin/bash -c monitoring monitoring
mkdir /home/monitoring
chmod 700 /home/monitoring
chown -R monitoring:monitoring /home/monitoring
```

В случае использования политик SELinux необходимо также добавить:

```
# cat /etc/security/selinux/src/policy/users
```

...

```
# BEGIN monitoring selinux:
user monitoring roles user_r;
# END monitoring selinux.
```

и дополнительно выполнить:

```
# cd /etc/security/selinux/src/policy
# make load
```

Перезагрузим xinetd:

```
# /etc/init.d/xinetd restart
```

и добавьте строки в crontab:

```
*/10 * * * * monitoring /usr/bin/show_iptables.sh # для GNU/Linux
*/10 * * * * monitoring /usr/bin/show_ipfw.sh # для BSD
*/1 * * * * monitoring /usr/bin/show_sar.sh
```

Перезагрузим crond:

```
# /etc/init.d/crond restart
```

Все вышесказанное нужно сделать на КАЖДОМ сервере, который вы хотите мониторить...

"Административная" сторона

Для начала убедитесь что все хорошо со стороны серверов

Для каждого сервера проверяем доступность необходимых сервисов:

```
# telnet server_name 9045
```

```
Trying serverIP...
Connected to serverIP.
Escape character is '^]'.
in_ftp          0
out_ftp         0
in_http         0
out_http        0
in_other        249
out_other       27
Connection closed by foreign host.
```

telnet server_name 9046

```
Trying serverIP...
Connected to server_name.
Escape character is '^]'.
Intel(R) Celeron(R) CPU 2.40GHz | 504 Mb
Connection closed by foreign host.
```

telnet server_name 9047

```
Trying serverIP...
Connected to server_name.
Escape character is '^]'.
/dev/sda3      2007996  411772  1596224  71% /
/dev/sda1      132206   10981   114399  43% /boot
/dev/sda5      248895    4256   231789   2% /tmp
/dev/sda6      5863496  2556028  3307468  64% /var
/dev/sda7      6843432  5128048  1715384  75% /usr
/dev/sda8      1976492  1048740   827348  56% /home
/dev/sda9      5863496  4479752  1383744  77% /usr/portage/distfiles
Connection closed by foreign host.
```

telnet server_name 9048

```
Trying serverIP...
Connected to server_name.
Escape character is '^]'.
Average:   proc/s
Average:    3.98
Average:   cswch/s
Average:   308.00
Average:   CPU   %user   %nice   %system   %iowait   %idle
Average:   all   1.76   0.00   0.47   0.00   97.76
.....
Average:   runq-sz  plist-sz  ldavg-1  ldavg-5  ldavg-15
Average:    1     88   0.05   0.21   0.21
Connection closed by foreign host.
```

telnet server_name 9049

Trying serverIP...

Connected to server_name.

Escape character is '^]'.

lm85b-i2c-0-2e

Adapter: SMBus I801 adapter at c800

Volt1_5: +1.48 V (min = +1.42 V, max = +1.58 V)

VoltCore: +1.50 V (min = +1.45 V, max = +1.60 V)

Volt3_3: +3.33 V (min = +3.13 V, max = +3.47 V)

Volt5: +5.10 V (min = +4.74 V, max = +5.26 V)

Volt12: +12.25 V (min = +11.38 V, max = +12.62 V)

CPU_Fan: 3377 RPM (min = 3000 RPM)

fan2: 0 RPM (min = 0 RPM)

fan3: 0 RPM (min = 0 RPM)

fan4: 0 RPM (min = 0 RPM)

TempCPU: +32 C (low = +10 C, high = +50 C)

TempBoard: +30 C (low = +10 C, high = +45 C)

TempRemot: +30 C (low = +10 C, high = +40 C)

CPU_PWM: 255

Fan2_PWM: 255

Fan3_PWM: 77

vid: +1.525 V (VRM Version 9.0)

Connection closed by foreign host.

Установка необходимых программ

Разархивируйте monitoring-*.tar.bz2.

Для Gentoo есть ебилд, для установления программы monitoring со всеми зависимостями с сайта стоит брать monitoring-*.ebuild.tar.bz2 и распаковать его в /usr/local дальше выполните:

emerge -pv monitoring

[ebuild N] net-analyzer/monitoring-0.11 USE="-admin -apache2 -lm_sensors -server"

Для административной стороны необходимо добавить USE=admin, а для поддержки apache-2* USE=apache2

export "USE=admin apache2"

emerge monitoring

Компиляцию и установление rrdtool, apache, Net-Telnet в случае использования

Gentoo можно пропускать, потому что эти пакеты уже будут установлены как зависимости, а переходить сразу к их настройке.

Инсталляция rrdtool

Базой данных, для хранения информации и инструмент генерации графиков, используется пакет rrdtool (round robin database tool). Зкачать пакет можно отсюда: <http://rrdtool.eu.org> или <http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/> Подробно ознакомиться с этим пакетом можно здесь: <http://www.bog.pp.ru/work/rrdtool.html> (Российская). В принципе, программа самостоятельно создаст необходимые базы так, что дополнительных знаний не нужно и следующий пункт Пример расчёта настройки базы RRD можно пропустить.

Пример расчёта настройки базы RRD

Расчёт настроек создания базы RRD файлы /etc/monitoring/*.rrd.cf Количество "слов" в файле будет соответствовать количеству графиков в разных масштабах времени. например для расчёта подобия MRTG: 5-ти минутный - данные за последние 2 дня: $x\text{-доля}=0.5$ количество отсчётов на ячейку=1 количество ячеек= $2*24*12=576 + \min + \max + \text{avg} + \text{cur} \sim 650$

0.5:1:650

30-ти минутный - дание за последние две недели: $x\text{-доля}=0.5$ количество отсчётов на ячейку= $30/5=6$ количество ячеек= $2*7*24*2=672 + \min + \max + \text{avg} + \text{cur} \sim 750$

0.5:6:750

2-ух часовой - данные за последние два месяца: $x\text{-доля}=0.5$ количество отсчётов на ячейку= $60*2/5=24$ количество ячеек= $2*31*24/2=744 + \min + \max + \text{avg} + \text{cur} \sim 850$

0.5:24:850

1-но дневный - данные за последние два года: $x\text{-доля}=0.5$ количество отсчётов на ячейку= $60*24/5=288$ количество ячеек= $2*366=732 + \min + \max + \text{avg} + \text{cur} \sim 900$

0.5:288:900

Таким образом имеем:

```
# cat /etc/monitoring/5min.rrd.cf
```

0.5:1:650 0.5:6:750 0.5:24:850 0.5:288:900

Инсталляция Net-Telnet

Данная версия программы для работы через сеть требует Net-Telnet perl модуль (можно при желании использовать и netcat): из-за этого нужно установить perl (что

пожалуй уже сделано). Далее Net-Telnet perl модуль берем и устанавливаем отсюда: <http://cpan.perl.org/modules/by-module/Net/> название файлу Net-Telnet-*.tar.gz

Инсталляция Apache

Нужно установить и настроить apache сервер, дополнительно можете добавить (admin/etc/apache/monitoring.conf):

```
# cat /etc/apache2/vhosts.d/monitoring.conf
### /etc/apache2/vhosts.d/monitoring.conf
### $Id: monitoring.conf,v 0.11 2006/09/28 16:27:12 hse Exp $
###
### For Monitoring *.shtml
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
### For authentication:
<Directory "/var/www/localhost/htdocs/administration/monitoring">
    AllowOverride All
    Options Includes
    <IfModule mod_access.c>
### For password authentication:
#     AuthUserFile /var/www/localhost/htdocs/administration/passwd_file
#     AuthGroupFile /dev/null
#     AuthName Administrative-information
#     AuthType Basic
#     Require valid-user
### For host/network authentication:
        Order deny,allow
        Deny from all
        Allow from 127.0.0.1
#     Allow from .cluster.linux
    </IfModule>
</Directory>
```

Добавляем пользователя для просмотра веб страницы:

```
# htpasswd -bcm /var/www/localhost/htdocs/administration/passwd_file username password
```

Если все хорошо идем дальше.

Установка программ для мониторинга

В файл admin/etc/host (/etc/monitoring/host) вносим перечень серверов, которые нужно мониторить (имена должны разрешаться в IP адреса). Корректируем под

свои потребности файл `admin/etc/monitoring.conf (/etc/monitoring/monitoring.conf)`

путь по которому установлено скрипт:

`installpath=/usr/`

путь к веб страницы:

`apachehtmldir=/var/www/html`

`webdirpath=/administration/monitoring`

путь к файлу с перечнем серверов для мониторинга:

`confpath=$installpath/etc/host`

путь к временных файлов:

`diskinfopath=/tmp/monitoring/disk.tmp`

`sarinfopath=/tmp/monitoring/sar.tmp`

`sensorsinfopath=/tmp/monitoring/sensors.tmp`

путь где установлен пакет rrdtool:

`rrdtoolpath=/usr/bin/rrdtool`

почтовый адресов:

`emailvalue=root@localhost`

По желанию (или за потребностью) правим другие файлы с каталога `etc/ (/etc/monitoring/)`:

diskusage memusage quantity cpu memory systemload - содержимое файлов должно отвечать тем параметрам которые Вы хотите мониторить. Все параметры должны выводиться командой `sar -A 1 1 |grep Average`: Названия файлов отвечают названиям графиков и баз которые создадутся и должны быть записаны в файле `system`. Значение параметров принадлежащих одному файлу должны быть одного типа!

system - первая колонка определяет какие графики получите и каждый ее элемент отвечать файлу из этого же каталога (см. выше). Вторая указывает через какие периоды времени база `rrdtool` будет принимать данные. Третья и четвертая колонка определяет минимальное и максимальное значение параметра которое еще может быть записано в базу. Пятая колонка определяет легенду (размерность).

colors - файл с номерами цветов в 16-ричной системе.

config - содержит имена конфигурационных файлов из этого же каталога отвечающие веб страницам что создаются (можно добавлять свои, но тогда их должны создать!).

disk - содержит названия графиков, периоды времени через которые база `rrdtool` будет принимать данные, минимальные и максимальные значение параметров которые мониторите, легенду (размерность).

network - содержит названия графиков, периоды времени через которые база

rrdtool будет принимать данные, минимальные и максимальные значение параметров которые мониторите, легенду (размерность).

sensors - содержит названия графиков, периоды времени через которые база rrdtool будет принимать данные, минимальные и максимальные значение параметров которые мониторите, легенду (размерность).

disk.rrd.cf network.rrd.cf sensors.rrd.cf system.rrd.cf - описывают структуру базы данных.

disk.msg network.msg sensors.msg system.msg - содержит информацию которую получите почтой в критических ситуациях.

Если хотите осуществить установку вручную, пропустите следующий пункт.

Автоматическая установка программ

Теперь можете воспользоваться скриптом для инсталляции. Для этого нужно откорректировать следующие строки в файле monitoringinstall.sh (/usr/sbin/monitoringinstall.sh) или пропустить этот пункт и следовать дальнейшей инструкции для установки в ручную.

имя пользователя и группы:

runuser=monitoring

rungroup=monitoring

директория где расположены cron настройки:

cronpath=/etc/cron.d

путь к инициализационному скрипту

crond crond=/etc/init.d/crond

директория где находятся настройки apache

apachconfdir=/etc/apache/conf

файл конфигурации apache:

apachconffile=apache.conf

путь к инициализационному скрипту apache

apached=/etc/init.d/apache

Теперь с правами рута выполните (нужно подождать 1 минуту, если не установили мониторинг сети, и 10 минут после установки мониторинга сети на последнем сервере, прежде чем выполнять следующую команду):

./monitoringinstall.sh --admin

В Gentoo просто

monitoringinstall.sh --admin

Если видите ошибки, наверное надо:

- 1 поправить файлы конфигурации в admin/etc/ (/etc/monitoring)
- 2 посмотреть в /home/monitoring/hostname/err/...
- 3 удалить файлы с /home/monitoring/hostname/ и перезапустить скрипт
- 4 удалить базы данных с /var/db/monitoring/... и перезапустить скрипт
- 5 bin/net_telnet.pl читает только 500 строк входа. Если в Вас больше, отредактируйте його:
line 32: while [\$i ne 500]

Тогда попробуйте еще, но добавьте имена сервисов, под рутом выполните:

```
# ./monitoringinstall.sh --admin=disk
# ./monitoringinstall.sh --admin=sensors
# ./monitoringinstall.sh --admin=system
# ./monitoringinstall.sh --admin=network
```

Если где-то допустили ошибку, можно просто выполнить monitoringuninstall.sh и попытаться еще раз. Можете пропустить следующий пункт.

Ручная установка программ

Устанавливаем скрипты для мониторинга, генерации базы данных и веб страницы скопировав с каталога monitoring/admin в каталог: /usr/ (\$INSTALL_PATH), используя опцию -p для сохранения прав. Выполняем:

```
# cp -p admin/bin/* $INSTALL_PATH/bin/
# mkdir /etc/monitoring
# cp -p admin/etc/* /etc/monitoring/
# mkdir $INSTALL_PATH/share/monitoring
# cp -p admin/share/* $INSTALL_PATH/share/monitoring/

# mkdir -p /var/www/localhost/htdocs/administration/monitoring
# chmod 755 /var/www/localhost/htdocs/administration/monitoring/
# groupadd -g 1111 monitoring
# useradd -u 1111 -g monitoring -d /home/monitoring -s /bin/bash -c monitoring
monitoring
# chmod -R 700 /home/monitoring
# chown -R monitoring:monitoring /home/monitoring
# chown -R monitoring:monitoring /var/www/localhost/htdocs/administration/monitoring/
```

И тогда выполняйте (подождите 1 минуту, если не устанавливали мониторинг сети, и 10 минут после установки мониторинга сети на последнем сервере,

прежде чем выполнять следующую команду):

```
# su monitoring
```

```
$ $INSTALL_PATH/bin/net.vert.1.sh
```

```
$ $INSTALL_PATH/bin/net.vert.10.sh
```

создаются базы данных в каталоге /var/db/monitoring/имя_сервера/ для всех серверов с файлу /etc/monitoring/host и дополнительная конфигурация в /home/monitoring

Выполняем:

```
$ $INSTALL_PATH/bin/png_graph.sh disk -1week 30min
```

```
$ $INSTALL_PATH/bin/png_graph.sh disk -1month 2hour
```

```
$ $INSTALL_PATH/bin/png_graph.sh disk -1year 1day
```

```
$ $INSTALL_PATH/bin/png_graph.sh network -1week 30min
```

```
$ $INSTALL_PATH/bin/png_graph.sh network -1month 2hour
```

```
$ $INSTALL_PATH/bin/png_graph.sh network -1year 1day
```

```
$ $INSTALL_PATH/bin/png_graph.sh sensors -2day 5min
```

```
$ $INSTALL_PATH/bin/png_graph.sh sensors -1week 30min
```

```
$ $INSTALL_PATH/bin/png_graph.sh sensors -1month 2hour
```

```
$ $INSTALL_PATH/bin/png_graph.sh sensors -1year 1day
```

```
$ $INSTALL_PATH/bin/png_graph.sh system -2day 5min
```

```
$ $INSTALL_PATH/bin/png_graph.sh system -1week 30min
```

```
$ $INSTALL_PATH/bin/png_graph.sh system -1month 2hour
```

```
$ $INSTALL_PATH/bin/png_graph.sh system -1year 1day
```

Если видите ошибки, наверное нужно:

1 поправить файлы конфигурации в /etc/monitoring

2 посмотреть в /home/monitoring/hostname/err/...

3 удалить файлы с /home/monitoring/hostname/ и перезапустить скрипт

4 удалить базы данных с /var/db/monitoring/... и перезапустить скрипт

5 bin/net_telnet.pl читает только 500 строк входа. Если в Вас больше, отредактируйте его:

```
line 32: while [$i ne 500]
```

Тогда попробуйте еще раз.

В каталоге /var/www/localhost/htdocs/administration/monitoring создаются начальная страница и каталоги с веб страницами для каждого сервера.

Если не возникло больше ошибок, используйте следующий файл для автоматизации всей работы с crond:

```
$ exit
```

```
# cat /etc/cron.d/monitoring
```

```

SHELL=/bin/bash
PATH=/bin:/usr/bin
MAILTO=root
HOME=/home/monitoring
LANG=POSIX

```

```

# Begin server part (may be you wont monitoring "admin" computer too)

```

```

*/1 * * * * monitoring /usr/bin/show_sar.sh
*/10 * * * * monitoring /usr/bin/show_iptables.sh      #GNU/Linux
#*/10 * * * * monitoring /usr/bin/show_ipfw.sh        #BSD
# End server part

```

```

# Begin disk admin part

```

```

*/30 * * * * monitoring /usr/bin/png_graph.sh hdisk -1week 30min
0 */2 * * * * monitoring /usr/bin/png_graph.sh hdisk -1month 2hour
0 0 * * * * monitoring /usr/bin/png_graph.sh hdisk -1year 1day
# End disk admin part

```

```

# Begin network admin part

```

```

*/30 * * * * monitoring /usr/bin/png_graph.sh network -1week 30min
0 */2 * * * * monitoring /usr/bin/png_graph.sh network -1month 2hour
0 0 * * * * monitoring /usr/bin/png_graph.sh network -1year 1day
# End network admin part

```

```

# Begin system admin part

```

```

*/5 * * * * monitoring /usr/bin/png_graph.sh system -2day 5min
*/30 * * * * monitoring /usr/bin/png_graph.sh system -1week 30min
0 */2 * * * * monitoring /usr/bin/png_graph.sh system -1month 2hour
0 0 * * * * monitoring /usr/bin/png_graph.sh system -1year 1day
# End system admin part

```

```

# Begin sensors admin part

```

```

*/5 * * * * monitoring /usr/bin/png_graph.sh sensors -2day 5min
*/30 * * * * monitoring /usr/bin/png_graph.sh sensors -1week 30min
0 */2 * * * * monitoring /usr/bin/png_graph.sh sensors -1month 2hour
0 0 * * * * monitoring /usr/bin/png_graph.sh sensors -1year 1day
# End sensors admin part

```

```

# Begin vert admin part

```

```

*/1 * * * * monitoring /usr/bin/net.vert.1.sh

```

```
*/10 * * * * monitoring /usr/bin/net.vert.10.sh  
# End vert admin part
```

Выполняем:

```
# /etc/init.d/crond restart
```

Если все хорошо

Теперь заходим Вашим любимым браузером на страницу <http://servername/administration/monitoring/index.html>. Здесь уже можем выбрать ссылки на сервер, который вас интересует :)

Добавление новых серверов

Если Вам когда-то придется добавить новый сервер, достаточно будет сделать следующее:

1. Выполнить для него пункт 2 („Серверная” сторона)
 2. Добавить его имя в файл `/etc/monitoring/host`
- Всё остальное создастся само !!! Наслаждайтесь :)))

Альтернативы

Если Вы недовольны реализацией мониторинга таким образом (по моему мнению достаточно неплохое решение ;)) выберите себе лучшую. Вот некоторый перечень альтернатив или дополнений :=)

Angel Network Monitor <http://www.paganini.net/angel/>

Autostatus <http://www.angio.net/consult/autostatus/>

Cacti <http://cacti.net>

HiWAYs <http://www.hiways.org/>

MARS <http://www.altara.org/mars.html>

Mon <http://www.kernel.org/software/mon/>

Monit <http://www.tildeslash.com/monit/>

Nagios <http://www.nagios.org>

Netup (French) <http://www.pasteur.fr/units/sis/netup/>

NocMonitor <http://www2.discpro.org/nocmon/>

NodeWatch <http://www.skendric.com/nodewatch/>

Penemo <http://www.communityprojects.org/apps/penemo/>

PIKT <http://pikt.org/>

RITW http://www.terravista.pt/Ancora/1883/ritw_e.html
RRDWorld <http://oss.oetiker.ch/rrdtool/rrdworld/index.en.html>
Scotty <http://wwwhome.cs.utwente.nl/~schoenw/scotty/>
Spong <http://spong.sourceforge.net/>
Sysmon <http://www.sysmon.org/>
ZABBIX <http://www.zabbix.com>
ZEUS <http://www.zeus.com/>

Настройка X

Настройка русских шрифтов в X

Введение

Введение временное

Ну вот и начался этот проект, призванный восполнить достаточно широкий пробел в документации по настройке шрифтов в Иксах. Надеюсь многочисленные русские пользователи проявят некую сознательность и отпишут сюда пару строчек.

Кто принимает участие

Macil - линуксоид из Ярославля. Обитает на yar.lancktelecom.ru/talk, forum.median.ru, а также Macil [at] jabber.ru

Wanderer - пользователь Linux из Москвы. Всерьез занимаюсь изучением и использованием Linux последние 4 года, до этого просто приглядывался ;-). На Gentoo пересел почти год назад и ничуть не жалею и даже наоборот, очень этим доволен. Связаться со мной можно посредством e-mail ([wanderer_mg \[at\] mail.ru](mailto:wanderer_mg@mail.ru)) или jabber ([wanderer \[at\] jabber.ru](mailto:wanderer@jabber.ru)).

Две системы рисования шрифтов

Исторически в X Window System существует 2 системы рисования шрифтов. Родная (core) и Xft.

Родная система – прямой потомок системы рисования шрифтов, введенной в X11R1, выпущенной в 1987 году. Изначально она могла рисовать только монохромные растровые шрифты, но на протяжении всех этих лет предпринимались более-менее успешные попытки заставить ее работать с векторными шрифтами.

Xft не является неотъемлемой частью X, хотя возможно будет ей в будущем. Это значит, что она распространяется не со всеми реализациями X. Она проектировалась для работы с векторными шрифтами, и делает это достаточно эффективно. В отличие от родной системы рисования, Xft поддерживает anti-aliasing и sub-pixel rasterisation. Кроме того, она позволяет приложениям полностью контролировать то, как будут рисоваться шрифты, обеспечивая возможность создания WYSIWYG приложений.

Нужно ли уточнять что эти две подсистемы совершенно не совместимы на уровне API? Из-за достаточно анархичного развития X в последнее время, возникла большая путаница в определениях и версиях используемых библиотек.

//Источник: вольный перевод файла README.fonts

Xft

Введение в Xft

Xft это библиотека, которая предоставляет клиентский API для рисования шрифтов. В настоящее время версия Xft 2.0 использует библиотеки fontconfig и freetype для выбора шрифтов и их рисования, соответственно. Xft 1.0 использовала внутренний механизм конфигурации, что было не очень хорошей идеей так как KDE и Pango "заимствовали" механизм конфигурации шрифтов и создали еще большую путаницу. Подробности можно посмотреть [\[здесь\]](#).

Нужно запомнить 3 вещи:

fontconfig - для конфигурации

freetype - для рисования

XFT - связывает все во едино

Freetype - библиотека рисования шрифтов

[Freetype] - совершенно независимый от кого-либо проект по созданию библиотеки рисования шрифтов. Библиотека предоставляет достаточно простой API, но поскольку обычные приложения заинтересованы в более высокоуровневых сервисах, в чистом виде не используется. Вместе с X идет модифицированная freetype, предназначенная для поддержки векторных шрифтов родной системой рисования. Я не вижу смысла даже пытаться ее компилировать (она компилируется в модуль freetype), так как большинство приложений перешли на использование Xft и путанице со шрифтами приходит конец. Рендеринг векторных шрифтов очень сложный процесс, и к сожалению, freetype не может его выполнять до конца и в некоторых шрифтах появляются не очень приятные артефакты. Но ничего лучшего (открытого и бесплатного) пока нет... Как обычно, существует 2 версии freetype: freetype1 и freetype2. Отличия их в том, что freetype1

может рисовать только шрифты формата TrueType. Сложно найти современный дистрибутив, где отсутствует freetype2.

Теория рендеринга векторных шрифтов

Что такое hinting

На английском много информации о hinting можно найти тут <http://www.myfonts.com/activity/hinting/>

Не уверен, однако, что можно тут публиковать такие ссылки.

Это — незавершённая статья. Вы можете помочь проекту, исправив и дополнив материал.

Что такое anti-aliasing

Это методы избавления от артефактов, возникающих при отрисовке векторного изображения в растовое. Когда векторное изображение (не дискретное, к примеру, формула $y=2*x$) отрисовывается в растовое, то в результате погрешностей и того, что цвет каждого пикселя строго зависит от формулы, возникают артефакты. То есть наклонную линию мы будем видеть ступенчато, а не плавно. Методы anti-aliasing, в основном, базируются на отрисовке изображения в большее разрешение и приведение к обычному путём билинейной фильтрации, т.е. цвет пикселя будет строиться из цветов нескольких пикселей большого изображения. Шрифты после применения anti-aliasing выглядят более гладко, т.к. более точно соответствуют "векторной формуле".

Это — незавершённая статья. Вы можете помочь проекту, исправив и дополнив материал.

Шрифты из Microsoft Windows

Установка шрифтов Microsoft Windows имея доступ в интернет

Если есть доступ к интернету, то можно просто установить пакет corefonts: emerge corefonts

Настройка родной подсистемы

Настройка Xft

Все настройки Xft хранятся в файлах /etc/fonts/fonts.conf и /etc/fonts/local.conf. Все

изменения нужно производить в файле `/etc/fonts/local.conf`. Изменение файла `/etc/fonts/fonts.conf` не рекомендуется, т.к. при обновлении пакета `fontconfig` он будет перезаписан и все настройки потеряны.

Отключение сглаживания для шрифтов меньше определенного размера

По умолчанию сглаживание действует для всех шрифтов, но для шрифтов маленького размера его лучше отключить, т.к. смотрится это почти всегда плохо.

Ниже приведен пример конфига `/etc/fonts/local.conf`, в котором указана отмена сглаживания для шрифтов, размер которых меньше 14. Code: `/etc/fonts/local.conf`

```
<?xml version="1.0"?>
<!DOCTYPE fontconfig SYSTEM "fonts.dtd">
<!-- /etc/fonts/local.conf file for local customizations -->
<fontconfig>

  <!-- Enable sub-pixel rendering -->

  <match target="font">
    <edit name="rgba" mode="assign"><const>rgb</const></edit>
  </match>
  <match target="font">
    <test name="size" compare="less_eq">
      <double>14</double>
    </test>
    <edit name="antialias" mode="assign">
      <bool>>false</bool>
    </edit>
  </match>
  <match target="font">
    <test name="pixelsize" compare="less_eq">
      <double>14</double>
    </test>
    <edit name="antialias" mode="assign">
      <bool>>false</bool>
    </edit>
  </match>

</fontconfig>
```

Настройка Qt и Gtk

Для Qt

```
emerge qt qtconfig
```

Для Gtk

```
emerge gtk-theme-switch switch2
```

Настройка размера шрифтов в X***Введение***

Столкнулся с проблемой (и не я один) размера шрифтов.

Процесс сталкивания выглядел так:

```
~ $ emerge xorg-x11
```

```
...
```

```
~ $ emerge kde
```

```
...
```

```
startx
```

Попал я, естессна сразу в KDE. И увидел очень мелкие шрифты, очень мелкие, похожие на точки. Примерно такие:

Естессна, первое, что я сделал - полез в центр управления и установил размер, соответствующий моему зрению. Им оказался 40. Немало. Тогда я понял, что что-то неправильно. Порылся в инете, не нашел ничего. Смирился. Однако вскоре узнал, что та же проблема у меня и в KDM, только там изменить размер немного сложнее. Изменил. Но проблему это не решило, потому что нашлись еще всякие шибко умные проги, которые пользовались собственными настройками шрифтов. Их (настройки) искать оказалось очень сложно и пришлось брать быка за рога. Итак,

Суть проблемы

Она состоит в том, что шрифты ориентированы на определенное физическое разрешение экрана - dpi (точек на дюйм). В винде по дефолту оно идет равным 96. X11 решил, что это слишком сложно не знаю для кого и, как выяснилось позже, выставил себе аж целых 25 dots per inch. В переводе в метрическую систему это около 1 точки на миллиметр. Соответственно, экран разрешением 1024x768 должен быть шириной в метр, высотой 77 см, чтоб читалось все как надо. Ну а поскольку реально экран у нас меньше, мы видели то, что мы видели.

Неправильный ход мысли

но полезный совет. Увеличить размер шрифта для KDM можно в этом файле:
/usr/kde/3.5/share/config/kdm/kdmrc

Но это не выход, как я уже говорил. Кстати, GDM настраивается в другом файле, его я не знаю, потому что пришел к следующему пункту:

Правильный ход мысли

Надо изменить разрешение, те самые dpi. Где? Как?

Меняем dpi

Тут, кстати, возможны варианты. Если Вы запускаете иксы из консоли по мере необходимости командой `startx`, то Вам сюда. Если иксы стартуют сами при загрузке системы - то сюда. В идеале стоит проделать оба варианта.

Для команды `startx`

Лезем туда, где этот стартикс лежит и правим его

~ \$ **`find /usr -name startx`**

У меня он оказался в `/usr/bin`, и с вероятностью 99,(9)% там же он будет и у Вас.

Вписать в него надо всего ничего параметр `-dpi 96` в ключ `defaultserverargs`

`defaultserverargs="-dpi 96"`

Те параметры, которые там есть не удаляйте, у них свое назначение, этот просто допишите к ним.

В принципе, на этом все. Можно

~ \$ `startx`

Жизнь удалась.

Для пользователей KDM, XDM и тому подобного

Вооружитесь калькулятором. В файле `xorg.conf` Вам надо изменить параметр `DisplaySize`. По умолчанию, он устанавливается равным разрешению (помните, я говорил в начале статьи), но ваш-то монитор меньше. Можно померять линейкой, можно посчитать, какие значения ему присвоить:

$$X = x * 25,4 / \text{dpi}$$

$$Y = y * 25,4 / \text{dpi}$$

где X, Y - значения, которые надо присвоить `DisplaySize`

x, y - разрешение монитора (x - ширина, y - высота)

dpi - то физическое разрешение, которое должно получиться

У меня при разрешении 1024x768 получилось так:

~ \$ vi /etc/X11/xorg.conf

```
Section "Monitor"
    Identifier "Monitor0"
    VendorName "HSD"
    ModelName "HSD150PX17-A"
    DisplaySize 270.9 203.2 # mm
EndSection
```

Вывод

Ерунда какая, ничего сложного, если сразу заглянуть в xorg.conf и почитать комментарии, то этого можно избежать. Но я не избежал и, кстати, долго с этим мучился.

Вывод-то в чем: будьте внимательнее!

P.S.

В Gnome есть такая настроечка, как dpi, сам не видел, но знающие люди говорят, что есть и рекомендуют ее тоже выставить на 96 или то, которое принято везде.

Хорг Х11 и прозрачность

Введение

Это руководство поможет в установке X.org на Linux/Gentoo машине, хотя общие настройки должны работать и на любом другом дистрибутиве.

Шаг за шагом

Установка X.org

Установите последнюю версию X.org с помощью команды `emerge xorg-x11`

После компиляции необходимо настроить X.org путем редактирования `/etc/X11/xorg.conf`

Если у вас было установлено XFree86 то рабочий `/etc/X11/XF86Config` будет автоматически использоваться для работы X.org. Для обеспечения целостности рекомендуется переименовать `/etc/X11/XF86Config` в `/etc/X11/xorg.conf`.

Для загрузки X.org выполните команду startx

Установка X Composite Extension

Откройте /etc/X11/xorg.conf для редактирования nano -w /etc/X11/xorg.conf пользуйте ваш любимый редактор ;)

Добавьте следующие строки в любом месте конфиг файла Файл: /etc/X11/xorg.conf

```
Section "Extensions"
    Option "Composite" "Enable"
    Option "RENDER" "Enable"
EndSection
```

При следующей перезагрузке X.org у вас будет включен Composite Extension. Это расширение включает предварительную отрисовку вне экрана, чтобы на экран выводилось уже законченное изображение. Это позволяет использовать различные эффекты (например прозрачность и тени).

Для достоверности можно проверить лог файл grep COMPOSITE /var/log/Xorg.0.log
(II) Initializing built-in extension COMPOSITE

Включение теней и настоящей прозрачности

KDE 3.4 или более поздняя

Начиная с версии 3.4 KDE поддерживает прозрачность через Xorg. На заголовке любого окна щелкните правой кнопкой мыши, выберите "настройка поведения окна" и в списке слева надите пункт полупрозрачность. Поставьте флажок "использовать прозрачность/тени", после чего станут доступны множество настроек прозрачности (активного, неактивного окна, окна при перетаскивании, плавные переходы прозрачности и т.д.) и теней.

xcompmgr и transset

Многие WM еще не обладают поддержкой новых функций X.org, так что нам придется использовать дополнительные инструменты для наших эффектов. Вначале нам надо установить xcompmgr

```
echo "x11-misc/xcompmgr ~x86" >> /etc/portage/package.keywords
```

```
emerge xcompmgr
```


Затем установим transset

```
echo "x11-misc/transset ~x86" >> /etc/portage/package.keywords
```

emerge transset

Теперь у нас есть все необходимое, чтобы включить тени, затенение и настоящую прозрачность. Запустите из Xterm `xcompmgr xcompmgr -c` Эта команда позволит вам увидеть эффекты от запущенного `xcompmgr`, если они вам не понравятся, то `CTRL+C` в консоли позволит вам убить процесс, а если они вам нравятся, то надо убить процесс и запустить его в фоновом режиме (сзади консоли): `xcompmgr -c &`
Примечание: символ `&` позволяет запустить `xcompmgr` в фоновом режиме, но при этом если убить сам терминал - процесс тоже сдохнет, чтобы этого не произошло можно использовать утилиту `nohup`: `nohup xcompmgr -c &` При этом весь вывод будет направлен в `nohup.out`

Таким образом `xcompmgr` включит "мягкие" тени. Если будет использован флаг `-s` то будут "грубые" тени.

Рабочий пример для `xcompmgr` с тенью и затенением `xcompmgr -cCfF -r7 -o.65 -l-10 -t-8 -D7 &`

Теперь о прозрачности. Выполните в консоли `transset` Теперь ваш курсор изменился, вы можете кликнуть на окне которое хотите сделать прозрачным. Если запустить `transset` с цифровым значением от 0 до 1 то это укажет насколько прозрачным должно быть окно

0 - полностью прозрачно

1 - не прозрачно

Попробуйте выполнить `transset 0.2`

Для возвращения окна назад просто выполните `transset` и кликните по прозрачному окну.

Как это выглядит

Изображение взято на <http://www.grebowiec.net/archives/xorg4.html>

Выключение эффектов

Для выключения всех эффектов выполните **`killall xcompmgr`**

Примечание

NVIDIA

При использовании NVIDIA видео карточек рекомендуется добавить в /etc/X11/xorg.conf нижеприведенные строки для того, чтобы можно было использовать расширение "Composite" и увеличить производительность при его использовании. Файл: /etc/X11/xorg.conf

```
Section "Device"
```

```
...
```

```
Option "RenderAccel" "true"
```

```
Option "AllowGLXWithComposite" "true"
```

```
...
```

```
EndSection
```

ВАЖНО: При использовании RenderAccel у вас могут возникнуть проблемы со стабильностью, так как эта опция экспериментальная и использовать ее вам предлагается на свой страх и риск.

При использовании AllowGLXWithComposite у вас могут возникнуть проблемы с opengl приложениями. Могут появляться артефакты, так как glx и composite не рекомендуется использовать одновременно.

ATI

При использовании ATI видео карточек совместно с ATI драйверами, дополните /etc/X11/xorg.conf опцией backingstore, для включения поддержки backing store, механизма с помощью которого пиксельные данные закрытого окна запоминаются X сервером и снимает необходимость посылки сообщений к X клиенту когда надо обновить изображение Файл: /etc/X11/xorg.conf

```
Section "Device"
```

```
...
```

```
Option "backingstore" "true"
```

```
...
```

```
EndSection
```

Заключение

Эти разработки находятся в самой начальной стадии и мы можем использовать их уже в X.org версии 6.8. По мере развития WM и приложений будет возможность воспользоваться новым функционалом X.org в полную меру без установки

дополнительных приложений. Некоторые программы, kwin к примеру, уже портируются.

Xorg. Переключение раскладок

Переключение USA/RUS + Scroll Lock Led

В секции InputDevice для клавиатуры надо дописать следующее:

```
Section "InputDevice"
    Identifier      "Keyboard1"
    Driver"kbd"
    Option          "AutoRepeat"  "500 30"
    Option "XkbModel"    "pc105"
    Option "XkbLayout"   "us,ru(winkeys)"
    Option "XkbOptions"  "grp:alt_shift_toggle, grp_led:scroll"
EndSection
```

Раскладки будут переключаться по Alt+Shift, с индикацией на Scroll Lock Led

Если, конечно, не забыть, что в Section "ServerLayout" должно быть прописано InputDevice "Keyboard1". По умолчанию там как раз Keyboard0, так что будьте внимательны.

а если вы захотите что-бы CTRL-C CTRL-V работали во всех раскладках так-же четко как в микрософт виндовс - то этого увы вам никто не расскажет :) - надо править запутанные конфиги xkb

Возможно, вариант русской раскладки winkeys у вас не заработает. Тогда измените следующую строку:

Option "XkbLayout" "us,ru"

и добавьте

Option "XkbVariant" ",winkeys"

Обратите внимание на запятую, она указывает на то, что вариант "winkeys" применяется именно к русской раскладке!

Установка Xgl

Введение

Проект Xgl позволяет ускорять посредством подсистемы OpenGL вывод

примитивов X сервера, а также обеспечивать ускорение composite и render. На данный момент мы можем получить быструю работу с прозрачностью, тенью и отрисовкой окон.

Более подробное руководство на английском языке размещено по адресу en:HOWTO XGL

Инсталяция

Xorg

Нам понадобится последняя версия Xorg. Необходимо размаскировать нужные пакеты и установить Xorg-7.0emerge xorg-x11 Примечание: Дополнительно можно почитать статью Настройка Portage

glitz

Так же необходимо скачать из CVS последнюю версию glitz. cvs -d:pserver:anoncvs@cvs.freedesktop.org:/cvs/cairo co glitz Установить ее.autogen.sh && ./configure --prefix=/usr/ && make && make install

glxcompmgr

Для работы с расширением composite скачаем утилиту glxcompmgr cvs -d:pserver:anoncvs@cvs.freedesktop.org:/cvs/xorg co app/glxcompmgr Скомпилируем и установимautogen.sh && ./configure --prefix=/usr/ && make && make install

metacity

Так уж сложилось, что пока glxcompmgr может себя показать во всей красе только с оконным менеджером metacity, да, он способен работать и с kwin, но при этом запуск kdesktop вызовет прекращение нормальной работы glxcompmgr. metacity

MesaLib

Если мы хотим использовать composite, необходимо собрать библиотеку libGL.so.1.2 из пакета MesaLib-6.4.1 с патчем mesa-glx-x11-render-texture-3.diff, который лежит в каталоге glxcompmgr.patch -p0 <mesa-glx-x11-render-texture-3.diff && ./configure --prefix=/usr/ && make Нет необходимости переустанавливать весь пакет т.к. нас интересует только файл ./lib/libGL.so.1.2. Заменим старый файл.cp ./lib/libGL.so.1.2 /usr/lib/opengl/xorg-x11/lib/

Xgl

Теперь скачаем. `cvs -d:pserver:anoncvs@cvs.freedesktop.org:/cvs/xorg login`

CVS password: нажмите Enter

`cvs -d:pserver:anoncvs@cvs.freedesktop.org:/cvs/xorg co -r xgl-0-0-1 xserver`

Код находится в `./hw/xgl`

И установим сам Xgl сервер. `./configure --prefix=/usr/ --enable-xglserver --enable-glx --enable-xkb && make && make install`

Запуск сервера для видеокарт nvidia

"Голый" сервер

Убедимся, что мы используем `libGL.so.1.2` от `nvidiaeselect` `opengl set nvidia` Для запуска сервера в отдельном окне откройте свой любимый терминал и выполните команду `Xgl :1 -ac -accel xv -accel glx:pbuffer -screen 800x600 &` Если сервер не запускается, то, возможно, необходимо отключить расширение `xkb`, добавив опцию `-kb` `Xgl :1 -kb -ac -accel xv -accel glx:pbuffer -screen 800x600 &` Также могут возникнуть проблемы с шрифтами, тогда путь к ним надо указать вручную `Xgl :1 -kb -ac -accel xv -accel glx:pbuffer -fp /usr/share/fonts/misc/ -screen 800x600 &` Для запуска сервера в полноэкранном режиме можно использовать опцию `-fullscreen` `Xgl :1 -kb -ac -accel xv -accel glx:pbuffer -fp /usr/share/fonts/misc/ -fullscreen &`

Для запуска на ATI картах с установленными драйверами `fglrx` необходимо вместо `... -accel xv...` Использовать `... -accel xv:pbuffer...`

Запускаем glxcompmgr

Сначала для примера запустим `xterm` и `metacity` из терминала. `export DISPLAY=:1xterm &metacity &` После этого необходимо переключиться на использование библиотеки `libGL.so.1.2` из пакета `MesaLibeselect` `opengl set xorg-x11` После этого можно запускать `glxcompmgr.glxcompmgr shadow wobbly &` После этого уже должен быть виден интересный эффект на запущенном `xterm'e`.

KDE, GNOME

Для запуска `kde` или `Gnome` необходимо запустить сначала `metacity` затем `kde(Gnome)` и в последнюю очередь `glxcompmgr`.

Возможные проблемы

Возможно у вас не получится запустить сервер т.к. файл `/usr/lib/libGL.so.1` будет

указывать не на `/usr/lib/OpenGL/nvidia/lib/libGL.so.1.2`, а на `/usr/lib/libnvidia-tls.so.1`. Поменяйте симлинк и сервер запустится.

Для переключения раскладки подойдет программа `xruskb`.

XGL

Введение

Xgl это X сервер, который использует OpenGL для прорисовки объектов. Вместе с композитным оконным менеджером Compiz или новой версией Metacity позволяет получить превосходное качество 3D и 2D эффектов рабочего стола. Xgl базируется на расширении Mesa, `GLX_EXT_texture_from_pixmap`, который представлен только в версиях Mesa 6.5 и старше.

На the XOrg mailing list предлагается написать спецификацию `GLX_EXT_texture_from_pixmap` расширения совместно с NVIDIA. Бета версия драйверов уже выпущенных NVIDIA, версия 9625, поддерживают это расширение, это означает что Xgl уже не нуждается в использовании Compiz на NVIDIA картах, кроме этого использование данной конфигурации приветствуется. Для информации об этом методе использования Xgl, пожалуйста ознакомьтесь на en:HOWTO nVidia GL Desktop Effects

Кроме того, Compiz тесно связан с GNOME-компонентами, Xgl и Compiz работают прекрасно во всех окружениях рабочих столов (KDE/GNOME/Xfce/*box). Если Вы всё же не используете GNOME, будьте готовы к установке необходимых библиотек и утилит.

Предупреждение: Данное программное обеспечение ещё не является стабильным, и хотя оно почти всегда работает, всё же используйте его с осторожностью.

Системные требования

см. the article on video card support under Xgl.

Установка Xgl Примечание: Установка должна быть произведена на системе с работающей X сессией используя эмулятор терминала типа `konsole` или `gterm`, выполнение этого условия необходимо для запуска `gconf-editor` или `gset-compiz` для конфигурации Compiz в конце установки.

Требования к программному обеспечению

Обновите Ваше локальное дерево портажей.

emerge --sync

Вам необходима модульная версия XOrg (версия 7.x). Для установки модульной версии XOrg ознакомьтесь с официальной Gentoo документацией или соответствующей статьёй wiki.

en:HOWTO Modular Xorg

Официальное руководство

Модульный XOrg (версия 7.x) был опубликован в стабильной ветке дерева портажей, поэтому Вы можете пропустить эту главу если Ваш Gentoo полностью обновлен.

Compiz использует рабочую оболочку GNOME GConf для конфигурации, и некоторые из опций конфигурации будут приведены здесь. Наиболее легким и популярным способом конфигурации является программа gconf-editor, одна из приложений GNOME. Конечно, и менее известная KDE программа KConfigEditor также может использоваться для конфигурации. Правда для этой программы пока не существует ebuild файла, но Вы можете загрузить исходный код этой программы с extragear.kde.org. Пожалуй самый нежелательный из способов это редактирование gconf конфигурационных файлов вручную, что является скучным и неблагодарным занятием.

Если Вы хотите использовать gconf-editor, то его необходимо установить.

emerge -nav gconf-editor

Также Вам необходимо установить Subversion с включенной поддержкой webdav. Вы должны отключить флаг nowebdav. Добавьте добавьте при необходимости следующую строку в /etc/portage/package.use: Примечание: Тут используется двойное отрицание в использовании USE флага.

```
echo "dev-util/subversion -nowebdav" >> /etc/portage/package.use
```

затем,

```
emerge -nav subversion
```

Загрузка Portage оверлея

Существуют два метода для создания требуемого оверлея; оверлей можно загрузить используя svn утилиту или использовать lautan утилиту которая будет управлять оверлеями вместо Вас. Используя lautan производить обновление

системы проще, так что решать Вам. Вне зависимости, что Вы будете использовать результат будет одним и тем же.

Метод по умолчанию: использование Layman утилиты

Если вы ещё не установили Layman, сделайте это сейчас.

emerge -nav layman

Также Вам необходимо добавить следующую строку в Ваш make.conf.

echo "source /usr/portage/local/layman/make.conf" >>/etc/make.conf

Если Вы используете Eix Caching System, удостоверьтесь что Вы используете самую новую версию; старые версии не поддерживают чтение внешних файлов make.conf.

Измените "nocheck" переменную на значение "yes" в конфигурационном файле утилиты layman. (/etc/layman/layman.cfg)

Затем добавьте оверлей с помощью Layman утилиты:

layman -f

layman -a xeffects

если вы используете layman версии 1.0.7 или позже, то оверлеи с неправильными атрибутами будут проигнорированы. Используйте опцию -k для обхода этой ситуации.

layman -k -a xeffects

Альтернативный метод: Создание оверлея вручную

Если Вы хотите обслуживать Ваш оверлей самостоятельно, то сделайте следующее. Иначе (если используется layman), вы можете перейти к главе Размаскирование необходимых пакетов. Примечание: Лучше всего не смешивать оверлеи и использовать их отдельно друг от друга.

Перейдите в Вашу директорию оверлеев

cd /usr/local/overlays

Загрузите последнюю версию из CoffeeBuzz's Subversion репозитория в директорию 'xgl-coffee'

svn co <http://svn.xgl-coffee.org/xgl-coffee/trunk> xgl-coffee

Теперь Вы создали оверлей дерева портежей в /usr/local/overlays/xgl-coffee.

Добавьте следующий путь в переменную PORTDIR_OVERLAY в /etc/make.conf.

Файл: /etc/make.conf

PORTDIR_OVERLAY="\${PORTDIR_OVERLAY} /usr/local/overlays/xgl-coffee"

Размаскирование необходимых пакетов

Много пакетов маркированы как нестабильные и должны быть размаскированы чтобы их можно было установить из дерева портажей. Добавьте следующие строки к файлу /etc/portage/package.keywords или /etc/portage/package.unmask

Файл: /etc/portage/package.keywords

Зависимости

dev-util/git

media-libs/glitx

media-libs/mesa

x11-apps/mesa-progs

sys-apps/man

x11-libs/cairo

dev-python/pycairo

x11-libs/qt

x11-misc/util-macros

x11-libs/glibproto

x11-apps/xvinfo

x11-apps/xlsclients

x11-libs/libwnck

x11-misc/xwinwrap

virtual/xft

gnome-base/gconf

gnome-base/libgnomeui

x11-libs/gtk+

dev-libs/glib

x11-libs/libdrm

#XGL пакет

x11-base/xgl

старые версии Compiz

```
x11-wm/compiz-quinnstorm
x11-misc/compiz-quinnstorm-plugins
x11-wm/compiz
x11-wm/cgwd
x11-misc/csm
x11-misc/cgwd-themes
x11-misc/cgwd-themes-extra
x11-misc/gset-compiz
x11-misc/compiz-manager
```

новая ветка Beryl для Compiz

```
x11-wm/beryl-core
x11-plugins/beryl-plugins
x11-misc/beryl-manager
x11-misc/beryl-settings
x11-wm/emerald
x11-misc/emerald-themes
```

Удостоверьтесь что флаги glitz, pdf и png включены в переменную USE для Cairo. Возможно Вам понадобится добавить следующую строку к файлу package.use:

Файл: /etc/portage/package.use

```
x11-libs/cairo glitz pdf png
```

Установка Xgl

Новые бета драйверы с закрытым исходным кодом от nVidia (версия 1.0.9625) уже включают в себя поддержку XGL. Поэтому не нужно более никаких установок. Если вы не хотите использовать новейшие бета драйвера, то следуйте к главе "не-nVidia пользователи". Данный способ работает на nVidia картах при этом установка и использование новых nVidia драйверов намного проще. Если Вы хотите использовать новейшие бета драйвера тогда Вы должны перейти к главе "nVidia Пользователи".

Не-nVidia пользователи

Во-первых Вы должны установить Cairo. Используйте опцию --oneshot чтобы не записывать этот пакет в список world (не захламляй свой world список).

```
# emerge --oneshot --ask --verbose cairo
```

Переинсталируй pango и gtk+ с использованием новоустановленного Cairo пакета.

```
# emerge --oneshot --ask --verbose pango gtk+
```

Далее у Вас есть выбор между beryl и compiz ветками:

compiz это оригинальное приложение написанное Давидом Равеманом (David Reveman) из Новел (Novell). Это приложение уже включено в дерево портажей.

beryl это снимок Quinnstorm's Ubuntu кодовой базы с некоторыми экспериментальными плагинами. Формально эта ветка известна под названием compiz-quinnstorm. Использует emerald как window decorator. это отдельный пакет не включенный в дерево портежей.

Проверьте что флаги dbus и svg включены в переменную USE для Compiz. Возможно Вам потребуется добавить следующую строку в package.use: Файл: /etc/portage/package.use

```
x11-wm/compiz dbus svg
```

Проверьте что пакеты glproto и libdrm установлены. Установите их если это требуется:

```
# emerge --ask --verbose --oneshot --noreplace glproto libdrm
```

В конечном итоге Вы должны будете установить Xgl и Mesa, которые автоматически скомпилируют и установят Compiz а также все его плагины и Xgl сервер.

```
# emerge --ask --verbose --oneshot --noreplace mesa
```

Опция 1: Для использования vanilla compiz:

```
# emerge --ask --verbose xgl compiz
```

Опция 2: Для использования beryl мета пакета:

```
# emerge --ask --verbose xgl beryl-core
```

Примечание: Установка beryl также устанавливает декоратор окон emerald и его темы. Таким образом emerald не надо устанавливать отдельно.

Примечание: NVIDIA пользователи пожалуйста переустановите nvidia-drivers после установки Xgl

Примечание: Если не удастся скомпилировать xgl, проверьте заплатку на <http://forums.xgl-coffee.org/viewtopic.php?t=178>

Примечание: Если не удастся скомпилировать beryl из за dbus, попробуйте обновить пакет dbus

Примечание: Если не удастся скомпилировать xgl и компиляция обрывается на

fbmmx.c или fbedge.c, попробуйте удалить флаг "-fforce-addr" из переменной CFLAGS. Также ознакомьтесь с документацией на форуме

Если всё прошло удачно, то Cairo должен присутствовать в системе как зависимость. Примечание: Для следующей проверки необходим установленный пакет gentoolkit.

Проверите с помощью следующей команды

equery d -o -p cairo

Предупреждение: Если команда не возвращает никаких пакетов, то значит чтото прошло не правильно!

После произведенных действий Вы должны получить функционирующую установку Xgl. Пожалуйста ознакомьтесь с Глава работающий Xgl.

nVidia пользователи

nVidia драйвер версии 9625 BETA и старше содержат XGL и AIGLX-подобные расширения и могут быть использованы совместно с другими композитными менеджерами окон. см. en:HOWTO nVidia GL Desktop Effects для более детальной информации.

Обновление Xgl

Из-за быстрой разработки Xgl и Compiz, оверлей обновляется как минимум один раз в неделю. Используйте один из приведённых методов для обновления оверлея.

Обновление с помощью Layman утилиты

Обновление оверлея с помощью Layman утилиты очень просто:

layman -s xeffects

Обновление вручную

Для обновления оверлея вручную, перейдите в директорию оверлея и введите ./svnup.sh. этот скрипт также выведет список изменений.

```
cd /usr/local/overlays/xgl-coffee/  
./svnup.sh
```

Запуск Xgl

Конфигурация Compiz

Для начала надо сконфигурировать Compiz и его плагинов. Если никакие плагины не указаны, тогда окна останутся без декораций! (не будет кнопок minimize/maximize/close, невозможно будет изменять размеры окна, ничего вообще толком не будет.)

Существуют два пути указания плагинов. Вы можете указать плагины в командной строке или использовать оконный редактор конфигурации например с помощью программ gconf-editor или gset-compiz. Но вы не можете использовать оба способа одновременно! Если gconf указан в командной строке, тогда Compiz проигнорирует все другие плагины указанные в командной строке и возьмёт список плагинов из вашей конфигурации gconf.

Таким образом если Вы впервые запускаете Xgl и Вы хотите конфигурировать в оконном редакторе, наступило время сделать это, до того как Вы выйдете из Вашей X сессии.

Способ "По умолчанию": Использование gconf-editor

gconf-editor

Перейдите в apps --> compiz --> general --> allscreens --> options и отредактируйте узел active_plugins находящийся справа. Примечание: Если в gconf-editor нет вкладок Compiz, тогда это можно исправить таким образом:

```
# export GCONF_CONFIG_SOURCE="xml:merged:/etc/gconf/gconf.xml.defaults"
# gconftool-2 --makefile-install-rule /etc/gconf/schemas/compiz.schemas
```

Способ "По умолчанию" для compiz-quinnstorm: Использование csm

csm

Плагины могут быть активированы кликая в checkbox после имени. Индивидуальная конфигурация плагинов осуществляется после щелчка мышью на имени плагина. Примечание: Compiz-quinnstorm поставляется вместе с Compiz Manager (compiz-manager). Это оконное приложение находящееся в системной панели и включает в себя Compiz Settings Manager (csm) и Compiz Theme Manager (gcompizthemer). Запустив на выполнение эти две утилиты (управляйте compiz и темами), перестаруйте оконный менеджер или декоратор из системной панели и установите запасной оконный менеджер который будет использоваться в случае неудачного старта.

```
# compiz-manager
```

or (if any dbus problem) :

```
# dbus-launch compiz-manager
```

Альтернативный способ: Использование gconftool-2

```
# gconftool-2 -s /apps/compiz/general/allscreens/options/active_plugins  
"[gconf,decoration,wobbly,fade,minimize,cube,switcher,move,resize,place,rotate,zoom,s  
cale]" -t list --list-type=string
```

Укажите активные плагины внутри квадратных скобок.

Альтернативный способ: Использование gset-compiz Предупреждение:

gset-compiz больше не пытается использовать все возможности Compiz. Используйте лучше gconf-editor или более новые программы для возможностей которыми не располагает gset-compiz.

gset-compiz

Плагин можно активировать поставив галочку сразу возле его имени. Конфигурация индивидуальных настроек плагина доступна после нажатия кнопки Plugins находящейся сразу под меню.

Как минимум, потребуются пакеты gconf и decoration. Также, если Вы хотите включить все доступные эффекты, надо также добавить wobbly, fade, switcher, move, resize, place, minimize, cube, rotate, zoom, и scale. Из имени пакета Вы сами можете догадаться, что выполняет каждый из плагинов, так что не торопитесь включать их все. Это как предпочтения "сладостей для глаз", установите плагины move и resize - у Вас появится возможность работать с их функциями.

Using Quinnstorm's compiz & cgwd

Quinnstorm's new compiz doesn't use gconf plugin anymore (so settings). To use it, replace any instance of compiz --replace gconf found in this document with dbus-launch compiz --replace dbus csm. And set it by csm or dbus-launch csm.

Quinnstorm's new window decorator must be called instead of gnome-window-decorator in order to function. If you chose to emerge cgwd, just replace any instance of gnome-window-decorator found in this document with dbus-launch cgwd.

Оконный режим

Запустив Xgl таким способом мы создадим новое окно с включенным Xgl сервером. Это отличный способ протестировать правильность установки прежде чем запускать XGL в полноэкранном режиме.

Чтобы попробовать его работу в оконном режиме при уже запущенном X сервере:

Code: Для ati пользователей

```
Xgl      :1      -ac      -accel      glx:pbuffer      -accel      xv:pbuffer      -fp
/usr/share/fonts/misc,/usr/share/fonts/other_fonts
```

Code: Для nVidia пользователей

```
Xgl      :1      -ac      -accel      glx:pbuffer      -accel      xv      -fp
/usr/share/fonts/misc,/usr/share/fonts/other_fonts
```

Примечание: Обсуждение использования pbuffer или fbo как аргументов accel, можно почитать здесь <http://forums.gentoo.org/viewtopic-t-455153-highlight-.html>

Примечание: Xgl не будет считывать значение FontPath из xorg.conf, по-этому, лучше в конце строки после опции -fp прописать пути к шрифтам разделяя их запятой. Используйте этот скрипт для получения путей к шрифтам прописанным в файле xorg.conf:

```
grep -i fontpath /etc/X11/xorg.conf | egrep -v "[:space:]*#" | sed "s/.\+\"(\.\\+\\)\\"/1,/g" |
xargs echo | sed "s/ / /g" | sed "s/,/$/"
```

Протестируйте Вашу конфигурацию:

Запустите композитный оконный менеджер (compiz or compiz-quinnsstorm), запустите декоратор окон (gnome-window-decorator or cgwd) и запустите приложение (xterm). Code: Для большинства: Если Вы используете compiz совместно с gnome-window-decorator

Учтите что название gnome-window-decorator изменилось на gtk-window-decorator 22 Сентября 2006 года.

```
LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ \
```

```
DISPLAY=:1 compiz gconf
```

```
DISPLAY=:1 gnome-window-decorator
```

```
DISPLAY=:1 xterm
```

Code: Альтернатива 1: если Вы используете compiz-quinnsstorm совместно с gnome-window-decorator

```
LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ \
```

```
DISPLAY=:1 dbus-launch compiz dbus csm
```

```
DISPLAY=:1 gnome-window-decorator
```

```
DISPLAY=:1 xterm
```

Code: Альтернатива 2: Если вы используете compiz-quinnsstorm с cgwd

```
LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ \
```

```
DISPLAY=:1 dbus-launch compiz dbus csm
```

```
DISPLAY=:1 dbus-launch cgwd
```

```
DISPLAY=:1 xterm
```

Если всё прошло успешно, мы можем начать конфигурировать Вашу систему для запуска Xgl с Вашим графическим менеджером аутентификации. Следуйте инструкциям которые помогут Вам настроить менеджер аутентификации. Это последний шаг необходим для удобного использования Xgl.

startx (Для пользователей startx)

Если вы обычно загружаетесь в консоль и Xorg сервер запускается через startx, а .xinitrc загружает ваш менеджер окон на DISPLAY:0, то вам нужно: написать скрипт startxgl для запуска сервера Xgl, а так же файл .xglnitrc для загрузки менеджера окон на DISPLAY:1. Эти действия не изменят вашей текущей конфигурации и вы сможете в любой момент вернуться в нормальный Xorg сервер.

1) Скопируйте startx (`cp /usr/bin/startx /usr/local/bin/startxgl`) и отредактируйте 2 строки: Файл: `/usr/local/bin/startxgl`

```
## userclientrc=$HOME/.xinitrc
```

```
userclientrc=$HOME/.xglnitrc
```

```
## xinit $clientargs -- $serverargs -deferglyphs 16 &
```

```
# ATI
```

```
xinit $clientargs -- /usr/bin/Xgl :1 $serverargs -ac -accel xv -accel glx:pbuffer
-deferglyphs 16 &
```

```
# NVIDIA
```

```
xinit $clientargs -- /usr/bin/Xgl :1 $serverargs -ac -accel xv -accel glx:fbo -deferglyphs
16 &
```

2) создайте файл `~/xglnitrc` (`vi ~/xglnitrc`) Файл: `~/xglnitrc`

```
sleep 2 # give Xgl some extra time to start
```

```
DISPLAY=:1 KDEWM=compiz-decorator startkde
```

Примечание: Пользователи Gnome должны вписать `gnome-session` вместо `startkde` и `WINDOW_MANAGER` вместо `KDEWM`. Пользователи других DM вписывают соответствующие им команды запуска. Для уточнения проверьте ваш `~/xinitrc`.

3) создайте файл `compiz-decorator` (`vi /usr/local/bin/compiz-decorator`) Файл: `/usr/local/bin/compiz-decorator`

```
## Start compiz or compiz-quininstorm
```

```
## compiz --replace gconf &
```

```
## dbus-launch compiz --replace dbus csm &
```

```
compiz --replace gconf &
```


sleep 2

```
## Start gnome-window-decorator or cgwd
## gnome-window-decorator &
## dbus-launch cgwd &
gnome-window-decorator &
```

Примечание: Команды для запуска и использования compiz-quinnstorm и cgwd в скрипте закоментированны. Если вы хотите пользоваться ими то не забудьте закоментировать текущие

Смените права доступа на /usr/local/bin/compiz-decorator что бы сделать его исполняемым:

```
# chmod +x /usr/local/bin/compiz-decorator
```

4) Для запуска вашего оконного менеджера в XOrg используйте startx, для запуска в Xgl используйте startxgl.

Скрипты тестировались на связке ATI и KDE. --Ash 17:48, 11 Сентября 2006 (UTC)

XDM (Базовая графическая система аутентификации)

Измените ссылку на X сервер: Файл: /etc/X11/xdm/Xservers
:0 local /usr/bin/Xgl vt7

KDM (для пользователей KDM)

Замените строку с ServerCmd в kdmrc на одну из следующих:

Файл: /usr/kde/<your version>/share/config/kdm/kdmrc

NVidia Command

ServerCmd=/usr/bin/Xgl -br -ac -accel glx:pbuffer -accel xv

ATI Command

ServerCmd=/usr/bin/Xgl -br -ac -accel glx:pbuffer -accel xv:pbuffer

Вы также должны поменять значение ServerTimeout так как Xgl запускается немного дольше, чем нормальный X сервер:

Файл: /usr/kde/<your version>/share/config/kdm/kdmrc

ServerTimeout=30

Если kdm не смог загрузится, измените значение ServerTimeout на 600. Учтите, что в этом случае время загрузки Xgl значительно увеличится.

Не используйте скрипт /usr/kde/3.5/share/config/kdm/Xstartup для запуска Compiz. Для его выполнения нужны права пользователя root.

Скопируйте скрипт в другое место, например сюда: Файл: /usr/local/bin/compiz-decorator

```
## Start compiz or compiz-quinnstorm
## compiz --replace gconf &
## dbus-launch compiz --replace dbus csm &
compiz --replace gconf &
```

```
sleep 2
```

```
## Start gnome-window-decorator or cgwd
## gnome-window-decorator &
## dbus-launch cgwd --replace &
gnome-window-decorator --replace &
```

Примечание: Команды для запуска и использования compiz-quinnstorm и cgwd в скрипте закомментированы. Если вы хотите пользоваться ими то не забудьте закомментировать текущие.

Сделайте вновь созданный скрипт /usr/local/bin/compiz-decorator исполняемым:

```
# chmod +x /usr/local/bin/compiz-decorator
```

Далее измените:

Файл: /etc/env.d/99kde-env
KDEWM=compiz-decorator

И наконец, выполните команду:

```
# env-update
```

Теперь после перезагрузки или выполнения команды /etc/init.d/xdm start вы запустите полнофункциональный Xgl со всеми эффектами без пользовательского вмешательства и аутентификацией с помощью KDM.

GDM (для пользователей GNOME)

Вы должны сделать 3 вещи, чтобы заставить Xgl работать с GDM и использовать Compiz для автоматической загрузки при авторизации пользователя.

Сконфигурируйте GDM для Xgl с отложенным запуском, чтобы при запуске нельзя было сказать, что она не работает

Сконфигурируйте GDM для использования с Xgl-сервером

Установите сессию GNOME для автоматического запуска из Compiz **Примечание:** Конфигурирование gdm для отложенного запуска Xgl доступно с gdm-2.14.1 и поздних версий. Если Вы используете раннюю версию, начните с шага 2, но,

возможно, Вам не удастся заставить gdm работать с Xgl. Также использование gdmsetup для конфигурирования gdm на использовании Xgl-сервера (Раздел 2a) может быть недоступно для версий gdm ранее 2.14.0. Если недоступны необходимые Вам функции, самостоятельно отредактируйте конфигурационный файл GDM (Раздел 2b).

Конфигурирование Gdm на отложенный запуск для запуска Xgl

Gdm сконфигурирована по умолчанию на завершение (kill) X-сервера, если загрузка заняла более 10 секунд. К сожалению, новые версии Xgl могут загружаться дольше. Пользователи Xgl в Gentoo, могут сконфигурировать Gdm (пример по версии 2.14.1) на ожидание более 10 секунд.

```
# emerge --ask --verbose >=gdm-2.14.1
```

Добавьте/измените следующие команды секции [daemon] файла /etc/X11/gdm/custom.conf. Строка: Файл: /etc/X11/gdm/custom.conf

```
GdmXserverTimeout=30
```

Если /etc/X11/gdm/custom.conf не существует, попробуйте /etc/X11/gdm/gdm.conf-custom или /etc/X11/gdm/gdm.conf.

Конфигурирование Gdm для использования сервера Xgl

Использование gdmsetup

Для начала сделайте резервную копию файла gdm.conf

```
cp /etc/X11/gdm/gdm.conf /etc/X11/gdm/gdm.conf.bak
```

Запустите gdmsetup

Выберите раздел Security

Щёлкните кнопку Configure X Server, расположенную снизу-справа раздела

Для начала мы удалим значения для стандартного X-сервера

Нажмите по стандартному серверу, который находится в списке Servers to Start

Нажмите Remove Button

Сейчас мы создадим значения для сервера Xgl

Нажмите кнопку Add/Modify

Задайте использование нового сервера VT 1. Используйте Standard для Server опцию — она отображает, какой профиль использовать для сервера, в идеале мы должны создать новый "Xgl"-профиль, а в настоящее время при помощи gdmsetup это сделать невозможно, так что отредактируем настройки для профиля Standard. Не вводите здесь каких-либо опций. Нажмите Ok по завершении.

В Server Settings измените Server Name на "Xgl"

Введите команду Xgl в поле Command, например, для пользователей ATI:

```
/usr/bin/Xgl :1 -ac -accel glx:pbuffer -accel xv:pbuffer
```

NVIDIA users should use:

```
/usr/bin/Xgl -br -ac -accel glx:pbuffer -accel xv
```

Измените в выпадающем списке Launch значение на Greeter

Установите Logins are handled by this computer

Установите Flexible (on demand)

Save и Close

Перезапустите gdm

Ручная настройка

Для начала сделайте резервную копию файла gdm.conf

```
cp /etc/X11/gdm/gdm.conf /etc/X11/gdm/gdm.conf.bak
```

Now open /etc/X11/gdm/gdm.conf in a text editor and search for the [servers] section. This should be near the end of the file.

Comment out 0=Standard and add a new line that reads 0=inactive and one that reads 1=Xgl.

Add the server information for the Xgl server, remember to substitute the correct Xgl server command! The one below is for ATI cards.

The modified /etc/X11/gdm/gdm.conf:

```
[servers]
```

```
#0=Standard
```

```
1=Xgl
```

```
# Definition of the xgl X server.
```

```
[server-Xgl]
```

```
name=Xgl
```

```
command=/usr/bin/Xgl :1 -ac -accel xv:pbuffer -accel glx:pbuffer
```

```
flexible=true
```

```
chooser=false
```

```
handled=true
```

```
priority=0
```

Restart gdm Примечание: For i810 users (not necessarily i915; this would kill my xorg sessions, I had to use the ATI command), the command for the above should be:

```
[server-xgl]
```

```
name=Xgl server
```

```
command=/usr/bin/Xgl -accel xv -accel glx:pbuffer -accel xv:pbuffer -ac -audit 0 -br -dpms -dpi 72
```

```
flexible=true
```

Via xsession

You can add another session for Xgl on display :1 (leaving the standard server on :0). It is a 2 step setup and imho the best option, first create a file to start Xgl and second create a new xsession file.

Create a new file startxgl.sh in /usr/bin which starts Xgl.

```
vim /usr/bin/startxgl.sh
```

And add one of the four options below.

NVIDIA (using GNOME)

```
#!/bin/bash
```

```
Xgl -fullscreen :1 -audit 0 -ac -br -accel glx:pbuffer -accel xv:fbo &
sleep 2 && DISPLAY=:1 gnome-session
```

NVIDIA (using KDE)

```
#!/bin/bash
```

```
Xgl -fullscreen :1 -audit 0 -ac -br -accel glx:pbuffer -accel xv:fbo &
sleep 2 && DISPLAY=:1 startkde
```

ATI and Intel (using GNOME)

```
#!/bin/bash
```

```
Xgl -fullscreen :1 -audit 0 -ac -br -accel glx:pbuffer -accel xv:pbuffer &
sleep 2 && DISPLAY=:1 gnome-session
```

ATI and Intel (using KDE)

```
#!/bin/bash
```

```
Xgl -fullscreen :1 -audit 0 -ac -br -accel glx:pbuffer -accel xv:pbuffer &
sleep 2 && DISPLAY=:1 startkde
```

Create a new xsession file: Файл: /usr/share/xsessions/xgl.desktop

[Desktop Entry]

Encoding=UTF-8

Name=Xgl

Comment=Start an Xgl Session

Exec=/usr/bin/startxgl.sh

Icon=

Type=Application

Now you will be able to select a new session via gdm's option menu. Options -> Select Session

The benefit is that it will not replace your normal XOrg server. So when you "break" your Xgl you can return to the normal XOrg server.

Adding Compiz to your GNOME session (replacing Metacity)

Here is a little startscript, which I've placed in /usr/bin/. Useful if you do not always run Xgl. Change or remove the setxkbmap line according to your keyboard layout. Файл: /usr/bin/compizrc

```
#!/bin/bash
#
# Start compiz within gnome-session
#
if [ `ps -A -o comm | grep -c '^Xgl$'` == "1" ]; then
    DISPLAY=:1 LD_LIBRARY_PATH=/usr/lib/openssl/xorg-x11/lib/ compiz --replace
gconf &
    DISPLAY=:1 gnome-window-decorator &
#    DISPLAY=:1 setxkbmap -model pc105 -layout it -variant basic # Change
according to your needs, and uncomment if needed

#    Or like that, if you need to switch between layouts.
#    setxkbmap -model pc105 -layout "us,ru(winkeys)" -variant winkeys -option
"grp:alt_shift_toggle,grp_led:scroll"
else echo "${0}: Error: Compiz and g-w-d not launched. Xgl not running?"
fi
```

Файл: /usr/bin/compizrc.modified

```
#!/bin/bash
#
# Start compiz within gnome-session
#
if [ `ps -A -o comm | grep -c '^Xgl$'` == "1" ]; then
# For old versions
#    DISPLAY=:0 LD_LIBRARY_PATH=/usr/lib/openssl/xorg-x11/lib/ compiz --replace
gconf &
# For Compiz-Quinnstorm with csm
    DISPLAY=:0 LD_LIBRARY_PATH=/usr/lib/openssl/xorg-x11/lib/ compiz --replace
dbus csm &
    DISPLAY=:0 dbus-launch cgwd --replace &
#    DISPLAY=:0 setxkbmap -model br-abnt2 -layout br -variant br
else echo "${0}: Error: Compiz and cgwd not launched. Xgl not running?"
fi
```

Code: Make the script executable

```
chmod 755 /usr/bin/compizrc
```

After logging into GNOME, start System->Settings->Sessions. Here you go to the 3rd tab Startup Programs and press the Add button. Enter the path to your Compiz startscript (e.g., /usr/bin/compizrc).

After logging out and back in Compiz will start, but you'll see no decorations and effects. That's because the Compiz gconf tree for the current user has just been created - without any plugins!

So start your configuration editor and go to apps/compiz/general/allscreens/options and edit the active_plugins key. This should contain the following items in this order: gconf decoration wobbly fade minimize cube rotate zoom scale move resize place menu switcher

If the key is not there, make it with a type of List

After restarting your Xgl Server should be accessible with GDM & gnome-session.

Xsession (Modifying startx)Примечание: This is for NVIDIA users.

Предупреждение: This currently will not work for ATI users because of the display issues, requiring to be run on DISPLAY=:1.

Примечание: I use similar configuration (see startx section) on ATI. Consider /usr/bin/Xgl :1 instead of /usr/bin/Xgl and starting Gnome at :1 as well, if DISPLAY:0 is problematic. --Ash 17:57, 11 September 2006 (UTC)

Copy /usr/bin/startx to /usr/bin/startxgl and change the following line towards the end of the file: Файл: /usr/bin/startxgl

```
xinit $clientargs -- $serverargs -deferglyphs 16 &
```

to

```
xinit $clientargs -- /usr/bin/Xgl $serverargs -ac -accel xv -accel glx:pbuffer -deferglyphs 16 &
```

This does not start Compiz so create a small start script and add it to the gnome-session manager (Once GNOME is up. The first time, you will have to open a terminal and call the script manually.) Файл: /usr/bin/compizrc

```
#!/bin/bash
```

```
#
```

```
# Start compiz within gnome-session
```

```
#
```

```
if [ `ps -A | grep Xgl | wc -l` == "1" ]; then
```

```
    LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ compiz --replace gconf &
    gnome-window-decorator &
```

```
#    setxkbmap -model pc105 -layout it -variant basic # Change according to your
needs, and uncomment if needed
```

```
# Or like that, if you need to switch between layouts.  
# setxkbmap -model pc105 -layout "us,ru(winkeys)" -variant winkeys -option  
"grp:alt_shift_toggle,grp_led:scroll"
```

```
fi
```

Примечание: This should start the desktop environment configured via XSESSION in /etc/rc.conf

Xfce4

With xfce4-session

xfce4-session is installed by default as part of the xfce4 meta-package. If you installed xfce4 this way, the following will work.

First, it is necessary to create a script that will start Compiz and gnome-window-decorator with the correct libraries and switches. Файл: /usr/bin/compizrc

```
LD_LIBRARY_PATH=/usr/lib/openssl/xorg-x11/lib compiz --replace gconf &  
gnome-window-decorator
```

Code: Make the script executable

```
chmod 755 /usr/bin/compizrc
```

Now, edit the xfce4-session settings to point to this rather than xfwm4. Файл: /etc/xdg/xfce4-session/xfce4-session.rc

[Failsafe Session]

Count=4

Client0_Command=compizrc

Client0_PerScreen=False

Client1_Command=xfce4-panel

Client1_PerScreen=True

Client2_Command=xftaskbar4

Client2_PerScreen=True

Client3_Command=xfdesktop

Client3_PerScreen=False

Примечание: Instead of editing system-wide file you may also add above section to ~/.config/xfce4-session/xfce4-session.rc file. Note also that you may need to delete content of the ~/.cache/sessions directory, if you still have xfwm4 rather than Compiz running.

Finally, create a script to start Xgl with Xfce4. Файл: /usr/bin/startxgl for NVIDIA

```
#!/bin/bash
```

```
Xgl -ac -accel xv:fbo -accel glx:pbuffer &  
xfce4-session
```

Файл: /usr/bin/startxgl for ATi

```
#!/bin/bash
```

```
Xgl -ac -accel xv:pbuffer -accel glx:pbuffer &  
xfce4-session
```

Code: Make the script executable

```
chmod 755 /usr/bin/startxgl
```

You will notice that xfce4-panel is now transparent, just like if you were using xfwm4's compositor. The pager will be distorted, but otherwise everything works beautifully. Windows even minimize to the correct spot on the taskbar.

Without xfce4-session

If you compiled a minimal Xfce (without xfce4-session, xffm4, etc.), then you can use this altered startxgl script from the gnome script above:

```
#!/bin/bash
```

```
echo ">Starting XGL at Display: $1"
```

```
echo "===== XGL ====="
```

```
Xgl :$1 -ac -accel xv -accel glx:pbuffer &  
sleep 3
```

```
echo "===== COMPIZ ====="
```

```
DISPLAY=: $1 LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ compiz --replace  
switcher decoration wobbly fade minimize cube rotate zoom scale move resize place &  
sleep 3
```

```
echo "===== XFCE ====="
```

```
DISPLAY=: $1 xfce-mcs-manager
```

```
DISPLAY=: $1 gnome-window-decorator &
```

```
DISPLAY=: $1 xftaskbar4 &
```

```
DISPLAY=: $1 xfdesktop &
```

```
DISPLAY=: $1 exec xfce4-panel
```

Again, xfce4-panel will be transparent as if you had the X.org composite extension turned on (but you don't) or if you use xfce4-svn, and the pager will be distorted. Everything else should be fine.

With startxfce4

To do it the old fashioned, or 'normal' way via startx/startxfce4 I did the following. First, startxfce4 shouldn't be called I don't think. just use an .xinitrc in your homedir and startx.
Файл: ~/.xinitrc

```
#!/bin/sh
```

```
/usr/bin/startxfce4
```

Now as root, we create compizrc, like above. Note however that for me, it worked best/only without the DISPLAY variable. Файл: /usr/bin/compizrc

```
#!/bin/bash
```

```
LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib/ compiz --replace gconf >  
~/compiz.log 2>&1 &
```

```
sleep 1;
```

```
gnome-window-decorator > ~/gnome-window.log 2>&1 &
```

Obviously the output redirections are optional (but handy for debugging). The sleep 1; might also be removed to speed things up. If things go TO fast however, you might get in trouble though. Don't forget to chmod 755 /usr/bin/compizrc

Lastly we need to use this new compizrc instead of the default xfce4 window manager.

Again, as root edit /etc/xdg/xfce4-session/xfce4-session.rc and replace xfwm4 with compizrc Файл: /etc/xdg/xfce4-session/xfce4-session.rc (old)

```
Client0_Command=xfwm4
```

Файл: /etc/xdg/xfce4-session/xfce4-session.rc (new)

```
Client0_Command=compizrc
```

Now only one thing is left, start Xgl. As above you can copy startx in /usr/bin/startx to /usr/bin/startxgl and make the following change: Файл: /usr/bin/startxgl (old)

```
xinit $clientargs -- $serverargs -deferglyphs 16 &
```

Файл: /usr/bin/startxgl (new)

```
xinit $clientargs -- /usr/bin/Xgl -ac -accel xv -accel glx:pbuffer $serverargs -deferglyphs  
16 &
```

Примечание: Todo: Make Client0_Command depend on wether startx or startxgl is beeing used as 3D apps don't appear to work as they should and switching back to the 'old way' is required.

Edit: it seems that when using the startxfce4 script, the /etc/xdg/xfce4-session/xfce4-session.rc file (and the .xinitrc) get ignored thus starting a non Xgl desktop. Making 'switchin' not needed. startxfce4 for oldschool, startxgl for new style

Entrance

Emerge latest entrance:

```
echo "=x11-misc/entrance-0.9.0.007 ~x86" >> /etc/portage/package.keywords
```

```
emerge -av entrance
```

Use the ecore_config utility to change the xserver string (choose according to your video card):

```
# ATI
```

```
ecore_config -k "/entranced/xserver" -s "/usr/bin/Xgl" -ac -accel xv:pbuffer -accel glx:pbuffer -nolisten tcp" -c /etc/entrance_config.cfg
```

```
# NVIDIA
```

```
ecore_config -k "/entranced/xserver" -s "/usr/bin/Xgl" -ac -accel xv:fbo -accel glx:pbuffer -nolisten tcp" -c /etc/entrance_config.cfg
```

Restart entrance.

Qingy

It is quite easy to change Qingy to start Xgl instead of a regular X server. All you need to do to make it start Xgl is change 2 lines in /etc/qingy/settings Файл: /etc/qingy/settings

```
...
```

```
# x_server = "/usr/X11R6/bin/XFree86"
```

```
x_server = "/usr/bin/Xgl"
```

```
# x_args = "-nolisten tcp"
```

```
x_args = "-ac -accel xv -accel glx:pbuffer -nolisten tcp"
```

```
...
```

Now Qingy should start Xgl instead of regular X. To make KDE and GNOME load Compiz you need to export a variable. KDEWM for KDE and WINDOW_MANAGER for GNOME. It is easiest to do this in /etc/X11/Sessions.

You need to have followed the instructions for adding plugins to gconf-editor. Then make a /usr/bin/compizrc as shown and chmod +x it. Файл: /usr/bin/compizrc

```
#!/bin/bash
```

```
LD_LIBRARY_PATH=/usr/lib/opengl/xorg-x11/lib compiz --replace gconf &
```

```
sleep 2
```

```
gnome-window-decorator &
```

```
setxkbmap -model itouch -layout gb & ## Change/Remove as appropriate
```

Then edit the appropriate file (or both) in /etc/X11/Sessions

For KDE: Файл: /etc/X11/Sessions/kde-3.5

```
#!/bin/sh
```

```
export KDEWM="/usr/bin/compizrc"
```

```
exec /usr/kde/3.5/bin/startkde
```

Or for GNOME add this to the top of the file : Файл: /etc/X11/Sessions/Gnome

```
#!/bin/sh
```

```
export WINDOW_MANAGER="/usr/bin/compizrc"
```

...

For Xfce4 edit /etc/xdg/xfce4-session/xfce4-session.rc as shown in a previous section. Hopefully Qingy should now start Xgl and Compiz instead of X and some other WM.

Generic startxgl script

There is a generic startup script in the overlay. For usage instructions start it without arguments.

Currently it support Xfce, Xfce-svn, GNOME and KDE.

```
cd /usr/local/overlays/xgl-coffee
```

```
./startxgl
```

Автоматическое переключение раскладок

Для начала сделаем ebuild в portage overlay

Создаём каталог:

```
mkdir -p /usr/local/portage/x11-misc/xneur cd /usr/local/portage/x11-misc/xneur
```

Создаём ebuild:

```
nano -w xneur-0.1.0_1.ebuild Файл: /usr/local/portage/x11-isc/xneur/xneur-0.1.0_1.ebuild
```

```
# Copyright 1999-2005 Gentoo Foundation
```

```
# Distributed under the terms of the GNU General Public License v2
```

```
DESCRIPTION="It's program like Punto Switcher, but has other final aim."
```

```
SRC_URI="http://www.xneur.ru/xneur/${P}.tar.gz"
```

```
HOMEPAGE="http://www.xneur.ru/"
```

```
DEPEND="virtual/x11"
```

```
LICENSE="GPL-2"
```

```
SLOT="0"
```

```
KEYWORDS="~x86 ~amd64 ~sparc ~alpha ~ppc ~hppa ~ppc64"
```

```
src_install () {
    make DESTDIR=${D} install || die
    #dodoc AUTHORS README COPYING INSTALL NEWS ChangeLog TODO .xneurrc
}

pkg_postinst() {
    # einfo "Run:"
    # einfo "gzcat /usr/doc/${P}/.xneurrc.gz > ~/.xneurrc"
}
```

Далее: ebuild xneur-0.1.0_1.ebuild digest Поскольку у нас бэта: echo "x11-misc/xneur ~x86" >> /etc/portage/package.keywords emerge xneur ;)

А за подробностями на <http://www.xneur.ru/>

Cedega 5.1

Введение

Cedega - коммерческая версия ПО для реализации WinAPI на *nix-платформах WINE, разработанная Transgaming, ранее известная как Winex. Поддерживается большинство популярных игр. В последней версии добавилась официальная поддержка Sid Meier's Civilization® IV, FIFA® 06 и Need for Speed™: Most Wanted.

Cedega 5.1 состоит из двух компонентов:

Point-to-play - графический интерфейс для простой установки игр, распространяется в виде .rpm/.deb/.tgz

Engine / engine update - модифицированный WINE, распространяется в виде файлов .cpkg - tar архивы, содержащие tgz архив и файл manifest с номером версии .

Установка

Установка самого эмулятора, без неудобного в использовании Point-to-play

ВЫГЛЯДИТ ТАК:

Code: Установка Cedega

su

tar -zxf cedega-engine-5.1-local-update.i386.cpkg

tar -C /usr -zxf cedega_5.1-1.i386.p2p.tgz bin winex

mv /usr/share/share/doc/* /usr/share/doc

mv /usr/share/share/man/man1/* /usr/share/man1

mv /usr/bin/winex3 /usr/bin/cedega

**mkdir /usr/lib/transgaming/cedega; mv /usr/winex /usr/lib/transgaming_cedega
exit**

Первоначальная настройка

Code: Первоначальная настройка

tar -C \$HOME -zxf cedega_5.1-1.i386.p2p.tgz .transgaming

nano ~/.transgaming/config

Проверка

cedega -version

Секреты командной строки

Общие команды

поиск команды Linux, ее описания и номера секции man страниц

apropos word

закодировать файл file с помощью GnuPG

gpg -c file

раскодировать файл file

gpg file.gpg

быстрый поиск по словарю слов, начинающихся с word

look word

подсветить слово word в файле /somefile

grep --color word /somefile

запустить command с низким приоритетом

nice command

назначить низший приоритет текущему шеллу (и всем потомкам). Может быть полезно, если вашу систему сильно замедляет установка нового пакета (emerge). Кстати, для автоматического понижения приоритета emerge используется переменная PORTAGE_NICENESS в файле /etc/make.conf

renice 19 -p \$\$

посмотреть код завершения предыдущей команды

echo \$?

скачать список новых страничек с нашего сайта в 01:00 в текущую директорию

echo "wget http://ru.gentoo-wiki.com/Special:Newpages" | at 01:00

в 17:45 послать пустое письмо с заголовком 'got the r00t?' на bugs@microsoft.com

echo "mail -s 'got the r00t?' bugs@microsoft.com < /dev/null" | at 17:45

напечатать 1234 в соответствии с настройками локали (в России обычно 1.234)

printf "%'d\n" 1234

запускать просмотр прерываний каждую секунду

watch -n1 "cat /proc/interrupts"

посмотреть, сколько времени занимает выполнение команды

time command

удобный алиас для вывода дампа

alias hd='od -Ax -tx1z -v'

полный путь к команде command

which command

вывести в 9 колонок по ширине терминала

ls | pr -T9 -W\$COLUMNS

установить время изменения для файла file (в формате YYMMDDhhmm)

touch -c -t 0304050607 file

показать иерархию запущенных процессов

pstree -p

показать процессы, использующие файл /dir/file (чаще всего использую нечто вроде 'lsof /mnt/cdrom'). Не забудьте сделать 'emerge -n lsof'

lsof /dir/file

Ввод-вывод

объединить stderr и stdout (вернее сказать перенаправить stderr в stdout)

gcc file.c 2>&1 | less

Перенаправить stderr в файл errors.log а stdout в файл compile.log для дальнейшего анализа

gcc file.c 2>errors.log 1>compile.log

Если нужно запретить вывод потока (например убрать в скрипте сообщения об ошибках) то достаточно перенаправить поток на устройство /dev/null, тогда сообщения уйдут в никуда:

rm -r /var/tmp/portage 2>&1 >/dev/null или rm -r /var/tmp/portage &> /dev/null

Порой нужно объединить выходной поток сразу нескольких команд, а поток одной изних отключить, тогда группу нужно объединить в скобки:

```
( cat /etc/gentoo-release ; cat /etc/passwd >/dev/null; cat /etc/group ) |less
```

Навигация по директориям.

вернуться в предыдущую директорию (не путать с 'cd ..')

cd -

вернуться в домашнюю директорию

cd

перейти в директорию dir, запустить command и автоматически вернуться назад

(cd dir; command)

добавить текущую директорию в стек, чтобы потом можно было сделать popd и вернуться к ней

pushd .

Дисковое пространство

показать список файлов с информацией о каждом файле (-l), отсортировав список по убыванию размера (-S) и перевернуть список (-r). Получим сортировку по возрастанию.

ls -lSr

показать, сколько места на диске занимает файл file и директория dir

du -sh file dir

показать свободное место на примонтированных ресурсах

df -h

то же самое но в инодах

df -i

показать геометрию размещения разделов жесткого диска (нужны права root)

fdisk -l***Работа с CD***

создать iso-образ диска и заархивировать его

dd bs=1M if=/dev/cdrom | gzip > cdrom.iso.gz

создать iso-образ из директории dir

mkisofs -r dir | gzip > cdrom.iso.gz

смонтировать cdrom.iso в /mnt/dir (для просмотра и правки)

mount -o loop cdrom.iso /mnt/dir

записать архивированный образ на диск

gzip -dc cdrom.iso.gz | cddrecord dev=0,0,0 -

рипнуть дорожки с Audio-CD в текущую директорию (в формате .wav)

cdparanoia -B

создать Audio-CD из всех .wav файлов в текущей директории

cddrecord dev=0,0,0 -audio *.wav

конвертировать файл track.cdda.wav формат .ogg

```
oggenc --tracknum="track" track.cdda.wav -o "track.ogg"
```

Работа с архивами

создать архив директории dir

```
tar c dir/ | bzip2 > dir.tar.bz2
```

```
tar -cjf dir.tar.bz2 dir
```

извлечь архив в директорию /to/dir (без '-C /to/dir' в текущую директорию)

```
bzip2 -dc dir.tar.bz2 | tar x -C /to/dir
```

```
tar -xjf dir.tar.bz2 -C /to/dir
```

создать архив всех .png файлов в директории dir/

```
find dir/ -name "*.png" | xargs tar rf dir.tar; bzip2 dir.tar
```

скопировать (с сохранением прав доступа!) директорию /dir/to/copy/ в /where/to/

```
( tar cf - /dir/to/copy ) | ( cd /where/to/ && tar xf - )
```

скопировать (с сохранением прав доступа!) содержимое директории /dir/to/copy в /where/to/

```
( cd /dir/to/copy && tar cf - . ) | ( cd /where/to/ && tar xf - )
```

скопировать (с сохранением прав доступа!) директорию /dir/to/copy/ в директорию /where/to/ на удаленной машине

```
( tar cf - /dir/to/copy ) | gzip | ssh user@remote 'cd /where/to/ && gzip -dc | tar xf -'
```

создать и сохранить бэкап жесткого диска на удаленной машине

```
dd bs=1M if=/dev/hda | gzip | ssh user@remote 'dd of=hda.gz'
```

Работа с файлами

удобный листинг по команде l

```
alias l='ls -l --color=auto'
```

вывести листинг с упорядочиванием по дате

```
ls -lrt
```

показать в папке dir файлы модифицированные раньше, чем 2 дня назад

find dir -mtime +2

удаляем в папке dir файлы старше 1 часа

find dir -type f -mmin +60 -exec rm -f {} \;

удаляем в папке dir файлы старше 10 дней

find dir -type f -mtime +10 -exec rm -f {} \;

найти в текущей директории (и ниже) .c и .h файлы содержащие строку "search string"

find -name "*.ch" | xargs grep -E "search string"

искать строку "search string" только в обычных файлах

find -type f | xargs grep -E "search string"

искать строку "search string" только в текущей директории (не спускаться ниже)

find -type f -maxdepth 1 | xargs grep -E "search string"

в текущей директории найти все файлы с расширением sql содержащие USER1 и заменить в них USER1 на USER2

tmp="/tmp/\$RANDOM\$\$tmp"; f="USER1"; r="USER2";

find . -name '*.sql' -exec grep -l "\$f" {} \; |

xargs --replace="{}" bash -c "(sed 's/\$f/\$r/g' < {} > \$tmp && cat \$tmp > {} && rm -f \$tmp)"

найти файл в базе данных программы slocate. Замечание: данное регулярное выражение эквивалентно маске *file*.txt

locate -r 'file[^/]*\.'txt'

Работа с файловой системой

отформатировать флоппи-диск с FAT

mkdosfs -c -f 16 -n "название тома" /dev/fd0 или mkfs -t fat16 /dev/fd0

"правильная" кодировка и права файлов для сменных носителей (floppy, CD, flash)

необходимо прописать в /etc/fstab!!!

для CDRom

/dev/cdrom /mnt/cdrom iso9660 ro,nosuid,noauto,exec,user,nodev 0 0

для "дискетки"

```
/dev/fd0 /mnt/floppy vfat iocharset=koi8-r, sync, nosuid, codepage=866, user, --, noauto, nodev, unhide 0 0
```

для раздела Windows

```
/dev/hda1 /mnt/win vfat user, exec, umask=0, codepage=866, iocharset=koi8-r 0 0
```

для "флешки"

```
/dev/sda1 /mnt/flash vfat user, exec, umask=0, sync, codepage=866, iocharset=koi8-r 0 0
```

вышенаписанное справедливо для локали koi8-r, ваша может отличаться (см. locale), в этом случае все "koi8-r" необходимо исправить на свои, и помните, что это, лишь, пример

Работа с календарем

вывести на экран календарь на текущий, предыдущий и следующий месяцы

cal -3

на какой день недели выпал в этом году день рождения Linux?

date --date='25 Aug' +%A

конвертировать в дату (в соответствии с локалью) - 130204800 секунд, прошедшие с начала эпохи Unix

date --date '1970-01-01 UTC 130204800 seconds'

Сколько сейчас времени на западном побережье США (используйте tzselect чтобы узнать параметр для TZ)

TZ="America/Los_Angeles" date

Работа с сетью

(Предварительно сделать emerge на net-tools, sys-apps/iproute2, net-dns/bind-tools)
показать сетевые интерфейсы

ip link show

показать статус сетевых интерфейсов

ethtool interface или /sbin/ifconfig

переименовать eth0 в wan

ip link set dev eth0 name wan

добавить ip 1.2.3.4 с маской 255.255.255.0 на eth0

ip addr add 1.2.3.4/24 brd + dev eth0

поднять интерфейс

ip link set dev interface up

опустить интерфейс

ip link set dev interface down

сделать шлюзом по умолчанию 1.2.3.254

ip route add default via 1.2.3.254

показать ip адрес для name

host name

показать прослушиваемые порты в системе (и кто их слушает)

netstat -lp --inet

показать активные соединения

netstat -p --inet

Математика

простое вычисление

echo "(321-123)/123" | bc -l

простое целочисленное вычисление с использование bash

echo "\$(((51+123)/2))"

использование python для научных вычислений

echo "print (10E3-123)/123" | python

приведение систем счисления (в данном случае, десятичной к шестнадцатеричной)

echo "obase=16;ibase=10;123" | bc

Более сложное вычисление - максимальная скорость передачи (в пакетах в секунду) в Fast Ethernet сети (100Mb)

```
echo "framing=20; minsize=64; (100*10^6)/((framing+minsize)*8)" | bc
```

А здесь мы строим график зависимости скорости передачи от размера пакета всё в той же 100-мегабитной сети.

```
echo "framing=20; plot [64:1518] (100*10**6)/((framing+x)*8)" | gnuplot -persist
```

Тонкость работы ccache с emerge

Вы пробовали оценить эффективность работы ccache при пересборке системы? Прodelайте опыт: очистите кэш (ccache -C) и запустите пересборку (emerge --emptytree world). После пересборки десятка-другого пакетов прервите процесс и посмотрите статистику (ccache -s) Запустите пересборку по новой и опять прервитесь на старом месте. Что показывает статистика? Много ли совпадений?

А теперь добавьте в /etc/make.conf строчки

```
CC=gcc
```

```
CXX=g++
```

очистите кэш и статистику и повторите эксперимент заново. Не правда ли, что после этого кэш только и заработал?

Суть: по состоянию на апрель 2005 ccache опознает как компилятор только стандартные имена типа gcc, g++ и не опознает такие имена как i686-pc-linux-gnu-gcc. Однако именно такие имена и используют пакеты в качестве компилятора после работы скриптов конфигурации configure

Keymap

Варианты раскладок клавиатуры для прописывания в /etc/conf.d/keymaps (в старых версиях baselayout переменная находится в файле /etc/rc.conf) Файл: /etc/conf.d/keymaps

```
...
```

```
KEYMAP="-u ru4"
```

```
...
```

Доступные варианты:

ru -- кодировка KOI-8, переключение [Right-Ctrl], одиночный символ [Right-Alt]

ru1 -- кодировка KOI-8, переключение [Right-Alt]

ru2 -- альтернативная кодировка (IBM866), переключение [Right-Alt]

ru3 -- кодировка KOI-8, 102 кнопочная клавиатура, переключение [CapsLock],

[Shift+CapsLock] -- фиксация регистра
ru4 -- кодировка KOI-8, 105 кнопочная клавиатура, переключение [CapsLock], [Shift+CapsLock] -- фиксация регистра
ru-cp1251 -- кодировка CP1251, переключение [Right-Ctrl], одиночный символ [Right-Alt]
ru-ms -- раскладка как в Microsoft Cyrillic keyboard layout, переключение [Right-Ctrl]
ru_win -- кодировка CP1251, переключение [Right-Ctrl], одиночный символ [Right-Alt]
ru-yawerty -- кодировка KOI-8, раскладка "ЯВЕРТЫ" (это бывает удобно при работе на клавиатуре без русских символов), переключение [Right-Alt]

Вариант, которого нет в стандартной поставке, но весьма удобен ru-utf. Переключение [Right-Ctrl], одиночный символ [Right-Alt]. Брать тут: <http://mlclm.narod.ru/ru-utf.map.gz>, сохранить в каталог /usr/share/kbd/keymaps/i386/qwerty или в случае отсутствия этого, в каталог /usr/share/keymaps/i386/qwerty.

Кроме того есть вариант ru-mab. Он даёт переключение по [Ctrl+Shift]. Взять можно тут: <http://moose.ylsoftware.com/gentoo.ru/ru-mab.map.gz> Code: /etc/conf.d/keymaps

...

KEYMAP="ru-utf"

Часто задаваемые вопросы

2. Введение

Как произносится слово Gentoo, и что оно означает?

Gentoo (Хинду) это вид маленького и быстрого пингвина, произносится по-английски как "gen-too" (джин-туу). Научное название пингвина - *Pygoscelis papua*. Имя Gentoo было дано пингвину жителями Фолклендских (Мальвинских) Островов (Islas Malvinas). По-русски прижилось произношение Gentoo как "генту".

Что делает Gentoo особенным?

Gentoo использует похожую на порты BSD систему, называемую Portage (англ.). Portage - это система управления пакетами, дающая огромную гибкость при установке и сопровождении программного обеспечения в системе Gentoo. Она предоставляет поддержку директив времени компиляции (с помощью USE-флагов), условные зависимости, сведения о пакете перед его установкой, безопасную установку (через "песочницу") и удаление ПО, системные профили, защиту конфигурационных файлов, а также многие другие возможности.

С Gentoo вы можете построить всю свою систему из исходных кодов, используя необходимую вам оптимизацию. Вы полностью контролируете, какие пакеты устанавливать, а какие нет. Gentoo предоставляет множество вариантов установки, удовлетворяющие вашим запросам, поэтому его называют метадистрибутивом.

Gentoo активно развивается. Поддерживается высокий темп разработки: заплатки быстро включаются в основное дерево проекта, документация обновляется ежедневно, часто добавляются новые функции в Portage, а официальные выпуски выходят дважды в год.

3. Установка

Все работает очень нестабильно, мои флаги оптимизации "-O9 -ffast-math -fomit-frame-pointer". В чем проблема?

Не пробуйте использовать что-либо выше, чем -O3 т.к. это не поддерживается текущими версиями gcc. Очень агрессивная оптимизация иногда заставляет компилятор изменять ассемблерный код так, что он уже выполняет несколько другие действия, чем предполагалось.

Попробуйте компилировать с флагами CFLAGS -O2 -march=<your_arch> прежде чем сообщать об ошибках.

Как изменить пароль root (или любого другого пользователя)?

Используйте passwd, чтобы сменить пароль пользователя, под именем которого вы вошли. Как root, вы можете сменить пароль любого пользователя в системе командой passwd username. Дополнительные параметры смотрите на странице: [man passwd](#).

Как добавить обычного пользователя?

Команда adduser username добавит пользователя с именем "username". Однако, такой метод не даст пользователю некоторые права которые вы, возможно, хотите ему предоставить, поэтому предпочтительней использовать:

Листинг 3.1: Использование useradd
`# useradd -m -G users,audio,wheel username`

В итоге в системе появится пользователь с именем "username". Параметр audio добавит его в группу audio и позволит пользователю работать со звуковыми платами. Параметр wheel добавит пользователя в группу wheel, которая позволит ему запускать команду su, чтобы получить привилегии суперпользователя.

Почему пользователь не может получить права root командой su?

В целях безопасности, пользователи могут выполнять команду su, только если они состоят в группе wheel. Чтобы добавить пользователя в группу wheel, выполните следующую команду из-под root:

Листинг 3.2: Добавление пользователя в группу wheel

```
# gpasswd -a username wheel
```

Как отключить devfs?

Gentoo может работать с devfs, udev, а так же поддерживает статический /dev. С появлением ядра 2.6, которое стабильно на большинстве архитектур, рекомендуется использовать udev. Информацию по настройке udev можно найти в Руководстве по udev.

Если вы хотите использовать статический /dev, установите параметр **RC_DEVICES="static"** в **/etc/conf.d/rc**.

Могу ли я обновить Gentoo без переустановки?

На самом деле, нет разницы между разными версиями после их установки. Gentoo 1.4 и старше основаны на glibc-2.3.x. Поэтому, выполнив `emerge --sync && emerge -uDN world`, вы приведете свою систему в самое актуальное состояние (latest Gentoo). Различия между отдельными версиями Gentoo заключаются в установочных носителях и прекомпилированных пакетах. Подробнее о профилях и о их роли при обновлении смотрите в руководстве по обновлению Gentoo.

Ядро не загружается (правильно), что мне теперь делать?

Вы не обязаны повторять каждый шаг установки, но понимание ядра и связанных с ним этапов установки необходимо. Предположим, вы установили Gentoo на /dev/hda1 (точка монтирования /boot) и /dev/hda3 (точка монтирования /) с разделом подкачки /dev/hda2:

Листинг 3.3: Конфигурация ядра

Загрузитесь с установочного CD и дождитесь приглашения командной строки

Сначала примонтируем все разделы:

```
# mount /dev/hda3 /mnt/gentoo
# mount /dev/hda1 /mnt/gentoo/boot
# swapon /dev/hda2
# mount -t proc none /mnt/gentoo/proc
```

Затем перейдем в вашу систему командой `chroot` и настроим ядро:

```
# chroot /mnt/gentoo /bin/bash
# env-update && source /etc/profile
# cd /usr/src/linux
# make menuconfig
```

Отметьте/уберите все что по вашему вы отметили/убрали неправильно

в прошлый раз

Затем выходите и откомпилируйте ядро:

```
# make && make modules_install
```

Теперь замените bzImage новой его версией:

```
# cp arch/i386/boot/bzImage /boot/<kernel_name>
```

Если используете LILO, выполните lilo; пользователи GRUB - пропустите этот шаг:

```
# /sbin/lilo
```

Теперь выйдите из среды chroot и перезагрузитесь.

```
# exit
```

```
# umount /mnt/gentoo/proc /mnt/gentoo/boot /mnt/gentoo
```

```
# reboot
```

С другой стороны, проблема может быть с вашим загрузчиком; тогда поступите так же, только вместо конфигурирования/компилирования ядра перенастройте свой загрузчик (его перекомпиляция не нужна).

Мой прокси-сервер требует аутентификации, что делать?

Чтобы Portage автоматически использовал реквизиты, определите их в /etc/make.conf:

Листинг 3.4: /etc/make.conf

```
HTTP_PROXY="http://username:password@yourproxybox.org:portnumber"
```

```
FTP_PROXY="ftp://username:password@yourproxybox.org:portnumber"
```

```
RSYNC_PROXY="rsync://username:password@yourproxybox.server:portnumber"
```

Как записать ISO файл на компакт-диск?

Вам необходимо записать его в raw-режиме. Это означает что вам нужно НЕ просто поместить его на CD, а интерпретировать сам файл как CD.

Существует множество утилит для записи CD; рассказывать о них всех будет сизифовым трудом. Однако описать несколько популярных никогда не повредит:

Используя EasyCD Creator, выберите File, Record CD from CD image. Затем поменяйте тип файла на ISO image file. Найдите ISO-файл и кликните Open. При нажатии на Start recording, ISO образ будет записан на диск.

Используя Nero Burning ROM, закройте мастер, автоматически появляющийся при запуске, нажмите Burn Image из меню File. Выберите необходимый образ и нажмите Open. Теперь нажмите кнопку Burn и наблюдайте за процессом прожига вашего нового CD.

Используя **cdrecord**, просто наберите cdrecord dev=/dev/hdc (заменив /dev/hdc именем своего CD-RW привода), а затем имя ISO образа :)

Используя **K3B**, выберите Tools > CD > Burn CD Image. Затем найдите ISO образ, используя окно 'Image to Burn'. Наконец, Нажмите Start.

В Mac OS X Panther, запустите Disk Utility из Applications/Utilities, нажмите Open из меню Images, в главном окне выберите примонтированный образ диска и нажмите Burn в меню Images.

В Mac OS X Jaguar, запустите Disk Copy из Applications/Utilities, выберите Burn Image из меню File, найдите образ ISO и нажмите кнопку Burn.

Какой диск/стадию надо использовать с моим процессором?

Для начала надо узнать, что за процессор вы используете. Предположим, что это Pentium-M. Теперь необходимо найти, с какими инструкциями процессора он совместим. Вы можете проконсультироваться на сайте производителя, однако обычно достаточно погуглить :-).

Если вы не уверены, выберите "меньший" файл CD/stage, например, от i686 или даже универсальный x86 (или эквивалентный в своей архитектуре). Так вы можете быть уверены, что ваша система точно заработает, но будет работать не так быстро, как если бы она была скомпилирована с использованием флагов оптимизации.

Учтите, что существует намного больше параметров, чем те с которыми скомпилированы бинарные пакеты Gentoo. Чтобы правильно выставить -march, ознакомьтесь с gcc guide.

Я не могу выйти в интернет после перезагрузки. В чем может быть дело?

Для начала проверьте, правильно ли определилась ваша сетевая карта ядром. Выполните ifconfig -a, найдите eth0 или wlan0 (если у вас беспроводная карта). Возможно, потребуется загрузить специфичный модуль ядра, чтобы ядро правильно опознало вашу сетевую карту. Если так, то убедитесь, что эти модули ядра перечислены в /etc/modules.autoload.d/kernel-2.6 (или kernel-2.4, если вы до сих пор используете ядро 2.4).

Если вы забыли включить поддержку вашей карты в ядро, придется его переконфигурировать (и пересобрать).

Если ваша сетевая карта найдена ядром, но настроена на использование DHCP, то вы могли забыть установить dhcpcd (emerge dhcpcd). Необходимо загрузиться с установочного CD, чтобы установить dhcpcd.

О том, как восстановить вашу систему, используя установочный CD, смотрите

здесь.

Я хочу загрузить Windows из GRUB или LILO, но вижу только черный экран. Что делать?

Это известная проблема. Windows отказывается грузиться и показывает черный/пустой экран, если установлен не на первый жесткий диск. Чтобы исправить ситуацию, надо, используя настройку загрузчика, "надурить" Windows, чтобы она "думала" что установлена на первый жесткий диск. Учтите, что в следующем примере Gentoo установлен на hda (первый диск), а Windows на hdb (второй). Подправьте ваш конфигурационный файл соответственно.

Листинг 3.5: Пример двойной загрузки для Windows в grub.conf

```
title Windows XP
    map (hd1) (hd0)
    map (hd0) (hd1)
    rootnoverify (hd1,0)
    chainloader +1
```

Листинг 3.6: Пример двойной загрузки для Windows в lilo.conf

```
other=/dev/hdb1
    label=WindowsXP
    table=/dev/hdb
    map-drive = 0x80
    to = 0x81
    map-drive = 0x81
    to = 0x80
```

Это заставит Windows поверить, что она установлена на первый жесткий диск, и грузиться без проблем. Дополнительную информацию можно найти в разделе Документация GRUB и в `man lilo.conf`, смотря, какой загрузчик используете.

Как установить Gentoo, используя файл Stage1 или Stage2?

Настольная книга Gentoo только объясняет, как установить ОС, используя файл stage3. Однако, все еще осталась возможность установки из stage1 или stage2. Возможность оставлена для нужд разработчиков (команда Release Engineering начинает со stage1, чтобы получить stage3), но не обязательна к использования пользователями: файл stage3 тоже очень хорошо подходит для самогенерации системы. Вам потребуется соединение с Интернетом.

Самогенерация (bootstrapping) - построение набора инструментов (библиотека Си и компилятор) для системы, после чего устанавливаются пакеты, входящие в ядро системы. Чтобы сгенерировать систему, выполните установку со stage3. Перед главой настройка ядра, отредактируйте сценарий bootstrap.sh под ваши нужды, а затем выполните его:

Листинг 3.7: Самогенерация системы

```
# cd /usr/portage/scripts
# vi bootstrap.sh

# ./bootstrap.sh
```

Затем пересоберите все ключевые системные пакеты только что собранным набором (библиотека - компилятор). Нам необходимо пересобрать их т.к. в файле stage3 они уже есть:

Листинг 3.8: Пересборка ключевых системных пакетов

```
# emerge -e system
```

Теперь можно переходить к настройке ядра. Прекомпилированные пакеты GRP больше использовать нельзя.

4. Управление пакетами

В каком формате хранятся пакеты?

Пакеты не "хранятся". Вместо этого Gentoo предоставляет набор сценариев, которые могут разрешать зависимости, закачивать исходный код, и компилировать версию пакета именно под ваши нужды. В общем, мы собираем двоичные файлы только для выпусков (версий) и снимков (snapshots). Gentoo Ebuild HOWTO (англ.) рассказывает о содержимом сценариев ebuild в деталях.

Для полных ISO-выпусков, мы создаем полный набор двоичных пакетов в расширенном .tbz2 формате, который совместим с .tar.bz2 и включает метаинформацию, присоединенную к концу файла. Они могут использоваться для быстрой и эффективной установки работающей (правда, не очень оптимизированной) версии пакета.

Также возможно создать RPM (файлы Redhat package manager) средствами Gentoo, но на данный момент нельзя использовать уже готовые RPM для установки.

Я хочу выполнить сценарий ./configure сам. Возможно ли это?

Да, но это не так просто, и не рекомендуется. Т.к. вы все равно должны понимать, как работает Portage, то лучше сделайте заплатку для ebuild-файла, выполняющую то, что вы хотите, и положите ebuild-файл в оверлей Portage (для этого он и существует). Такой подход более подходит для поддержки, и обычно легче. Чтобы узнать больше, см. Ebuild HOWTO (англ.).

Как использовать emerge, если я за межсетевым экраном?

Смотрите вопросы о проху, rsync, и Закачка исходных кодов вручную.

Что делать, если rsync мне не подходит?

Если вы за межсетевым экраном, который не пропускает rsync трафик, то вы можете использовать **emerge-webrsync**, который закачает и установит снимок дерева Portage через обычный HTTP. Обратите внимание на раздел проху этого документа, чтобы выяснить, как закачивать исходные коды и снимки Portage через прокси-сервер.

У меня дома лишь слабое модемное подключение. Могу ли я скачать исходные коды в другом месте, а затем добавить их в систему?

Конечно. Выполните `emerge --pretend package`, чтобы понять, какие программы необходимо установить. Чтобы увидеть, какие исходные коды необходимо скачать и откуда, выполните `emerge -fp package`. Закачайте исходники, и принесите их на любом носителе домой. Скопируйте их в `/usr/portage/distfiles/`, затем просто наберите `emerge package`. Но имейте в виду: это нудное занятие.

Исходные коды хранятся в /usr/portage/distfiles. Безопасно ли удалить их?

Удаление этих файлов никак не повлияет на каждодневную производительность. Однако лучше оставить самые последние версии файлов; часто несколько файлов ebuild выпускаются с расчетом на одну и ту же версию исходных кодов. В таком случае, если вы удалили архивы с исходными текстами, а затем решили обновить ПО, потребуется заново скачать нужные исходники через Интернет.

Существуют программы, разработанные пользователями, для удаления всех исходников, кроме самых последних версий. Заметьте, они хоть и работают, но официально не поддерживаются. Используйте их на свой страх и риск.

Что находится в `/var/tmp/portage`? Можно ли удалить файлы и каталоги из `/var/tmp/portage`?

Во время компиляции, Gentoo сохраняет исходники пакетов в `/var/tmp/portage`. Эти файлы и каталоги обычно удаляются после успешной установки, но не всегда. Можете спокойно очищать этот каталог, если `emerge` не запущен. Чтобы быть уверенным, всегда выполняйте `pgrep emerge` перед очисткой каталога.

5. Использование

Как установить международную раскладку клавиатуры?

Отредактируйте переменную `KEYMAP` в файле `/etc/conf.d/keymaps`. Чтобы консоль заработала правильно с расширенными знаками вашей раскладки, вам может также потребоваться настроить переменные `CONSOLETRANSLATION` и `CONSOLEFONT` у себя в `/etc/conf.d/consolefont` (за дальнейшими сведениями о локализации своей среды обращайтесь к нашему руководству по локализации (англ.)). Затем выполните перезагрузку (`reboot`), или перезапустите сценарии `keymaps` и `consolefont`:

Листинг 5.1: Перезагрузка `keymaps`

```
# /etc/init.d/keymaps restart
# /etc/init.d/consolefont restart
```

Разрешение имен DNS работает только у суперпользователя.

У файла `/etc/resolv.conf` неправильно выставлены права; Выполните `chmod`, как описано ниже:

Листинг 5.2: Смена прав `/etc/resolv.conf`

```
# chmod 0644 /etc/resolv.conf
```

Почему мой пользователь не может использовать свой файл `crontab`?

Вы должны внести своего пользователя в группу `cron`.

Как включить numlock при загрузке?

Если вы работаете в командной строке, вам всего лишь необходимо выполнить **rc-update add numlock default &&/etc/init.d/numlock start**.

Каждый GUI предоставляет собственные средства для этого; смотрите раздел справки или руководство.

Как сделать очистку терминала при выходе?

Для этого добавьте `clear` в ваш сценарий `~/.bash_logout`:

Листинг 5.3: Очистка терминала при выходе

```
$ echo clear >> ~/.bash_logout
```

Если вы хотите, чтобы у новых пользователей это работало автоматически, добавьте ту же строку в `/etc/skel/.bash_logout`:

Листинг 5.4: Очистка экрана при выходе у новых пользователей

```
# echo clear >> /etc/skel/.bash_logout
```

Не удается запускать X приложения из-под root'a (после su)

Похоже, что эта проблема появляется только при графическом входе. Пользователи, использующие **startx**, с ней не сталкиваются. Проблема в Gentoo's PAM, однако выход довольно прост: добавьте следующую строку в файл `/etc/profile`.

Листинг 5.5: Экспорт XAUTHORITY

```
export XAUTHORITY="${HOME}/.Xauthority"
```

6. Сопровождение

ReiserFS и порча файловых систем - как исправить, и т.п.

Если раздел с ReiserFS испорчен, попробуйте загрузиться с установочного диска Gentoo и выполните `reiserfsck --rebuild-tree` на испорченной файловой системе. Эта команда предназначена для восстановления файловой системы, но все же из-за ее повреждения некоторые файлы или каталоги могут быть потеряны.

7. Разработка

Где оставлять сообщения об ошибках?

Используйте нашу систему Bugzilla. Если вы не уверены в актуальности ошибки, можете зайти на канал **#gentoo** в сети IRC FreeNode

Как часто появляются новые стабильные выпуски?

Пакеты Gentoo обычно обновляются вскоре после обновления основного кода программы. А если речь идет о том, как часто выходят файлы стадий/профили/выпуски ISO самого дистрибутива, смотрите страницу Release Engineering Project (англ.) Новые стабильные версии анонсируются в списке рассылки gentoo-announce. Смотрите вопрос об обновлении.

Мой динамик орет как сумасшедший. Как выключить динамик в консоли?

Консольные звуки можно выключить, используя `setterm`:

Листинг 7.1: Использование setterm

```
# setterm -blength 0
```

Если вы хотите, чтобы звуки выключались при загрузке, поместите эту команду в сценарий `/etc/conf.d/local.start`. Однако, это избавит вас от звуков только в текущем терминале. Чтобы выключить звук и на других терминалах, перенаправьте вывод команды на нужный терминал:

Листинг 7.2: Использование setterm (bis)

```
# setterm -blength 0 >/dev/vc/1
```

Нужно заменить `/dev/vc/1` на терминал, в котором вы хотите выключить звук.

8. Ресурсы

Где найти информацию о Gentoo Linux?

Официальная документация Gentoo находится по адресу <http://www.gentoo.org/doc/ru/>.

Могу ли я купить CD с Gentoo Linux?

Установочные диски для всех поддерживаемых архитектур имеются в нашем магазине Gentoo. При покупке дисков в нашем магазине, вы поддерживаете разработку дистрибутива. Поэтому, если можно, покупайте диски у нас.

Также список продавцов находится на странице загрузки Gentoo Linux.

В этом списке нет ответа на мой вопрос. Что делать?

В первую очередь, просмотрите соответствующую документацию, если это не помогло, в Google можно найти всевозможные списки рассылки Gentoo. Чтобы искать в списках рассылки Gentoo, наберите "lists.gentoo.org что-то-там" в строке поиска. Если все попытки тщетны, и вы готовы просто покончить с Gentoo, сходите на irc канал: `#gentoo on irc.freenode.net`.

Примечание: Также можно заглянуть на <http://forums.gentoo.org>, где, ко всему прочему, есть русскоязычное сообщество пользователей - прим. Пер.

Наивные вопросы юных линуксоидов

Вступление

Перейдя из ОС Windows в Linux пользователи сталкиваются с кучей непонятных им вещей. Есть много вопросов которые их тревожат, но спросить они не решаются - боятся. Так давайте же поможем им поскорее разобраться с новой замечательной системой. Может и сами узнаем чего нового. Примечание: Так как Gentoo новичкам не поставить, то этот материал носит скорее пропагандистский характер

Общие вопросы

Как ставить программы?

Первая проблема которая встает перед линуксоидом это установка программ. Программы можно ставить из исходников и бинарных пакетов. На бинарниках завязаны такие дистрибутивы как RedHat, Mandrake, Fedore Core, ASP Linux, ALT Linux, Debian и прочие. С исходниками работают Gentoo Linux и *BSD-системы.

Чтобы установить программу в Gentoo Linux нужно только ввести: emerge программа emerge скачает все необходимое с интернета, скомпилирует и установит. Если же программы нет в дереве портов, то придется ставить ее из тарболла.

Поправочка: В Gentoo тоже можно ставить прекомпиленные (заранее собранные) программы путем вызова команды: emerge --usepkg --getbinpkg программа или короче так: emerge -gk программа Но сборка из исходных текстов предпочтительнее.

Что еще за тарболл?

Все программы в исходных текстах поставляются в виде тарболлов (файлы с расширением tar.gz, tar.bz2, tbz2, tgz, впрочем последние два - большая редкость). Тарболл по своей сути - это просто архив, который можно распаковать следующими командами: tar -xvjf super_proga_0.1.tar.bz2 для tar.bz2(или tbz2) и tar -xvzf super_proga_0.1.tar.gz для tar.gz(tgz). И архив будет распакован в текущий каталог.

Кстати -v - абсолютно лишнее. :) Dron

Я бы сказал так, распаковка `tar -zxvf`, архивы `bz` и `gz2 -jxvf`, удобно запомнить, различается только `j` в начале. => `das`

Некоторым проще запоминать `xzf` как `eXtract Zipped File. j` - для `bzip2 wt`

Дальнейшие инструкции по установке можно прочитать в файле `INSTALL` или `README`, но чаще всего установка выглядит так: `./configure make` Далее от пользователя `root`: `make install`

Но пользователям `Gentoo` лучше прочитать следующий раздел.

Gentoo Way

Установка из тарболлов может показаться достаточно простой на первый взгляд, однако, она может затруднить управление пакетами и создать некоторые проблемы в дальнейшем, поэтому лучше воспользоваться мощностью системой управления пакетами `portage`. Наверняка, о беспроблемной сборке необходимой вам программы уже позаботилось сообщество пользователей и разработчиков `Gentoo`. Чтобы узнать это, попробуйте найти нужную вам программу в базе портежей набрав: `emerge -s [название программы]` или по описанию: `emerge -S [ключевое слово]` что правда занимает достаточно много времени, и лучше использовать специально для этого написанные утилиты, например `esearch emerge esearch eupdatedb esearch [название программы]` или по описанию `esearch -S [ключевое слово]` так же можно проводить поиск по категориям. Например `esearch -F mail-client` покажет все почтовые клиенты, информация о которых есть в дереве портежей. Так же при поиске можно использовать регулярные выражения.

Неплохим подспорьем может послужить GUI фронтенд к портежам, например `Porthole`(интерфейс `GTK`). Для любителей `KDE` есть `Kentoo`(встраивается в центр управления) и `Guitoo`.

Если вам потребовалось установить программу из неофициального `ebuild'a`, сначала прочтите `Portage_Overlay`.

Если же `ebuild` скрипта найти не удалось или нужно наложить дополнительные патчи, то это значит, что пора освоить написание скриптов. Для этого можно воспользоваться `skel.ebuild`ом который находится в каталоге `portage`. Так же не повредит прочтение `en:HOWTO Create an Updated Ebuild`.

Удобной средой разработки и правки `ebuild` скриптов является `abeni`

А кто такой root и как им стать?

Говоря языком Windows, root - это администратор. Но в отличие от Windows, root - абсолютный хозяин системы. Он может сделать все, даже удалить системные файлы, или произвести низкоуровневое форматирование жесткого диска. По идее, root всегда знает что делает, поэтому такая власть вполне оправданна. Рутром можно стать несколькими способами:

первоначально войти систему под этим аккаунтом.

через su. Для этого в консоли введите su

```
[user@gentoo user]$su
```

Password:

```
[root@gentoo user]#
```

Пользователь user должен быть в группе wheel. При вводе пароля он отображаться не будет.

через sudo. Это дополнительная программа, которую можно установить командой:

\$ emerge sudo

Ее преимущества перед обычным su заключаются в том, что можно протоколировать все выполненные команды, а значит понять, кто и как именно сделал что-то неправильно. :) Кроме того, можно разрешить определенным пользователям выполнять только определенные команды с правами root, что существенно повышает безопасность системы. И, наконец, для использования sudo не нужно знать пароль root - достаточно помнить свой. ;) Все эти и другие настройки производятся редактированием файла конфигурации sudo - **/etc/sudoers**.

Пользоваться ей очень просто:

\$ sudo команда

(например, "sudo ls -l"). Если же вам нужно выполнить не одну команду, а просто временно стать root, используйте следующий синтаксис:

\$ sudo -s

Для того, чтобы перестать быть root после этого, достаточно ввести команду

\$ exit

Секреты консоли

Основные команды

Можно на 100% гарантировать, что количество команд в вашей Linux системе больше пары тысяч. Но практически все линуксоиды используют из этого количества не более 100 команд, хотя и это, пожалуй, завышенное число. Человеку, который линукс видел только на экранах, будет полезно узнать, как же использовать эту консоль.

Команда man

Linux - отлично документированная система. Подробную информацию о программе, команде или основным конфигурационным файлам можно получить при помощи команды `man`. Например `man whereis` выдаст вам подробную справку по использованию команды `whereis`, а команда `man fstab` выдаст синтаксис конфигурационного файла `/etc/fstab`.

Если вам не удалось найти нужную информацию, можно заглянуть в `/usr/share/doc` (в Gentoo Linux так же находится в `/usr/doc`). Там хранится документация программ.

Примечание: Для более детальной информации обратитесь в соответствующий раздел Страницы MAN

Команда cd

Без этой команды навигация по файловой системе будет невозможна. Смысл её простой - сменить текущий каталог. А синтаксис такой: `cd /home/vasya/katalog` Можно использовать как абсолютные, так и относительные пути к папкам.

Абсолютный путь - это путь указанный от корневого каталога(`/`).

Относительный путь - это путь указанный относительно текущего каталога.

Например, если вы находитесь в каталоге `/home/vasya`, то для перехода в `/home/vasya/katalog` вам нужно ввести только `cd katalog` Необходимо отметить, что в Linux нет такого понятия, как "диски". Все разделы жестких дисков, CD-приводы, сетевые диски и т.д. отображаются в виде каталогов файловой системы (обычно в `/mnt`).

Чтобы узнать в каком каталоге вы сейчас находитесь используется команда `pwd`.

Команда ls

Используется для получения списка файлов и каталогов. Например, `ls` выведет нам содержимое текущего каталога. Для вывода информации о каталоге, в котором вы сейчас не находитесь, используется `ls /etc`. Наверняка вас не удовлетворят только имена файлов. Для увеличения количества выводимой информации используйте опцию `"-l"` `ls -l`. Теперь `ls` выведет вам много полезной информации, такой как права, время последней модификации, размер и прочее. `ls -la` с опцией `"-a"` покажет скрытые файлы (by morgoth)

Команда `cp`

Предназначена для копирования файлов и каталогов. Например, скопируем файл `/home/vasya/katalog/linux.txt` в каталог `/home/vasya`. Делается это так: `cp /home/vasya/katalog/linux.txt /home/vasya` А вдруг мы хотим чтобы файл `linux.txt` получил имя `gentoo.txt`? Тогда делаем так: `cp /home/vasya/katalog/linux.txt /home/vasya/gentoo.txt` По умолчанию `cp` не копирует каталоги. Чтож, заставим его это делать. Скопируем `/home/vasya/katalog` в каталог `/tmp` так: `cp -R /home/vasya/katalog /tmp` Ключ `-R` указывает на рекурсивное копирование. Как и в случае с файлами, можно поменять конечное имя каталога: `cp -R /home/vasya/katalog /tmp/новое_имя` У `cp` есть еще один полезный ключ: `-p`. Он предназначен для сохранения всех параметров файла (владельца, прав доступа, битов `suid`, `sgid`). Рекомендуем всегда указывать ключ `-p`, особенно если вы частенько сидите под суперпользователем (чего делать категорически не рекомендуется).

Команда `rm`

Предназначена для удаления файлов. Например, удалим файл `/home/vasya/katalog/linux.txt`: `rm /home/vasya/katalog/linux.txt` или если каталог `/home/vasya/katalog` текущий, тогда можно так: `rm linux.txt` Будьте осторожны, удаленный файл нельзя восстановить, как в `Dos` или `Windows`.

Для удаления пустого каталога `/home/vasya/testdir` введите `rmdir /home/vasya/testdir` Не бойтесь указывать абсолютный путь к каталогу - будет удален только последний элемент пути. Для удаления этого же каталога рекурсивно (вместе с подкаталогами и файлами) введите `rm -r /home/vasya/testdir`

Команда `df`

Предназначена для просмотра оставшегося места на диске `df -h` Ключ `-h` позволяет увидеть вывод в удобном для человека формате (например `1K 234M 2G`)

А как мне быть если путь или имя очень длинные?

Длинные пути к файлам и каталогам - довольно частое явление. И бороться с этим достаточно легко. Все современные оболочки (aka shell) обладают возможностью автодополнения. Что это значит? Например, есть файл `cd /home/vasya/katalog/vtoroi_katalog/oooooooooooooooochendlinoeimyakataloga`. Ясно, что вбивать имя полностью - это сомнительное удовольствие. ;) Поэтому поручим часть работы shell'у. Мы пишем только часть имени, а остальное будет дополнять шелл. Для дополнения необходимо нажать Tab. Например: `cd /h<TAB>/v<TAB>/k<TAB>/v<TAB>/o<TAB>`

Примечание: в оболочке Zsh достаточно написать

`cd /h/v/k/v/o` и нажать "Tab". Оболочка сама развернет сокращение в полный путь. Но тут могут возникнуть проблемы, в том случае, если в каталоге `/home` есть 2 подкаталога `/home/vasya` и `/home/vika`. Шелл не сможет понять куда же вы хотите попасть. Чтобы помочь ему нужно указать еще одну букву в пути, то есть `/home/va<TAB>`, чтобы выбор был однозначным. Если вы не помните части пути, то можете два раза нажать Tab и вы увидите список возможных вариантов. Например: `ls /home/<TAB><TAB>` выведет на экран содержимое директории `/home`.

Как сделать чтобы программа запущенная из терминала работала после его закрытия?

Такая ситуация возникает довольно часто. Например вам необходимо обновить систему по ssh находясь далеко от своего компьютера, но удерживать связь постоянно вы не можете. Для этой цели можно использовать программу `nohup`. Например, так можно запустить `xmms`: `nohup xmms` Вызываемой программе можно передавать аргументы. Теперь при закрытии терминала `xmms` будет работать, работать и работать ;) или: `xmms &` но это только освободит терминал. После закрытия терминала приложение завершиться.

Так же можно использовать программу `screen`.

Графическая оболочка X Window

А как работать с буфером?

Работа с буфером обмена в системе X Window гораздо удобнее, чем в ОС Windows. Для копирования и вставки текста делаем так:

выделяем нужный текст мышкой

в нужном месте щелкаем на среднюю кнопку или колесико мышки, и текст

вставляется там, где курсор.

Конечно, вы можете использовать и комбинации ОС Windows (Ctrl+C, Ctrl+V), однако, следует помнить, что комбинация Ctrl+C экстренно остановит выполнение команды в терминале, другими словами, попытавшись скопировать вывод программы комбинацией Ctrl+C, вы рискуете убить ее выполнение. Будьте внимательнее.

Возможно также использование комбинации Ctrl+Insert для копирования в буфер и Shift+Insert для вставки из него.

Как заставить иксы стартовать при загрузке?

Для этого их необходимо добавить в уровень запуска default. Для этого необходимо выполнить: `rc-update add xfs default` и `rc-update add xdm default`. Первой командой мы заставили автоматически стартовать сервер шрифтов, а вторая указывает на автозапуск менеджера входа в систему. Для того, чтобы вместо xdm запускался другой менеджер входа необходимо немного отредактировать `/etc/rc.conf`.

Файл: `/etc/rc.conf`

..

`DISPLAYMANAGER="kdm"`

..

Теперь вместо xdm будет запускаться kdm.

Файл: `/etc/rc.conf`

..

`DISPLAYMANAGER="gdm"`

..

Теперь вместо xdm будет запускаться gdm более красивый менеджер.

Термины

Мы стараемся придерживаться терминов, уже определенных здесь, а сталкиваясь с новыми трудностями при переводе и редактировании — пополняем словарь.

Кроме того, мы стремимся, чтобы терминология в разных пакетах системы была единой независимо от их происхождения.

Вики (WiKi) (от гавайского: быстрый)—

гипертекстовая среда (обычно Веб-сайт) для сбора и структуризации письменных сведений (в программировании: документации, советов, часто задаваемых вопросов(FAQ), мини-руководств(HOWTO)).

Характеризуется следующими признаками:

- Множество авторов. Некоторые вики могут править все посетители.
- Возможность многократно править текст посредством самой вики-среды (вебсайта), без применения особых приспособлений на стороне редактора.
- Проявление изменений сразу после их внесения.
- Разделение информации на страницы, где у каждой своё название.
- Особый язык разметки, позволяющий легко и быстро разметать в тексте структурные элементы, форматирование, гиперссылки и т. п.
- Учёт изменений (учёт версий) текста и возможность отката к ранней версии.

Для создания вики-среды необходимо особое ПО — движок вики. Это частный вид системы управления сайтом, довольно простой в своём устройстве и функциональности, ибо почти все действия по структуризации и обработке сведений делаются пользователями вручную. Главная особенность -- посетители сайта могут сами редактировать содержимое этих страниц и создавать новые. Wiki сделаны так, чтобы редактировать страницы было просто; для редактирования ничего кроме программы веб-браузера не нужно.

ALSA, или Advanced Linux Sound Architecture —

продвинутая Звуковая Архитектура Linux, обеспечивает аудио- и MIDI- (Musical Instrument Digital Interface — Цифровой Интерфейс Музыкальных Инструментов) функциональность в операционной системе Linux. ALSA является основной звуковой подсистемой в ядрах 2.6, заменившей собой OSS (Open Sound System - Открытая Звуковая Система), которая использовалась в ядрах 2.4

Х сервер (или Иксы) -

обычного пользователя могут испугать мысли о необходимости печатать в командной строке. Почему бы ему не прокладывать свой путь на территории свободы выбора Gentoo (и Linux вообще), просто «водя мышкой»? Ну, конечно, вы сможете это сделать :-). В Linux предлагается широкое разнообразие пользовательских интерфейсов и сред, которые можно установить поверх установленной у вас системы.

Это один из самых больших сюрпризов, с которым сталкиваются новички: графический интерфейс пользователя — это всего лишь прикладная программа, выполняемая в системе. Он не является частью ядра Linux или каких-либо других компонентов системы. Это — мощный инструмент, позволяющий полностью

задействовать графические возможности вашего компьютера.

Так как стандартизация очень важна, для организации прорисовки и перемещения окон по экрану, взаимодействия пользователя с мышью и клавиатурой, а также других важных базовых аспектов, был создан стандарт под названием X Window System, сокращённо X11 или просто X. Он используется в Unix, Linux и других Unix-подобных операционных системах во всем мире.

Приложение, которое дает пользователям Linux возможность запускать графические интерфейсы пользователя и использует стандарт X11, называется Xorg-X11, ответвление проекта XFree86. Проектом XFree86 было решено использовать лицензию, возможно, несовместимую с лицензией GPL, поэтому рекомендуется использовать Xorg. Пакет XFree86 больше не поддерживается в официальном дереве портежей.

Проект X.org

Проектом X.org создана и поддерживается бесплатно распространяемая реализация системы X11 с открытым исходным кодом. Это открытая инфраструктура рабочей станции, основанная на X11.

Xorg обеспечивает интерфейс между устройствами вашего компьютера и нужными вам графическими программами. Кроме того, Xorg является полностью сетевым, то есть вы можете запускать программу на одной системе, а отображать ее — на другой.

Ссылка на оригинал: <http://www.rugentoo.org/glossary/index.html>

<i>Термин</i>	<i>Перевод</i>
activation	активация; активизация; включение; приведение в действие; запуск
architecture-specific	архитектурно-зависимый (о режиме или параметре); посвящённый определённой архитектуре (о руководстве)
account	учётная запись
advanced	расширенный (напр., настройка); углублённый (напр., обучение)
attribute	атрибут (например, файла); свойство, параметр (иногда)
authentication	аутентификация
authorization	авторизация
authorized	авторизованный; допущенный; санкционированный
autocompletion	автодополнение (слов в строке)
baselayout	так и оставить — baselayout; при первом упоминании в тексте руководств хорошо бы в

<i>Термин</i>	<i>Перевод</i>
	скобочках писать (размещение основных файлов и сценарии инициализации системы)
basic	базовый, основной; <i>элементарный</i> (lingvo.ru)
basics	основы, основные действия
binary	двоичный; бинарный
binary [executable]	исполняемый файл
bind-mounting ...	монтирование ... с параметром bind (см. man mount :)
bindings	привязки при первом упоминании в тексте имеет смысл писать в скобочках английский термин
board of trustees	совет попечителей
bonding (, channel trunking)	объединение ([магистральных] сетевых каналов)
boot runlevel	загрузочный уровень исполнения
bootable (disk)	загрузочный (диск)
bootloader	[начальный] загрузчик; загрузчик ОС
bootsplash	загрузочная заставка
bootsplash- enabled kernel	ядро с [включённой] загрузочной заставкой
boot (bootstrap) partition	загрузочный раздел
bootstrapping	начальная загрузка; самонастройка; самозагрузка [само]генерация (сборка) системы (в некоторых контекстах в Gentoo)
branch (portage)	ветвь (дерева портежей)
bridge (, network)	[сетевой] мост
broadcast [address]	широковещательный адрес
browse to	открыть [веб-страницу]
browser	браузер
bug	запрос (требующий реагирования); ошибка (разновидность запроса)
bug report	отчёт об ошибке <i>или</i> запрос [форма запроса]
bugtracking system	система распределения запросов (<i>внимание! не багов, не ошибок — потому что не только их; и</i>

Термин	Перевод
	<i>не отслеживания, управления или контроля — потому что распределения — это корректный перевод одного из американизмов)</i>
build	сборка (программный пакет)
cache	кэш
card	плата (деталь компьютера); карта [PCMCIA, флэш-памяти]
cascading (stacked) profile	каскадный профиль
CD	компакт-диск
change request	запрос на изменение (разновидность запроса)
chapter	глава [руководства]
child [tag]	вложенный [тег] (о XML)
choice	выбор, вариант выбора; альтернатива; пункт меню (документация по Gentoo помешана на идеологической теме свободы выбора пользователя, русский перевод стоит разнообразить)
choices	мн. ч. от choice; возможность (свобода) выбора
chroot	[команда] chroot; изменение корневого каталога
chrooted environment	среда с изменённым корневым каталогом
Code of Conduct	кодекс поведения
coldplugging	холодное подключение (с выключением оборудования)
color depth	глубина цвета; тоже самое что bits per pixel (bpp)
compose	составлять; верстать (о разметке XML)
configuration file	конфигурационный файл; файл настроек
configuring; configuration	настройка [системы, чего-либо]; конфигурация [ядра]
content	текст, содержимое; разделы (о разметке GuideXML)
contributor	участник
council council meeting	совет собрание совета
daemon	демон

Термин	Перевод
	(программа слежения за системными событиями; тж. служба)
declare	объявлять (переменную)
default [by -]	по умолчанию
define	задавать, указывать, определять (последний вариант используйте с осторожностью, чтобы не путать определение в смысле установки значения с определением в смысле обнаружения)
desktop (environment) desktop (computer)	графическая среда (о KDE/Gnome и т.п.); рабочая станция (настольный компьютер, в отличие от сервера);
desktop documentation resources	документация по графической среде Gentoo; документация по использованию в качестве рабочей станции
dependency	зависимость
dependency [package]	зависимый пакет
developer	разработчик
device file	файл устройства
device manager	диспетчер устройств
device node	специальный файл [устройства]
dial-in access dial-up access	[входящий] коммутируемый доступ
directory	каталог
display manager	диспетчер отображения (не экрана!)
documentation listing	перечень документации
documentation repository	хранилище документации
documentation resources	документация (имеется в виду спектр доступной документации)
down [interface]	отключён
download	загрузка (из интернета); загрузить
drive	привод
driver	драйвер, <i>мн.ч.</i> драйверы
dump	программа [резервного копирования] dump;

Термин	Перевод
	дамп [памяти]
ebuild	сборочный файл; пакет; файл ebuild программа ebuild (при переводе транслитерация не допускается из-за исключительной неблагозвучности для русского читателя!) Примечание: <i>e-build</i> — вероятно, авторское сокращение от <i>emerge build file</i>
emerge	[команда] emerge; средство установки/удаления программ Примечание: <i>автор Gentoo хотел, чтобы программы «являлись» как по волшебству (изначально система называлась Енох в честь загадочного библейского персонажа); для этого придумал команду «emerge хуз», что буквально переводится как «хуз, явись!».</i> Остальные термины системы установки подогнаны под неё
environment variables	переменные среды
experience [, improvement]	впечатление (от Gentoo); [повышение] удобства работы с Gentoo
FAQ, frequently asked questions	ЧаВо, часто задаваемые вопросы
feature	возможность [программы], функция; особенность
feature request	запрос на добавление функции (разновидность запроса)
fetch	извлечь (ebuild/portage tree)
firewall	межсетевой экран
firmware	прошивка; встроенное ПО
floppy [disk]	дискета
framebuffer	кадровый буфер (используется для промежуточного хранения исходных данных для программного или аппаратного формирования <i>видеокадра</i> или <i>полукадра</i> для вывода на экран монитора)
front-end	внешний интерфейс; пользовательский интерфейс
functionality	возможности, функции
generic	универсальный (о цели компиляции); общий, обычный (generic printer — обычный принтер)
Gentoo	[система] Gentoo (женский или общий род)
Gentoo Documentation	проект документации Gentoo

Термин	Перевод
Project	
Gentoo Handbook	настольная книга Gentoo (<i>не</i> руководство)
Gentoo Minimal Installation CD Minimal LiveCD	Минимальный установочный компакт-диск Gentoo
Gentoo Universal Installation CD	Универсальный установочный диск Gentoo
groupware	[сервер, программа, пакет] обеспечения взаимодействия (рабочих групп)
guide	руководство (требуется термин, отличный от Handbook);
hard link	жёсткая ссылка
hardened [Gentoo project]	укреплённая система, проект укрепленного Gentoo
hardware	аппаратура; устройства
hardware [device]	устройство
hardware requirements	требования к оборудованию
herd	группа [сопровождения пакета] (не стадо и не толпа!)
highlighted	выделенный (об элементе или о тексте)
host	узел, компьютер
howto	описание [выполнения чего-то] «Как [сделать то-то]» — в заголовке
hub	(сетевой) концентратор
image [kernel, disk ...]	образ
image [graphics]	изображение
init script; initscript	сценарий инициализации (**)сценарий запуска??? тж. runlevel script
initialization	инициализация [переменной]; подготовка [раздела]
initrd (initial root disk)	начальный корневой диск (initrd); диск initrd
initrd image	образ диска initrd
injecting [into Portage]	внедрение [ПО в дерево Portage]

Термин	Перевод
installation	установка
installation CD	... установочный [компакт-]диск; «...» — тип диска (минимальный, универсальный и т.д.)
installer	программа установки
instructions	указания; инструкции
internet	интернет (с маленькой буквы, м.р., склоняется — интернета, интернетом)
invalid	недопустимый
IP address	IP-адрес (сетевой адрес протокола интернета в виде последовательности чисел)
issue	см. bug
job	задание (пакетное; на печать и т.п.)
jumper	перемычка
kernel	ядро
keyword	ключ (в контексте типа ~arch keyword); ключевое слово
keymap	раскладка [клавиатуры]
laptop	ноутбук, портативный компьютер
latest	новейший
lightweight	упрощённый (о синтаксисе)
linker	компоновщик
linking [static]	компоновка
linking [dynamic]	динамическое связывание
LiveCD	загрузочный [установочный] [компакт-]диск (LiveCD)
LiveUSB	загрузочный USB-накопитель (LiveUSB)
locale	регион; [набор] региональных настроек (само слово locale означает «местность»)
log	журнал [регистрации событий]
logger	средства журналирования
login [user's]	учётная запись
login [process]	вход в систему
login [as]	войти в систему [как / под именем]
loopback connector	[разъём-] заглушка

Термин	Перевод
machine	машина; компьютер
maintainer	куратор (пакета и т.п.)
maintenance, package	сопровождение (пакета)
man page	страница справки
manager	диспетчер (о программе)
mark [up]	вёрстка, оформление, разметка (текста документа XML)
mask [package]	замаскировать [пакет] (не «заблокировать»)
MBR, master boot record	главная загрузочная запись
merge [with; together] merge [software with system]	объединять [с]; устанавливать [ПО]
metastructure	организация проекта, метаструктура
modeless	немодалный
mount	монтировать; смонтировать [дисковый раздел] (включить в состав файловой системы) Вопрос к обсуждению: хочется в будущем уйти от этого термина, например, заменив его на подключение/отключение или присоединение/отсоединение. Термин произошёл от процесса установки в накопители больших ЭВМ пакетов дисков (которые действительно надо было монтировать [закреплять], иначе они могли физически улететь, покалечив кого-нибудь), и потерял смысл.
mountpoint	точка монтирования [дискового раздела] (см. комментарий к mount)
multilib	многорежимный профиль (с одновременной поддержкой нескольких наборов библиотек для различных режимов среды пользователя, например, 32- и 64-битного)
naming	именование
nano	nano (текстовый редактор)
netmask	маска [подсети]
networking	подключение к сети; сетевое подключение (в контексте настройки узла)
NIS	NIS (network information service — служба сетевой информации)
open documentation	лицензия открытой документации

Термин	Перевод
license	
option	параметр [командной строки]; директива вариант [выбора]; возможность [пользователя]; (никогда не опция!)
optional	необязательный [шаг, компонент]; дополнительный; факультативный
output, [program]	сообщения [программы]
overlay	оверлей
override	подмена, подменить; переопределение, переопределять
overwrite	перезаписать
package	пакет [программного обеспечения]
Package CD	компакт-диск с пакетами (Package CD)
package maintenance system	система управления (!) пакетами
packaging system	система сборки (!)
paragraph	абзац (не параграф!)
part	часть [руководства] (общая структура: часть-глава-раздел-абзац; раздел намеренно для совместимости подчинён главе в нарушении логики)
partition	[дисковый] раздел; раздел устройства; (не раздел на устройстве! что-либо записываем тоже «в» раздел!)
partitioning	разбиение
patch	сущ. заплатка, исправление, файл исправлений
patched	исправленный
PC	компьютер (персональный компьютер, если есть противопоставление миниЭВМ)
ping	команда ping; проверка отклика [командой ping]
pipe	канал (механизм передачи данных между процессами)
policy	правила; положение (документ); политика
pool	группа; пространство (ресурсов, памяти)

Термин	Перевод
portage [package]	портеж; пакет (системы Portage)
Portage [system]	[система] Portage (женский или общий род, без перевода); <i>Примечание: для «явления» программ (см. примечание к emerge) автор взял за основу систему port, и предположительно использовал игру слов:</i> port-portage <i>сохраняется узнаваемость</i> port-age — «век портов», «зрелый port» portage — «кипа, связка»
portage tree	дерево портежей
portage snapshot	снимок [копия] дерева портежей [в некоторый фиксированный момент времени]
prebuilt [package]	готовый [двоичный] пакет
precompiled [package]	см. prebuilt [package]
printer-friendly version	версия для печати
private key	секретный ключ [шифрования]
privilege	полномочия
proxy	прокси-сервер («сервер-посредник» для доступа в интернет)
public key	открытый ключ [шифрования]
purpose	назначение (в заголовке раздела документа)
quick installation guide	краткое руководство по установке
raise [privilege]	превышать полномочия
reboot	перезагрузка; перезагрузить
reference [manual]	справочное руководство
reference [platform]	эталонная платформа
release notes	замечания к выпуску
release, [software]	выпуск [ПО]
release [version]	окончательная (стабильная) версия
relevance	соответствие (напр., критерию); отношение (к чему-либо)
remote	удалённый [компьютер]; дистанционный

Термин	Перевод
rendered document	представление документа (о XML)
repository	хранилище
rescue [disk]	аварийное восстановление; диск аварийного восстановления
return [an error]	выдавать [ошибку]
root [user]	[учётная запись] root; администратор
router	(сетевой) маршрутизатор
runlevel	уровень исполнения
runlevel script	сценарий инициализации (тж. initscript) (**)
script	сценарий
section [, guide]	раздел [руководства]
security	безопасность; защита
security advisory	предупреждение об уязвимости (защиты системы)
serial	последовательный
service	служба (тж. демон)
setting (of variable)	значение; установка значения переменной
signature	признак; идентификатор; подпись
skin	образ; внешний вид? [вариант] оформления? костюм?
slot	слот
socket [network]	сокет
software	программа; программное обеспечение
software management tool	система управления программным обеспечением
source code	исходные коды (**) как лучше: коды или код?
source mirror	зеркало-источник [Gentoo] (хотя это и зеркало исходных кодов, на нем есть и двоичные образы)
specific	определённый, конкретный (<i>очень редко</i> специфичный!)
stage	стадия (в Gentoo — генерации операционной системы, вообще — часто установки или загрузки системы)
stage file (archive)	файл [архив] стадии; stage-файл [stage-архив] (при упоминании сохранять английское написание файла, чтобы его находили)

Термин	Перевод
start [device]	запуск [устройства]
statement	выражение
stop [device]	остановка [устройства]
striping [raid]	чередование (RAID 0, RAID 5)
subarch[itecture]	разновидность архитектуры
summary	аннотация (краткое описание документа)
superuser	администратор (ср. пользователь)
supplicant [WPA]	запросчик
swap	подкачка
switch, [command line]	параметр [командной строки]
switch, [network]	(сетевой) коммутатор
symlink	символьная ссылка (тж. в других ОС — softlink, alias, shortcut, shadow: псевдоним, ярлык, тень)
syntax	синтаксис (выражения); формат, структура (текстового конфигурационного файла)
system information	системные параметры (в руководстве по настройке), сведения о системе (при ответе на запрос пользователя)
system, [computer]	[компьютерная] система
tag (xml/html)	тег (в XML или HTML; при указании на метки вместе с содержимым — элемент); метка (проч.)
tarball	архив (набор файлов, помещённый в один файл-контейнер)
target [of compilation]	цель [компиляции]
time zone	часовой пояс
tips & tricks	полезные советы (<i>не применяются выражения: «советы и хитрости», «маленькие хитрости»</i>)
tool	средство; утилита; (иногда, по смыслу) служба
toolchain	(**) средства компиляции?
traffic	трафик; движение, поток (употребляется реже)
tweak	ухищрение (сущ.) подстроить (гл.)

Термин	Перевод
unmount	размонтировать (см. комментарий к mount)
up [interface]	включён
upstream [developers]	разработчики самого пакета (входящего в состав Gentoo)
USE flags	USE-флаги (признаки использования функций, указываемые в переменной USE)
USE-expanded variable	переменная, преобразующаяся в USE флаг
user	учётная запись (о реквизитах пользователя); пользователь (о человеке)
user-space	режим пользователя; [адресное] пространство пользователя
userland	режим пользователя (см. user-space) (**)
variable	переменная; значение переменной (будьте внимательны!)
verbose	подробный; расширенный [вывод, листинг] подробные сообщения
wallpaper	обои (на графическом рабочем столе)
web site	веб-сайт
window manager	диспетчер окон
wireless (сущ.)	беспроводное подключение; беспроводная сеть; беспроводное устройство (по контексту)
wrapper	обёртка? упаковщик? (**)

Запрещённые термины, или КАК НЕ НАДО

Употребление этих терминов — лучший способ «облегчить» работу редактора: при этом задержка публикации вашего материала (из-за того, что правка документа будет вызывать непреодолимую грусть) — обеспечена ;-)

Неверный перевод	Правильно
брандмауэр, firewall	межсетевой экран
карта(видео/сетевая/...)	плата (а карта — игральная [card] или географическая [map])
гид	руководство (гид — это экскурсовод)
десктоп	настольный (прил.) [графическая] среда (сущ.)
директория	каталог (по-русски директория — справочник или орган

Неверный перевод	Правильно
	<i>власти)</i>
ебилд	сборочный файл [ebuild] <i>(догадываетесь, почему?)</i>
локаль	регион; региональные настройки
менеджер	диспетчер
опция, опциональный	возможность; функция; параметр
патч	заплата, исправление
переменные окружения	переменные среды <i>(речь идёт о среде исполнения)</i>
продвинутый	расширенный, углубленный
релиз	выпуск <i>(о программе)</i> издание <i>(о книге)</i>
репозиторий документации	хранилище документации
сигнатура	признак, идентификатор, подпись
система уведомления о ошибках и изменениях; система управления изменениями система отчёта об ошибках	система распределения запросов
скрипт	сценарий
суперпользователь сверхпользователь	администратор; [корневая] учётная запись root (root administrative account) <i>Споры о разности понятий в разных ОС — отставить (я не виноват, что именно в Linux упрощённая модель допуска)</i>
сценарий управления сервисом	сценарий инициализации <i>(расширение смысла разумно, но это изменение есть смысл вносить в оригинальный текст, иначе будет путаница)</i>
ускорение не двухмерной графики	ускорение двумерной (нет буквы x) графики
фреймбуфер	кадровый буфер <i>(термин устоялся, когда Линус ещё не родился)</i>

Текст документа распространяется на условиях лицензии [Creative Commons — Attribution / Share Alike](#).

Этот неофициальный сайт создал и поддерживает [Алексей Чумаков](#). © 2005—2006 проект «русский генту».

MAN (справки по командам)

Emerge

Имя:

emerge - интерфейс командной строки к системе портежей (Portage system)

Синтаксис:

emerge [опции] [действие] [ebuild | файл_tbz2 | класс | зависимость]...

emerge [опции] [действие] файл_ebuild

emerge sync | rsync

emerge --help -h [system | config | sync]

emerge --version

Описание:

emerge - это полный интерфейс командной строки к системе портежей. Используется, в основном, для установки пакетов, причем, emerge может автоматически обрабатывать любые зависимости, заданные в требуемом пакете. Утилита emerge может также обновлять дерево портежей, предоставляя доступ к новым и обновленным пакетам. Утилита emerge также обеспечивает бесппроблемное обновление версий установленных пакетов. Позволяет работать с пакетами как в виде исходных кодов, так и в двоичном виде, также может использоваться для создания двоичных пакетов с целью дальнейшего распространения. Пакеты EBUILD, файлы TBZ2, классы и зависимости

Основное назначение утилиты emerge - установка пакетов. Устанавливаемые пакеты можно задать одним из четырех основных способов: как пакет ebuild, как файл tbz2, как класс или как зависимость.

ebuild

ebuild необходимо задавать, как минимум, в виде допустимого имени в каталоге пакетов портежей (valid Portage package directory name) без указания версии или категории, например, portage или python. Можно добавить также категорию и номер версии, например, sys-apps/portage или python-2.2.1-r2. Утилита emerge

игнорирует завершающую косую черту, так что, можно использовать автоматическое завершение для имён файлов. В качестве значения ebuild можно также задавать реальное имя файла, например, `/usr/portage/app-admin/python/python-2.2.1-r2.ebuild`.

Файл_tbz2

Файл_tbz2 должен быть корректным файлом .tbz2, созданным с помощью команды

`ebuild <пакет>-<версия>.ebuild` пакет или

`emerge --buildpkg [категория/]<пакет>`, или

`quickpkg /var/db/pkg/<категория>/<пакет>`.

Класс

Классы пакетов - удобный способ обозначить большую группу пакетов. В настоящее время поддерживается два класса: `system` и `world`. Класс `system` содержит набор пакетов, считающихся необходимыми для правильной работы системы. Класс `world` содержит все пакеты в системе плюс все пакеты, перечисленные в файле `/var/cache/edb/world` или `/var/lib/portage/world` (`=>portage-2.0.51`). Подробнее см. далее в разделе Файлы

Учтите, что эти команды обычно используются вместе с `--update`.

Зависимость

Зависимость описывает ограничения на пакет, который будет установлен. Например, `>=dev-lang/python-2.2.1-r2` что соответствует последней доступной версии языка Python, большей или равной 2.2.1-r2. Аналогично, `<dev-lang/python-2.0` соответствует последней доступной версии языка Python до 2.0. Учтите, что во многих командных интерпретаторах потребуется замаскировать символы вроде `'<'` и `'='` ; чтобы избежать проблем с маскировкой, берите зависимость в одиночные или двойные кавычки.

Действия

Нет действия

Если действие не указано, в систему добавляются указанные пакеты и удовлетворяются все их зависимости. В качестве аргументов можно указывать пакеты ebuild, файлы tbz2, классы или зависимости. Учтите, что для установки

файла `tbz2` необходимо указать опцию `--usrpkg`. Пакеты добавляются в конце файла `world`, так что они учитываются при последующих обновлениях.

--clean

-c

Очищает систему, удаляя пакеты, не влияющие на функциональность системы. В качестве аргументов можно указывать пакеты `build`, классы или зависимости. Например,

`emerge clean binutils` удаляет старые версии `binutils`;

`emerge clean net-www/mozilla-0.9.9-r2` удаляет указанную версию `Mozilla`.

Использовать эту команду, в общем случае, безопасно. Учтите, что пакеты в единственном экземпляре (`unslotted packages`) команда `clean` не удаляет.

--depclean

Выявляет все пакеты, установленные в системе без видимой причины. Утилита `emerge` генерирует список пакетов, которые должны быть установлены, на основе списка системных пакетов и файла `world`. Затем она сравнивает этот список со списком фактически установленных пакетов; найденные различия выдаются как ненужные пакеты, которые предлагается удалить. Предупреждение: Удаление некоторых пакетов может привести к тому, что пакеты, связанные с удаленными, перестанут работать и начнут сообщать о недостающих библиотеках. Снова установите с помощью `emerge` пакет, который перестал работать, и эта проблема будет решена. Учтите, что изменения флагов `USE` могут существенно повлиять на результат, выдаваемый при выполнении действия `depclean`.

--help

-h

Выдает справочную информацию по утилите `emerge`. Добавление одного из прочих перечисленных дополнительных аргументов приводит к выдаче более детальной справочной информации по соответствующему действию. Встроенная справочная документация утилиты `emerge` обновляется чаще, чем эта страница справочного руководства; обратитесь к ней при возникновении проблем, которые не поможет решить эта страница справочного руководства.

--info

Выдает информацию для включения в отчеты об ошибках. Эта информация

помогает разработчикам при решении проблем, о которых вы сообщаете. Включайте, пожалуйста, эту информацию при посылке отчета об ошибке. Расширенную информацию можно получить при указании опции `--verbose`.

При "впрыске" пакета для него вставляется "заглушка" так, что система портежей считает его установленным. Это удобно если необходимо, например, установить двоичную версию XFree86 для нетривиальной видеокарты или просто добавить собственные пакеты. Необходимо указать категорию и конкретную версию вставляемого пакета. Например, `emerge inject sys-kernel/gentoo-sources-2.4.19`

--prune

-P

Предупреждение: Это действие может удалить важные пакеты! Удаляются все, кроме последних, версий соответствующих пакетов. Если для пакета есть несколько слотов, остается только последняя версия в каждом слоте. При этом не происходит проверки зависимостей, поэтому могут быть удалены пакеты, необходимые для нормальной работы системы. Используйте действие `clean`, если не уверены абсолютно в том, что делаете. В качестве аргументов можно указывать пакеты `ebuild`, классы или зависимости -- примеры см. в описании действия `clean`.

--regen

Вызывает проверку и обновление кеша зависимостей для всех пакетов `ebuild` в дереве портежей. Кеш используется для ускорения поиска и построения деревьев зависимостей. Эта команда не рекомендуется для пользователей `rsync`, поскольку `rsync` обновляет кеш, используя кеши на сервере. Если вы не знаете, чем отличается "пользователь `rsync`" от других пользователей, значит, вы - "пользователь `rsync`" :). Пользователи `rsync` должны просто выполнить команду `emerge sync` для регенерации кеша.

--search

-S

Ищет соответствия переданной строке в дереве портежей. Искомая строка представляет собой регулярное выражение. Например, команда

```
emerge search "^kde"
```

ищет любой пакет, имя которого начинается с "kde";

```
emerge search "gcc$"
```

ищет любой пакет, имя которого заканчивается на "gcc";

```
emerge search "office"
```

ищет любой пакет, имя которого содержит слово "office".

Если поиск необходимо выполнять и в описании пакета, используйте опцию `--searchdesc` или `-S`.

--sync

Иницииирует обновление дерева портежей с одного из зеркал rsync.gentoo.org. Учтите, что любые изменения, выполненные в дереве портежей, при этом будут стерты. За исключением специальных случаев, для обновления используется `rsync`. Чтобы узнать, как избежать удаления изменения, см. описание `PORTDIR_OVERLAY` на странице справочного руководства `make.conf(5)`.

--unmerge

-C

Предупреждение: При этом могут быть удалены важные пакеты! Удаляются все соответствующие пакеты. Проверка зависимостей не выполняется, поэтому могут быть удалены пакеты, необходимые для нормальной работы системы. В качестве аргументов можно задавать пакеты `ebuild`, классы или зависимости - примеры см. выше в описании `clean`.

--ask

-a

--buildpkg

-b

Говорит `emerge` о необходимости построить двоичные пакеты для всех соответствующих пакетов `ebuild`, помимо их обычного добавления в дерево. Полезно для тех, кто занимается поддержкой пакетов или при администрировании нескольких систем Gentoo Linux (строим один раз, устанавливаем пакеты `tbz2` с помощью `emerge` везде). Пакет будет создан в каталоге `${PKGDIR}/All`. В качестве альтернативы для уже включенных в дерево пакетов можно использовать опцию `quickpkg`, которая создает пакеты `tbz2` из "живой" файловой системы.

--buildpkgonly

-B

Создает двоичные пакеты для всех обработанных пакетов `ebuild`, не включая их фактически в дерево пакетов. Проблема только в том, что все зависимости времени построения уже должны быть включены в систему с помощью `emerge`.

--changelog

-l

Используйте эту опцию вместе с действием `--pretend`. В результате, будут показаны записи `ChangeLog` для всех пакетов, которые будут обновляться.

--columns

При использовании совместно с `--pretend` выдает имена пакетов, новую и старую версию выравненными по столбцам для простоты вырезания и копирования.

--debug**-d**

Требует от `emerge` работать в отладочном режиме. В этом режиме среда построения `bash` будет работать с опцией `-x`, выдавая детальную отладочную информацию в стандартный выходной поток. Опция `--debug` отлично подходит для поиска синтаксических ошибок `bash`.

--deep**-D**

При использовании совместно с опцией `--update`, этот флаг требует от `emerge` учитывать все дерево зависимостей пакетов, а не просто проверять непосредственные зависимости пакетов. Это, например, позволяет учесть обновления в библиотеках, не перечисленных явно в зависимостях пакета.

--emptytree**-e**

Создает видимость того, что дерево установленных пакетов содержит только `glibc`; эту опцию стоит использовать совместно с `--pretend`. Она позволяет разработчикам получить полное представление о дереве всех зависимостей пакета и перестраивать деревья в целом с использованием новейших версий библиотек.

--fetchonly**-f**

Вместо создания(установки) пакета, просто выбирает(скачивает, загружает) (`fetches`) в `../distfiles` все пакеты (главный пакет и все его зависимости).

--fetch-all-uri**-F**

--getbinpkg

-g

--getbinpkg

-G

--help

-h

--newuse

-N

используя совместно с опцией --update, позволяет проверить(и переустановить) все пакеты, флаги которых на данный момент были изменены в /etc/make.conf

--noconfmem

Требует от портежей не учитывать записи слияния, показывающий, что файл конфигурации в каталоге CONFIG_PROTECT уже был включен в дерево. Система портежей обычно будет включать такие файлы только раз, чтобы не заставлять пользователя возиться с одной и той же конфигурацией несколько раз. Этот флаг приведет к включению файла конфигурации каждый раз.

--nodeps

-O

Включает указанные пакеты, не включая зависимости. Учтите, что процесс построения может закончиться неудачно, если зависимости окажутся не удовлетворенными.

--noreplace

-n

Пропускает указанные в командной строке пакеты, если они уже были установлены. Без этой опции любые указанные в командной строке пакеты, пакеты ebuild или зависимости будут приводить к повторному включению соответствующего пакета системой Portage, даже если он уже установлен. Учтите, что зависимости система Portage по умолчанию повторно не включает.

--nospinner

Отключает для сеанса "крутилку" (spinner). "Крутилка" активируется, если команда

подана с терминала (TTY). Этот флаг отключает ее, независимо от устройства.

--oneshot

Включает пакеты как обычно, но не добавляет в профиль world с целью дальнейшего обновления.

--onlydeps

-o

Включает (или делает вид, что включает) зависимости указанных пакетов, а не сами пакеты.

--pretend

-p

Вместо реального включения просто показывает, что должно устанавливаться, если не указать --pretend. Использование опции --pretend настоятельно рекомендуется перед установкой любого незнакомого пакета. В выдаваемом результате:

N = новый

U = обновление

R = замена

B = блокируется уже установленным пакетом.

--quiet

-q

Результаты могут отличаться, но, в общем случае, объем выдаваемых системой портежей результатов сокращается или выдача уплотняется.

--resume

Возобновляет последнее действие по включению пакетов. Учтите, пожалуйста, что возобновить можно только действие, закончившееся с ошибкой из-за сбоя. Если делать нечего, будет просто выдано сообщение об успешном выполнении.

--searchdesc

-S

Сопоставляет строку поиска с полем описания и именем пакета. Учтите, что сопоставление с регулярным выражением ведется и в описании пакета.

--skipfirst

Это действие допустимо только совместно с `--resume`. Оно удаляет первый пакет в списке кандидатов на возобновление установки, так, чтобы включение могло продолжиться даже при наличии неисправимой или несущественной ошибки. Эту опцию надо использовать только в случаях, когда пропуск пакета не нарушает зависимости.

--tree

-t

--update

-u

Обновляет пакеты до самой новой из доступных версий. Учтите, что `--update` еще не обладает всеми необходимыми возможностями. Он не будет автоматически обновлять зависимости пакетов в файле `world`, если их самих там еще нет.

--upgradeonly

-U

Обновляет пакеты, исключая обновления, приводящие к понижению версии устанавливаемого пакета. Слоты (SLOTs) учитываются на простейшем уровне.

--usepkg

-k

Требует от `emerge` использовать двоичные пакеты (из каталога `$PKGDIR`), если они есть, избегая этим, по возможности, продолжительных компиляций. Эта опция полезна для установок с компакт-дисков; можно выполнить `export PKGDIR=/mnt/cdrom/packages`, а затем использовать эту опцию, чтобы `emerge` "вытягивала" пакеты с CD для удовлетворения зависимостей.

--usepkgonly

-K

Работает так же, как и `--usepkg`, но включает только двоичные пакеты. Все необходимые двоичные пакеты должны быть доступны на момент проверки

зависимостей или emerge просто прекратит работу.

--verbose

-v

Требует от emerge работать в многословном (verbose) режиме. В настоящее время этот флаг приводит к тому, что emerge выдает сообщения об ошибках GNU info, если они будут.

--version

-V

Выдает номер версии emerge. Эту опцию нельзя сочетать с другими; имя и формат действия при этом не важны.

Выдаваемые результаты

При использовании emerge с флагами pretend/verbose, результаты на первый взгляд могут показаться сложными для понимания. Этот раздел объясняет, что означает вся эта "стенографическая" информация.

[blocks B]

[blocks B] app-text/dos2unix (from pkg app-text/hd2u-0.8.0)

Пакет Dos2unix блокирует включение в дерево пакета hd2u. Блокирование выявляется, когда два пакета перекрываются по файлам или могут вызвать те или иные повреждения в системе. Однако обычно включать одновременно блокирующие друг друга пакеты не нужно, поскольку обычно они обеспечивают одинаковые возможности.

[ebuild N]

[ebuild N] app-games/qstat-25c

Пакет Qstat - новый в вашей системе и будет включаться в дерево с помощью emerge впервые.

[ebuild R]

[ebuild R] sys-apps/sed-4.0.5

Пакет sed 4.0.5 уже был включен, но если выполнить команду еще раз, система портежей повторно включит этот пакет (в данном случае, sed).

[ebuild F]

```
[ebuild F ] media-video/realplayer-8-r6
```

Пакет `realplayer` требует, чтобы вы загрузили исходники вручную. При попытке включить пакет в дерево, если исходные тексты пакета не найдены, система портежей остановится и выдаст инструкции о том, как загрузить недостающие файлы.

```
[ebuild U ]
```

```
[ebuild U ] net-fs/samba-2.2.8_pre1 [2.2.7a]
```

Пакет `Samba 2.2.7a` уже был включен и может быть обновлен до версии `2.2.8_pre1`.

```
[ebuild UD]
```

```
[ebuild UD] media-libs/libgd-1.8.4 [2.0.11]
```

Пакет `Libgd 2.0.11` уже был включен, но если выполнить команду, система портежей автоматически понизит (Downgrade) версию пакета до `1.8.4`. Это может произойти, если более новая версия пакета была замаскирована, поскольку оказалась поврежденной или создает угрозу защите системы, а исправление еще не выпущено.

Еще это может произойти потому, что пакет, который вы пытаетесь включить с помощью `emerge` требует для успешного включения более старую версию пакета. В данном случае, библиотека `libgd 2.x` несовместима с `libgd 1.x`. Это означает, что пакеты, созданные с использованием `libgd 1.x`, не скомпилируются с версией `2.x`, и ее надо понизить, прежде чем удастся эти пакеты успешно установить.

```
[ebuild U-]
```

```
[ebuild U-] x11-base/xfree-4.3.0 [4.2.99.902]
```

Дефис (-) представляет отсутствие информации о слоте (SLOT information) пакета `Xfree`. Это происходит, когда предыдущая версия была просто "впрыснута" (подробнее см. описание действия `inject`, или оказалась настолько старой, что соответствующий слот не существует. В любом случае, более новая версия `Xfree` доступна для обновления.

```
[ebuild U ]
```

```
[ebuild U ] net-analyzer/nmap-3.15_beta2 [3.15_beta1] -gtk -gnome
```

Значения `-gtk` и `-gnome` отражают статус USE-переменных при включении пакета `nmap`. Здесь мы видим, что `nmap` может использовать USE-переменные `gtk` и `gnome`, но в текущих установках переменные `gtk` и `gnome` отключены. Поэтому дополнительная поддержка `gtk` и `gnome` не будет включена в `nmap` при его

установке.

Примечание

Статус USE выдается только если используются опции `--pretend` и `--verbose`.

ПРИМЕЧАНИЯ

Практически всегда надо перед любой попыткой установки или обновления выполнять ее же, но с опцией `--pretend`. Это позволяет увидеть, что будет сделано, а также выявить блокирующие пакеты, которые надо будет предварительно удалить. Это тем более важно для классов `system` и `world`, которые могут вызвать обновление большого количества пакетов, если дерево портежей было достаточно активно.

Обычно также имеет смысл использовать опцию `--update`, которая позволяет игнорировать уже полностью обновленные пакеты, но обновлять при этом все остальные. При установке пакета с неустановленными зависимостями, если зависимости эти не указаны явно в списке параметров, они не будут добавляться в файл `world`. Если необходимо внести зависимости в файл `world` для дальнейшего обновления, не забудьте явно указать их в качестве параметров `emerge`.

В командной строке можно указывать USE-переменные, переопределяющие значения, указанные в стандартных местах, что позволяет избежать использования ряда нежелательных зависимостей. Флаги USE, заданные в командной строке, НЕ запоминаются. Например, команда

```
USE="-x -gnome" emerge mc
```

установит пакет `mc` с указанными установками USE.

Если команда `emerge --update system` или

`emerge --update world` не сработает с выдачей сообщений об ошибках, причина может быть в том, что пакет `ebuild` использует новую возможность, которой нет в этой версии утилиты `emerge`. Можно сначала выполнить

`emerge --update portage` для обновления до последней версии, которая должна поддерживать любые необходимые новые возможности.

Примечание: Будьте осторожны при использовании пакетов, находящихся в стадии разработки. Проблемы и ошибки, возникающие в результате неправильного использования замаскированных пакетов, приводят к напрасной

трате времени разработчиков Gentoo. Подумайте заранее, сможете ли справиться с проблемами, которые могут возникнуть.

Маски в системе портежей обеспечивают три основные функции: они позволяют задать период тестирования, в течение которого пакеты могут использоваться на машинах; они предотвращают использование пакета, который начал сбоить, и позволяют маскировать существующие пакеты, сломанные или угрожающие защите. Маскирование можно делать двумя способами: с помощью файла `package.mask` и с помощью ключевых слов. Как демаскировать пакет при использовании этих способов описано ниже. Учтите также, что если утилита `emerge` передается пакет `ebuild`, то все виды маскировки игнорируются и `emerge` будет пытаться включить пакет в систему.

package.mask

Файл `package.mask`, в основном, блокирует использование пакетов, вызывающих проблемы или известных, как приводящие к проблемам на некоторых машинах. Он находится в каталоге `/etc/portage`. Пакеты можно демаскировать в файле путем добавления символа комментария (`#`) в начале строки, маскирующей пакет.

Ключевые слова

Ключевые слова (KEYWORDS) также используются для маскировки пакетов, находящихся на стадии тестирования. Есть определяющие архитектуру ключевые слова для каждого пакета, позволяющие системе портежей определить, с какими системами совместим пакет. Пакеты, компилирующиеся на некоторой архитектуре, но не считающиеся "стабильными", маскируются путем указания тильды (`~`) перед названием архитектуры. Утилита `emerge` проверяет значение переменной среды `ACCEPT_KEYWORDS`, прежде чем разрешить или запретить включение пакета, замаскированного с помощью КЛЮЧЕВЫХ СЛОВ.

Чтобы проинформировать утилиту `emerge`, что необходимо построить эти "тестовые" версии пакетов, можно установить переменной `ACCEPT_KEYWORDS` значение `"~arch"`, где `arch` - одно из значений: `x86`, `ppc`, `sparc`, `mips`, `alpha`, `arm`, `hppa`. Например, команда `ACCEPT_KEYWORDS="~x86" emerge xfree` приведет к тому, что `emerge` будет учитывать замаскированные ключевыми словами версии XFree в качестве потенциальных кандидатов на обновление.

Предупреждение: Не устанавливайте этой переменной значение, отличающееся от соответствующего вашей архитектуре.

Как сообщать об ошибках

Пожалуйста, сообщайте о любых обнаруженных ошибках через bugs.gentoo.org web-сайт

Пожалуйста, включайте в отчет об ошибке результат, который выдает утилита

emerge.

Ссылки

emerge --help, ebuild(1), ebuild(5), make.conf(5)

Ряд вспомогательных приложений находится в каталоге `/usr/lib/portage/bin`. Пакет `app-admin/gentoolkit` содержит полезные сценарии, например, `qpkg` (средство запросов к пакетам).

Файлы

`/var/cache/edb/world` (`/var/lib/portage/world` в версиях `>=portage-2.0.51`)

Содержит список всех указанных пользователем пакетов. Можно без опаски редактировать этот файл, добавляя пакеты, которые необходимо учитывать при обновлении класса `world`, и удаляя те, которые обновлять при этом не нужно.

`/etc/make.conf`

Содержит переменные для процесса построения пакетов, переопределяя значения, заданные в `make.globals`. Именно этот файл надо редактировать вместо перечисленных далее.

`/etc/make.globals`

Содержит стандартные переменные для процесса построения. Не редактируйте этот файл.

`/etc/portage/mirrorsCode: # cat /etc/portage/mirrors`

local `http://192.168.0.10` `http://192.168.0.1:8080`

`http://distfiles.gentoo.org` `http://distro.ibiblio.org/pub/linux/distributions/gentoo`

Содержит список зеркал, на которых будет производиться поиск пакетов, запрошенных `emerge`. Стоит отметить, что данный список имеет приоритет выше, чем `GENTOO_MIRRORS` в файле `/etc/make.conf`, следовательно, данный список будет обрабатываться первым, причем зеркала, прописанные под словом `local`, будут обрабатываться самыми первыми.

`/etc/portage/package.keywordsCode: # cat /etc/portage/package.keywords`

`x11-misc/xcompmgr ~x86`

MAN (справки по командам)

x11-misc/transset ~x86
app-portage/getdelta ~x86
app-portage/deltup ~x86
dev-util/bdelta ~x86
net-im/jabberd ~x86
net-wireless/wifiscanner ~x86
net-wireless/wlassistant ~x86

Позволяет установить ACCESS_KEYWORDS для конкретных пакетов. Данный файл обрабатывается позже чем /etc/make.conf, следовательно его параметры будут актуальнее...

/etc/portage/package.useCode: # cat /etc/portage/package.use

x11-wm/fluxbox kde
app-misc/mc -X

Позволяет установить USE флаги для конкретных пакетов. Данный файл обрабатывается позже чем /etc/make.conf, следовательно его параметры будут актуальнее...

/etc/make.profile/package

Содержит список пакетов, используемых для базовой системы. Классы system и world учитывают содержимое этого файла. Не редактируйте его.

/etc/make.profile/make.defaults

Содержит специфические для профиля переменные для процесса построения пакетов. Не редактируйте этот файл.

/etc/make.profile/use.defaults

Содержит список пакетов, которые, в случае установки, приводят к установке по умолчанию соответствующего флага использования (use flag). Не редактируйте этот файл.

/etc/make.profile/virtuals

Содержит список стандартных пакетов, используемых для разрешения виртуальных зависимостей. Не редактируйте этот файл.

/var/cache/edb/virtuals

Содержит список пакетов, используемых для разрешения виртуальных зависимостей. В случае, когда поиск виртуальных зависимостей заканчивается неудачно, можно переупорядочить записи соответствующих виртуальных зависимостей в этом файле. Первый пакет, указанный после `virtual`, используется для сопоставления с `virtual`.

/usr/portage/profiles/use.desc

Содержит основной список флагов USE с описаниями их функций. Не редактируйте этот файл.

Примеры конфигурационных файлов

Автор: [Евгений Батогов а.к.а JohnBat26](#)

С версии: 1.0

Обновлено: 1.5 (30.03.08)

Мой компьютер:

Ноутбук DELL Inspiron 9400:

- Intel Core 2 Duo 2 Ghz.
- RAM: 2 gb DDR2.
- VIDEO: Nvidia Geforce 7900 GS 256 Mb DDR3
- HDD: 120 Gb.
- Max. Res. 1920X1200 pixels
- FS: Only XFS !

Программное обеспечение:

- Gentoo Linux.
- Kernel: 2.6.24
- GCC: 4.2.3
- Glibc: 2.7
- X.org: 1.4.0.90-r2
- KDE 3.5.9
- ...

1. Мой make.conf

Важно: в данном файле я использую оптимизацию: O3. Этот вариант не гарантирует надежной работы. Поэтому используйте O2, когда не хотите рисковать.

```
# These settings were set by the catalyst build script that automatically built this stage
# Please consult /etc/make.conf.example for a more detailed example
CFLAGS="-O3 -march=native -mtune=native -msse3 -mfpmath=sse -pipe -fomit-frame-pointer"
CHOST="x86_64-pc-linux-gnu"
CXXFLAGS="${CFLAGS}"
MAKEOPTS="-j3"
USE="64bit 7zip X a52 aac aalib accessibility acl acpi aim alsa amarok amazon
    amuled apm archive arts asf async audiofile automount avi bash-completion
    bcmath berkdb big-iron binary-drivers bitmap-fonts bluetooth branding
    bzip2 c++ cairo caps capture cardbus ccache cdda cddb cdinstall cdio
    cdparanoia cdr cdrom cdsound chm chroot cisco clamav clamd colordiff
    compress connectionstatus contactnotes cpio cpudetection cpulimit
    cracklib crypt cscope ctype cups curl curlwrappers custom-cflags cviewer
    cxx cyrillic darcs dba dbus dell depth32 designer-plugin device-mapper
    devil dhcp dia dialup dict directfb divx divx4linux djvu dri dv dvb dvd
    dvdr dvdread dvi dxr3 editor effects emerald emovix enca encode exif
```

```

expat extensions fat fbcon fbsplash festival ffmpeg fftw firefox flac
flash font-server fontconfig foomaticdb fortran freeimage ftp gadu games
gd gdb gdbm gif gimp gimpprint ginac glitz gmail gnuplot gpac gpg gphoto2
gpm graphics graphviz groupwise grub gtk gtk2 hal hbc i hddtemp highlight
history htmlhandbook iconv icq id3tag ieee1394 ifp imagemagick imap imlib
inkjar interbase ipq-mon iptv ipv6 ipw3945 ipw4965 irc jabber java java6
javascript jbig jce jpeg jpeg2k kcal kde kdecards kdenlive kdenlivefinal
kdehiddenvsibility kdepim kdexdeltat kdgraphics kdm kdrive kernel_linux
kig-scripting kipi kontakt krb4 lame laptop latex lcd lcmsjpeg ldap
ldapsam lha libcac libclamav libg++ libgcrypt libnotify libvisual libwww
live lm_sensors logitech-mouse logrotate magic maps matroska md5sum
memlimit mikmod mime mmx mng modplug mozbranding mozdevelop mozilla mp2
mp3 mp4 mp4live mpeg mpeg2 mplayer mplayer-bin musepack musicbrains mysql
ncurses nemesi netmeeting networkmanager nfs nls nokia6600 nomotif
nowlistening nptl nptlonly nsplugin ntfs numeric nvidia obex objc objc++
objc-gc octave office ofx ogg oggvorbis oodict openexr opengl openssl
openvpn oscar overlays pam pcpntl pcre pda pdf pdflib perl pie pink plasma
player pmount pmu png portage posix postgres povray ppp pptp pulseaudio
pymol python qq qt qt-copy qt-static qt3 qt3support qt4 query-browser
quicktime rar rc5 rdesktop readline real realmedia rss rtc ruby samba
sametime sasl savepwd sbig scanner scanfolder scanner sdl search sensor
session sharedext sharedmem simplexml skins slang slp smp sms smtp
snapshot sndfile soap sockets solver sound sounds spamassassin speech
speedo speex spell spexx sql sqlite sqlite3 sse sse2 ssl statistics stats
stream subtitles subversion svg svnserve symlink sysfs tcl tcltk tcpd
texteffect tga theora thesaurus threads tiff translator transparency
trayicon truetype tunepimp type1 udev unicode upnp usb utempter utf8 vcd
videos vim vim-pager vim-syntax vim-with-x visualization vnc vncviewer
vorbis vorbis-psy weather webpresence widescreen wifi winpopup wireshark
wmf wmp wxgtk1 wxwindows xcomposite xforms xfs xine xinerama xml xml2 xmp
xorg xosd xpm xprint xrender xskatcards xsl xv xvid yahoo zip zlib -daap
-privacy -unzip"
ACCEPT_KEYWORDS="~amd64 ~x86 amd64 x86"
AUTOCLEAN="yes"
PORTAGE_NICENESS=39
#FEATURES="ccache candy userfetch buildpkg -sandbox collision-protect"
FEATURES="ccache candy userfetch buildpkg -sandbox"
CCACHE_DIR="/var/tmp/ccache/"
CCACHE_SIZE="4G"
CC="gcc"
CXX="g++"
#PORTDIR_OVERLAY="/usr/portage/local/alsa /usr/portage/local/infra /usr/portage/local/qstardict
#/usr/portage/local/stardict /usr/portage/local/layman /usr/portage/local/google /usr/portage/local/layman/
#sunrise /usr/portage/local/layman/xeffects /usr/portage/local/krusader"

GENTOO_MIRRORS="http://trumpetti.atm.tut.fi/gentoo ftp://ftp.ussg.iu.edu/pub/linux/gentoo
ftp://ftp.ucsb.edu/pub/mirrors/linux/gentoo/ http://ftp.ucsb.edu/pub/mirrors/linux/gentoo/
http://gentoo.chem.wisc.edu/gentoo/ ftp://gentoo.mirrors.pair.com/ http://gentoo.mirrors.tds.net/gentoo/
ftp://gentoo.mirrors.tds.net/gentoo/ http://gentoo.netnitco.net/ ftp://gentoo.netnitco.net/pub/mirrors/gentoo/

```

```
source/ http://mirror.espri.arizona.edu/gentoo/
#http://mirrors.acm.cs.rpi.edu/gentoo/ ftp://ftp.ndlug.nd.edu/pub/gentoo/ http://open-
systems.ufl.edu/mirrors/gentoo #http://gentoo.llarian.net/ ftp://gentoo.llarian.net/pub/gentoo
#http://gentoo.binarycompass.org #http://mirror.datapipe.net/gentoo ftp://mirror.datapipe.net/gentoo http://
prometheus.cs.wmich.edu/gentoo #http://modzer0.cs.uaf.edu/public/gentoo/
#http://mirror.usu.edu/mirrors/gentoo/ ftp://mirror.usu.edu/mirrors/gentoo/ #ftp://lug.mtu.edu/gentoo http://
mirror.phy.olemiss.edu/mirror/gentoo http://mirror.mcs.anl.gov/pub/gentoo/
#ftp://mirror.mcs.anl.gov/pub/gentoo/ http://gentoo.mirrors.easynews.com/linux/gentoo/
#http://gentoo.cites.uiuc.edu/pub/gentoo/ ftp://gentoo.cites.uiuc.edu/pub/gentoo/
#ftp://ftp.www.edu/pub/mirrors/ftp.gentoo.org http://gentoo.localhost.net.ar/
#ftp://mirrors.localhost.net.ar/pub/mirrors/gentoo http://www.las.ic.unicamp.br/pub/gentoo/
#ftp://ftp.las.ic.unicamp.br/pub/gentoo/ http://gentoo.inode.at/ ftp://gentoo.inode.at/source/
#http://gd.tuwien.ac.at/opsys/linux/gentoo/ ftp://gd.tuwien.ac.at/opsys/linux/gentoo/
#http://ftp.belnet.be/mirror/rsync.gentoo.org/gentoo/ ftp://ftp.belnet.be/mirror/rsync.gentoo.org/gentoo/
#http://mirror.bih.net.ba/gentoo/ ftp://mirror.bih.net.ba/gentoo/
#ftp://ftp.sh.cvut.cz/MIRRORS/gentoo/gentoo #http://gentoo.supp.name/ http://mirror.uni-c.dk/pub/gentoo/
http://ftp.linux.ee/pub/gentoo/distfiles/ #ftp://ftp.linux.ee/pub/gentoo/distfiles/
#http://trumpetti.atm.tut.fi/gentoo/ ftp://trumpetti.atm.tut.fi/gentoo/ #http://ftp.public.fix.fi/gentoo/
ftp://ftp.public.fix.fi/gentoo http://gentoo.modulix.net/gentoo/ #http://ftp.club-internet.fr/pub/mirrors/gentoo
ftp://gentoo.imj.fr/pub/gentoo/ #ftp://ftp.tu-clausthal.de/pub/linux/gentoo/ ftp://sunsite.informatik.rwth-
aachen.de/pub/Linux/gentoo #http://linux.rz.ruhr-uni-bochum.de/download/gentoo-mirror/ "
```

```
#FETCHCOMMAND="/usr/bin/getdelta.sh \${URI}"
SYNC="rsync://rsync.gentoo.org/gentoo-portage"
INPUT_DEVICES="keyboard mouse synaptics evdev"
VIDEO_CARDS="nv nvidia vesa"
LINGUAS="ru"
PORTDIR="/usr/portage"
ALSA_CARDS="hda-intel"
```

2. Файл *xorg.conf* для *NVIDIA Geforce*

Section "Module"

```
Load      "ddc" # ddc probing of monitor
Load      "dbe" # Double buffer extension
SubSection "extmod"
    Option "omit xfree86-dga" # don't initialize the DGA extension
EndSubSection
Load      "record"
Load      "xtrap"
Load      "type1"
Load      "freetype"
Load      "synaptics"
Load      "glx"
Load      "wfb"
```

EndSection

Section "Files"

```
FontPath  "/usr/share/fonts/misc:unscaled"
FontPath  "/usr/share/fonts/Type1/"
```

```

FontPath  "/usr/share/fonts/TTF/"
FontPath  "/usr/share/fonts/corefonts"
FontPath  "/usr/share/fonts/freefonts"
FontPath  "/usr/share/fonts/sharefonts"
FontPath  "/usr/share/fonts/terminus"
FontPath  "/usr/share/fonts/ttf-bitstream-vera"
FontPath  "/usr/share/fonts/unifont"
FontPath  "/usr/share/fonts/100dpi:unscaled"
FontPath  "/usr/share/fonts/75dpi:unscaled"
FontPath  "/usr/share/fonts/artwiz"
FontPath  "/usr/share/fonts/cyrillic"
EndSection

Section "ServerFlags"
    Option  "AllowMouseOpenFail" "true"
    Option  "SuspendTime"        "10"
    Option  "OffTime"            "15"
    Option  "AutoAddDevices"     "false"
EndSection

Section "InputDevice"
    Identifier "Keyboard1"
    Driver     "kbd"
    Option     "XkbRules"        "xorg"
    Option     "XkbModel"        "microsoftpro"
    Option     "XkbLayout"       "us,ru"
    Option     "XkbVariant"      ",winkeys"
    Option     "XkbOptions"      "grp:shift_toggle,grp_led:scroll"
    Option     "AutoRepeat"      "500 30"
EndSection

Section "InputDevice"
    Identifier "USBMouse"
    Driver     "mouse"
    Option     "Device"          "/dev/input/mice"
    Option     "Buttons"         "7"
    Option     "CorePointer"
    Option     "Protocol"        "auto"
    Option     "ZAxisMapping"     "4 5 6 7"
EndSection

Section "InputDevice"
    Driver     "synaptics"
    Identifier  "TouchPad"
    Option     "Device"          "/dev/input/mice"
    Option     "Protocol"        "auto-dev"
    Option     "LeftEdge"       "1900"
    Option     "RightEdge"      "5400"

```

Примеры конфигурационных файлов

```
Option "BottomEdge" "1800"
Option "TopEdge" "3900"
Option "FingerLow" "25"
Option "FingerHigh" "30"
Option "MaxTapTime" "180"
Option "MaxTapMove" "220"
Option "VertScrollDelta" "100"
Option "MinSpeed" "0.02"
Option "MaxSpeed" "0.18"
Option "AccelFactor" "0.0010"
Option "SendCoreEvents" "true"
Option "Repeater" "/dev/ps2mouse"
Option "SHMConfig" "true"
EndSection
Section "Monitor"
    Identifier "DellLFP"
    HorizSync 28.0 - 96.0
    VertRefresh 43.0 - 60.0
    Option "DPMS"
# DisplaySize 320 240
EndSection

Section "Device"
    # Some names might need to be changed hereafter:
    Identifier "NVIDIA GeForce 7900GS"
    Driver "nvidia"
    VendorName "nVidia Corporation"
    BoardName "GeForce 7900 GS"
    BusID "PCI:1:0:0"
    #Option "NoLogo" "1"
    Option "UseDisplayDevice" "DFP"
    #### 2D ACCELERATION
    #Option "RenderAccel" "false" # hardware Render acceleration
    #Option "BackinStore" "true" #[] prevent artifacts?
    #Option "NoRenderExtension" "true"
    ## some options
    Option "SWcursor" "false" #[]
    Option "Render" "true"
    Option "Composite" "true"
    #### 3D ACCELERATION
    Option "EnablePageFlip" "yes" #[] Improves performance
    Option "AGPFastWrite" "yes" #[]
    Option "AGPMode" "4" # Supports AGP 4x
    # VideoRam 262144
    # Insert Clocks lines here if appropriate
    #Option "IgnoreDisplayDevices" "CRT, TV"
    Option "TripleBuffer" "True"
```

```

Option "AddARGBGLXVisuals" "True"
Option "AllowGLXWithComposite" "true"
Option "Coolbits" "1"
EndSection

Section "Screen"
    Identifier "Screen1"
    Device "NVIDIA GeForce 7900GS"
    Monitor "DellLFP"
    DefaultDepth 24
    SubSection "Display"
        Viewport 0 0
        Depth 8
        Modes "1920x1200" "1400x1050" "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Viewport 0 0
        Depth 16
        Modes "1920x1200" "1400x1050" "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    SubSection "Display"
        Viewport 0 0
        Depth 24
        Modes "1920x1200" "1400x1050" "1280x1024" "1024x768" "800x600" "640x480"
    EndSubSection
    Option "RenderAccel" "true"
    Option "AddARGBGLXVisuals" "True"
    Option "DisableGLXRootClipping" "True"
EndSection

Section "ServerLayout"
    Identifier "Flat Panel Only"
    Screen "Screen 1"
    #InputDevice "Touchpad" "AlwaysCore"
    InputDevice "TouchPad"
    InputDevice "USBMouse" "CorePointer"
EndSection
Section "Extensions"
    Option "Composite" "enable"
EndSection

```

3. Файл *xorg.conf* для ATI radeon:

```

Section "ServerLayout"
    Identifier "X.org Configured"
    Screen 0 "Screen0" 0 0
    InputDevice "Synaptics" "CorePointer"

```

Примеры конфигурационных файлов

```
    InputDevice  "Mouse"    "SendCoreEvents"
    InputDevice  "Keyboard0" "CoreKeyboard"
    Option       "OffTime"  "3"      # 3 indicates number of min until monitor-off
EndSection
```

```
Section "ServerFlags"
    Option      "AllowMouseOpenFail"
EndSection
```

```
Section "dri"
# Access to OpenGL ICD is allowed for all users:
#  Mode 0666
# Access to OpenGL ICD is restricted to a specific user group:
    Group 27 # video
    Mode 0660
EndSection
```

```
Section "Files"
    FontPath "/usr/share/fonts/misc:unscaled"
    FontPath "/usr/share/fonts/Type1/"
    FontPath "/usr/share/fonts/TTF/"
    FontPath "/usr/share/fonts/corefonts"
    FontPath "/usr/share/fonts/freefonts"
    FontPath "/usr/share/fonts/sharefonts"
    FontPath "/usr/share/fonts/terminus"
    FontPath "/usr/share/fonts/ttf-bitstream-vera"
    FontPath "/usr/share/fonts/unifont"
    FontPath "/usr/share/fonts/100dpi:unscaled"
    FontPath "/usr/share/fonts/75dpi:unscaled"
    FontPath "/usr/share/fonts/artwiz"
    FontPath "/usr/share/fonts/cyrillic"
EndSection
```

```
Section "Module"
    Load "glx"
    # This loads the miscellaneous extensions module, and disables
    # initialisation of the XFree86-DGA extension within that module.
    SubSection "extmod"
        Option "omit xfree86-dga" # don't initialise the DGA extension
    EndSubSection
    Load "dbe"
    Load "dri"
    Load "xtrap"
    Load "record"
    Load "freetype"
    Load "type1"
EndSection
```



```

Section "InputDevice"
    Identifier "Keyboard0"
    Driver "kbd"

    Option      "XkbModel"      "aspire5020"
    Option      "XkbLayout"     "se"
    #Option     "XkbVariant"    "nodeadkeys"
EndSection

```

```

Section "InputDevice"
    Identifier "Synaptics"
    Driver     "synaptics"
    Option     "Device"      "/dev/psaux"
    Option     "Protocol"    "auto-dev"
    Option     "LeftEdge"    "1700"
    Option     "RightEdge"   "5300"
    Option     "TopEdge"     "1700"
    Option     "BottomEdge"  "4200"
    Option     "FingerLow"   "25"
    Option     "FingerHigh"  "30"
    Option     "MaxTapTime"  "180"
    Option     "MaxTapMove"  "220"
    Option     "VertScrollDelta" "100"
    Option     "MinSpeed"    "0.09"
    Option     "MaxSpeed"    "0.18"
    Option     "AccelFactor" "0.0015"
    Option     "SHMConfig"   "on"
# # Option   "Repeater"     "/dev/ps2mouse"
EndSection

```

```

Section "InputDevice"
    Identifier "Mouse"
    Driver     "mouse"
    Option     "Device"      "/dev/input/mice"
    Option     "Protocol"    "imps/2"
    Option     "ZAxisMapping" "4 5"
    Option     "Buttons"     "5"
EndSection

```

```

Section "Monitor"
    Identifier "Monitor0"
    VendorName "LPL"
    ModelName  "0"
    Option     "DPMS"
EndSection

```

```
# === ATI device section ===
```

```

Section "Device"
    Identifier "Card0"
    Driver      "fglrx"
    VendorName  "ATI Technologies Inc"
    BoardName   "ATI Mobility X600"
# #### generic DRI settings ####
# === disable PnP Monitor ===
    #Option          "NoDDC"
# === disable/enable XAA/DRI ===
    Option "no_accel"      "no"
    Option "no_dri"        "no"
# === misc DRI settings ===
    Option "mtrr"          "off" # disable DRI mtrr mapper, driver has its own code for mtrr
# #### FireGL DDX driver module specific settings ####
# === Screen Management ===
    Option "DesktopSetup"    "0x00000100"
    Option "MonitorLayout"   "NONE,LVDS"
    Option "IgnoreEDID"      "off"
    Option "HSync2"          "unspecified"
    Option "VRefresh2"       "unspecified"
    Option "ScreenOverlap"   "0"
# === TV-out Management ===
    Option "NoTV"            "yes"
    Option "TVStandard"      "NTSC-M"
    Option "TVHSizeAdj"      "0"
    Option "TVVSizeAdj"      "0"
    Option "TVHPosAdj"       "0"
    Option "TVVPosAdj"       "0"
    Option "TVHStartAdj"     "0"
    Option "TVColorAdj"      "0"
    Option "GammaCorrectionI" "0x00000000"
    Option "GammaCorrectionII" "0x00000000"
# === OpenGL specific profiles/settings ===
    Option "Capabilities"     "0x00000000"
# === Video Overlay for the Xv extension ===
    Option "VideoOverlay"     "on"
# === OpenGL Overlay ===
# Note: When OpenGL Overlay is enabled, Video Overlay
#       will be disabled automatically
    Option "OpenGLOverlay"    "off"
# === Center Mode (Laptops only) ===
    Option "CenterMode"       "off"
# === Pseudo Color Visuals (8-bit visuals) ===
    Option "PseudoColorVisuals" "off"
# === QBS Management ===
    Option "Stereo"           "off"
    Option "StereoSyncEnable" "1"

```

```
# === FSAAs Management ===
```

```
Option "FSAAScale"          "no"
Option "FSAAScale"          "1"
Option "FSAADisableGamma"   "no"
Option "FSAACustomizeMSPos"  "no"
Option "FSAAMSPosX0"         "0.000000"
Option "FSAAMSPosY0"         "0.000000"
Option "FSAAMSPosX1"         "0.000000"
Option "FSAAMSPosY1"         "0.000000"
Option "FSAAMSPosX2"         "0.000000"
Option "FSAAMSPosY2"         "0.000000"
Option "FSAAMSPosX3"         "0.000000"
Option "FSAAMSPosY3"         "0.000000"
Option "FSAAMSPosX4"         "0.000000"
Option "FSAAMSPosY4"         "0.000000"
Option "FSAAMSPosX5"         "0.000000"
Option "FSAAMSPosY5"         "0.000000"
```

```
# === Misc Options ===
```

```
Option "UseFastTLS"          "0"
Option "BlockSignalsOnLock"  "on"
Option "UseInternalAGPGART"  "no"
Option "ForceGenericCPU"     "no"
Option "DynamicClocks"        "on" # Use ATI Powerplay features
BusID "PCI:1:0:0" # vendor=1002, device=3150
Screen 0
```

```
EndSection
```

```
#Section "Device"
```

```
### Available Driver options are:-
```

```
### Values: <i>: integer, <f>: float, <bool>: "True"/"False",
```

```
### <string>: "String", <freq>: "<f> Hz/kHz/MHz"
```

```
### [arg]: arg optional
```

```
#Option "NoAccel"             # [<bool>]
#Option "SWcursor"            # [<bool>]
#Option "Dac6Bit"              # [<bool>]
#Option "Dac8Bit"              # [<bool>]
#Option "BusType"              # [<str>]
#Option "CPPIOMode"            # [<bool>]
#Option "CPusecTimeout"        # <i>
#Option "AGPMode"              # <i>
#Option "AGPFastWrite"         # [<bool>]
#Option "AGPSize"              # <i>
#Option "GARTSize"             # <i>
#Option "RingSize"             # <i>
#Option "BufferSize"           # <i>
#Option "EnableDepthMoves"     # [<bool>]
#Option "EnablePageFlip"       # [<bool>]
#Option "NoBackBuffer"         # [<bool>]
```

Примеры конфигурационных файлов

```

#Option "DRIRemote" # [<bool>]
#Option "PanelOff" # [<bool>]
#Option "DDCMode" # [<bool>]
#Option "MonitorLayout" # [<str>]
#Option "IgnoreEDID" # [<bool>]
#Option "UseFBDev" # [<bool>]
#Option "VideoKey" # <i>
#Option "MergedFB" # [<bool>]
#Option "CRT2HSync" # [<str>]
#Option "CRT2VRefresh" # [<str>]
#Option "CRT2Position" # [<str>]
#Option "MetaModes" # [<str>]
#Option "MergedDPI" # [<str>]
#Option "NoMergedXinerama" # [<bool>]
#Option "MergedXineramaCRT2IsScreen0" # [<bool>]
#Option "DisplayPriority" # [<str>]
#Option "PanelSize" # [<str>]
#Option "ForceMinDotClock" # <freq>
#Option "RenderAccel" # [<bool>]
#Option "SubPixelOrder" # [<str>]
#Option "ShowCache" # [<bool>]
#Option "DynamicClocks" # [<bool>]
#Option "VGAAccess" # [<bool>]
#Option "LVDSProbePLL" # [<bool>]
#Option "ReverseDDC" # [<bool>]
#Option "BIOSHotkeys" # [<bool>]
# Identifier "Card0"
# Driver "ati"
# VendorName "ATI Technologies Inc"
# BoardName "ATI Mobility X600"
# BusID "PCI:1:0:0"
#EndSection

Section "Screen"
    Identifier "Screen0"
    Device "Card0"
    Monitor "Monitor0"
    DefaultDepth 24

    SubSection "Display"
        Viewport 0 0
        Depth 16
#        Modes "1280x800"
    EndSubSection
    SubSection "Display"
        Viewport 0 0
        Depth 24
#        Modes "1280x800"

```

EndSubSection
EndSection

4. Файл [/etc/fstab](#)

Важные команды Gentoo Linux

1. ***emerge --ask --verbose*** (или ***emerge -av***) - просмотр USE-флагов перед началом компиляции.
2. ***emerge имя_пакета*** – установка пакета.
3. ***emerge sync*** – обновление локального дерева портежей.
4. ***echo "category/some_package some_flags" >> /etc/portage/package.use --*** прописывание USE-флагов в */etc/portage/package.use*, для этого выполните

Unix Toolbox

Ссылка на оригинал: <http://cb.vu/unixtoolbox.xhtml>

Revision: 11

С версии: 1.5

This document is a collection of Unix/Linux/BSD commands and tasks which are useful for IT work or for advanced users. This is a practical guide with concise explanations, however the reader is supposed to know what s/he is doing.

The latest version of this document can be found at <http://cb.vu/unixtoolbox.xhtml>.

Running kernel and system information

```
# uname -a # Get the kernel version (and BSD version)
# cat /etc/SuSE-release # Get SuSE version
# cat /etc/debian_version # Get Debian version
```

Use /etc/DISTR-release with DISTR= lsb (Ubuntu), redhat, gentoo, mandrake, sun (Solaris), and so on.

```
# uptime # Show how long the system has been running + load
# hostname # system's host name
# hostname -i # Display the IP address of the host.
# man hier # Description of the file system hierarchy
# last reboot # Show system reboot history
```

1.1 Hardware Informations

Kernel detected hardware

```
# dmesg # Detected hardware and boot messages
# lsdev # information about installed hardware
# dd if=/dev/mem bs=1k skip=768 count=256 2>/dev/null | strings -n 8 # Read BIOS
```

Linux

```
# cat /proc/cpuinfo # CPU model
# cat /proc/meminfo # Hardware memory
# grep MemTotal /proc/meminfo # Display the physical memory
# watch -n1 'cat /proc/interrupts' # Watch changeable interrupts continuously
# free -m # Used and free memory (-m for MB)
# cat /proc/devices # Configured devices
# lspci -tv # Show PCI devices
# lsusb -tv # Show USB devices
# lshal # Show a list of all devices with their properties
# dmidecode # Show DMI/SMBIOS: hw info from the BIOS
```

FreeBSD

```
# sysctl hw.model # CPU model
# sysctl hw # Gives a lot of hardware information
# sysctl vm # Memory usage
# dmesg | grep "real mem" # Hardware memory
# sysctl -a | grep mem # Kernel memory settings and info
# sysctl dev # Configured devices
# pciconf -l -cv # Show PCI devices
# usbdevs -v # Show USB devices
# atacontrol list # Show ATA devices
```

1.2 Load, statistics and messages

The following commands are useful to find out what is going on on the system.

```
# top # display and update the top cpu processes
# mpstat 1 # display processors related statistics
# vmstat 2 # display virtual memory statistics
# iostat 2 # display I/O statistics (2 s intervals)
# systat -vmstat 1 # BSD summary of system statistics (1 s intervals)
# systat -tcp 1 # BSD tcp connections (try also -ip)
# systat -netstat 1 # BSD active network connections
# systat -ifstat 1 # BSD network traffic through active interfaces
# systat -iostat 1 # BSD CPU and disk throughput
# tail -n 500 /var/log/messages # Last 500 kernel/syslog messages
# tail /var/log/warn # System warnings messages see syslog.conf
```

1.3 Users

```
# id # Show the active user id with login and group
# last # Show last logins on the system
# who # Show who is logged on the system
# groupadd admin # Add group "admin" and user colin (Linux/Solaris)
# useradd -c "Colin Barschel" -g admin -m colin
# userdel colin # Delete user colin (Linux/Solaris)
# adduser joe # FreeBSD add user joe (interactive)
# rmuser joe # FreeBSD delete user joe (interactive)
# pw groupadd admin # Use pw on FreeBSD
# pw groupmod admin -m newmember # Add a new member to a group
```

```
# pw useradd colin -c "Colin Barschel" -g admin -m -s /bin/tcsh
```

```
# pw userdel colin; pw groupdel admin
```

Encrypted passwords are stored in /etc/shadow for Linux and Solaris and /etc/master.passwd on FreeBSD. If the master.passwd is modified manually (say to delete a password), run #

```
pwd_mkdb -p master.passwd to rebuild the database.
```

To temporarily prevent logins system wide (for all users but root) use nologin. The message in nologin will be displayed.

```
# echo "Sorry no login now" > /etc/nologin # (Linux)
# echo "Sorry no login now" > /var/run/nologin # (FreeBSD)
```

1.4 Limits

Some application require higher limits on open files and sockets (like a proxy web server,database). The default limits are usually too low.

Linux

Per shell/script

The shell limits are governed by ulimit. The status is checked with ulimit -a. For example to change the open files limit from 1024 to 10240 do:

```
# ulimit -n 10240 # This is only valid within the shell
```

The ulimit command can be used in a script to change the limits for the script only.

Per user/process

Login users and applications can be configured in /etc/security/limits.conf. For example:

```
# cat /etc/security/limits.conf
```

```
* hard nproc 250 # Limit user processes
```

```
asterisk hard nofile 409600 # Limit application open files
```

System wide

Kernel limits are set with sysctl. Permanent limits are set in /etc/sysctl.conf.


```
# sysctl -a # View all system limits
# sysctl fs.file-max # View max open files limit
fs.file-max=102400 # Permanent entry in sysctl.conf
# cat /proc/sys/fs/file-nr # How many file descriptors are in use
```

FreeBSD

Per shell/script

Use the command `limits` in `cs`h or `tc`sh or as in Linux, use `ulimit` in an `sh` or `bash` shell.

Per user/process

The default limits on login are set in `/etc/login.conf`. An unlimited value is still limited by the system maximal value.

System wide

Kernel limits are also set with `sysctl`. Permanent limits are set in `/etc/sysctl.conf` or `/boot/`

`loader.conf`. The syntax is the same as Linux but the keys are different.

```
# sysctl -a # View all system limits
# sysctl kern.maxfiles=XXXX # maximum number of file descriptors
kern.ipc.nmbclusters=32768 # Permanent entry in /etc/sysctl.conf
kern.maxfiles=65536 # Typical values for Squid
```

kern.maxfilesperproc=32768

```
kern.ipc.somaxconn=8192 # TCP queue. Better for apache/sendmail
# sysctl kern.openfiles # How many file descriptors are in use
# sysctl kern.ipc.numopensockets # How many open sockets are in use
```

See The FreeBSD handbook Chapter 11₁ for details.

Solaris

The following values in `/etc/system` will increase the maximum file descriptors per proc:

```
set rlim_fd_max = 4096 # Hard limit on file descriptors for a single proc
set rlim_fd_cur = 1024 # Soft limit on file descriptors for a single proc
```

1.5 Runlevels

Linux

Once booted, the kernel starts `init` which then starts `rc` which starts all scripts belonging to a runlevel. The scripts are stored in `/etc/init.d` and are linked into `/etc/rc.d/rcN.d` with N the runlevel number.

The default runlevel is configured in `/etc/inittab`. It is usually 3 or 5:

grep default: `/etc/inittab`

id:3:initdefault:

The actual runlevel (the list is shown below) can be changed with `init`. For example to go from 3 to 5:

```
# init 5 # Enters runlevel 5
```

0 Shutdown and halt

1 Single-User mode (also S)

2 Multi-user without network

3 Multi-user with network

5 Multi-user with X

6 Reboot

Use `chkconfig` to configure the programs that will be started at boot in a runlevel.

```
# chkconfig --list # List all init scripts
# chkconfig --list sshd # Report the status of sshd
1.http://www.freebsd.org/handbook/configtuning-kernel-limits.html
# chkconfig sshd --level 35 on # Configure sshd for levels 3 and 5
# chkconfig sshd off # Disable sshd for all runlevels
```

Debian and Debian based distributions like Ubuntu or Knoppix use the command `update-rc.d`

to manage the runlevels scripts. Default is to start in 2,3,4 and 5 and shutdown in 0,1 and 6.

```
# update-rc.d sshd defaults # Activate sshd with the default runlevels
# update-rc.d sshd start 20 2 3 4 5 . stop 20 0 1 6 . # With explicit arguments
# update-rc.d -f sshd remove # Disable sshd for all runlevels
# shutdown -h now (or # poweroff) # Shutdown and halt the system
```

FreeBSD

The BSD boot approach is different from the SysV, there are no runlevels. The final boot state (single user, with or without X) is configured in `/etc/ttys`. All OS scripts are located in `/etc/rc.d/` and in `/usr/local/etc/rc.d/` for third-party applications. The activation of the service is configured in `/etc/rc.conf` and `/etc/rc.conf.local`. The default behavior is configured in `/etc/defaults/rc.conf`. The scripts responds at least to start|stop|status.

`# /etc/rc.d/sshd status`

sshd is running as pid 552.

```
# shutdown now # Go into single-user mode
# exit # Go back to multi-user mode
# shutdown -p now # Shutdown and halt the system
# shutdown -r now # Reboot
```

The process `init` can also be used to reach one of the following states level. For example #

`init 6` for reboot.

0 Halt and turn the power off (signal `USR2`)

1 Go to single-user mode (signal `TERM`)

6 Reboot the machine (signal `INT`)

c Block further logins (signal `TSTP`)

q Rescan the `ttys(5)` file (signal `HUP`)

1.6 Reset root password

Linux method 1

At the boot loader (lilo or grub), enter the following boot option:

`init=/bin/sh`

The kernel will mount the root partition and `init` will start the bourne shell instead of `rc` and then a runlevel. Use the command `passwd` at the prompt to change the password and then reboot. Forget the single user mode as you need the password for that.

If, after booting, the root partition is mounted read only, remount it rw:

`# mount -o remount,rw /`

```
# passwd # or delete the root password (/etc/shadow)
```

```
# sync; mount -o remount,ro / # sync before to remount read only
```

`# reboot`

FreeBSD and Linux method 2

FreeBSD won't let you go away with the simple `init` trick. The solution is to mount the root partition from an other OS (like a rescue CD) and change the password on the disk.

- Boot a live CD or installation CD into a rescue mode which will give you a shell.
- Find the root partition with `fdisk` e.g. `fdisk /dev/sda`
- Mount it and use `chroot`:

`# mount -o rw /dev/ad4s3a /mnt`

```
# chroot /mnt # chroot into /mnt
```

```
# passwd
```

```
# reboot
```

Alternatively on FreeBSD, boot in single user mode, remount / rw and use passwd.

```
# mount -u /; mount -a # will mount / rw
```

```
# passwd
```

```
# reboot
```

1.7 Kernel modules

Linux

```
# lsmod # List all modules loaded in the kernel
```

```
# modprobe isdn # To load a module (here isdn)
```

FreeBSD

```
# kldstat # List all modules loaded in the kernel
```

```
# kldload crypto # To load a module (here crypto)
```

1.8 Compile Kernel

Linux

```
# cd /usr/src/linux
```

```
# make mrproper # Clean everything, including config files
```

```
# make oldconfig # Create a new config file from the current kernel
```

```
# make menuconfig # or xconfig (Qt) or gconfig (GTK)
```

```
# make # Create a compressed kernel image
```

```
# make modules # Compile the modules
```

```
# make modules_install # Install the modules
```

```
# make install # Install the kernel
```

```
# reboot
```

FreeBSD

To modify and rebuild the kernel, copy the generic configuration file to a new name and edit it as needed. It is however also possible to edit the file `GENERIC` directly.

```
# cd /usr/src/sys/i386/conf/
```

```
# cp GENERIC MYKERNEL
```

```
# cd /usr/src
```

```
# make buildkernel KERNCONF=MYKERNEL
```

```
# make installkernel KERNCONF=MYKERNEL
```

To rebuild the full OS:

```
# make buildworld # Build the full OS but not the kernel
```

```
# make buildkernel # Use KERNCONF as above if appropriate
```

```
# make installkernel
```

```
# reboot
```

```
# mergemaster -p # Compares only files known to be essential
```

```
# make installworld
```

```
# mergemaster # Update all configuration and other files
```

```
# reboot
```

For small changes in the source, sometimes the short version is enough:

```
# make kernel world # Compile and install both kernel and OS
```

```
# mergemaster
```

```
# reboot
```

2. PROCESSES

2.1 Listing and PIDs

Each process has a unique number, the PID. A list of all running process is retrieved with `ps`.

```
# ps -auxefw # Extensive list of all running process
```

However more typical usage is with a pipe or with `pgrep`:

```
# ps axww | grep cron
```

```
586 ?? ls 0:01.48 /usr/sbin/cron -s
```

```
# pgrep -l sshd # Find the PIDs of processes by (part of) name
```

```
# fuser -va 22/tcp # List processes using port 22
```

```
# fuser -va /home # List processes accessing the /home partiton
```

```
# strace df # Trace system calls and signals
```

```
# truss df # same as above on FreeBSD/Solaris/Unixware
```

```
# history | tail -50 # Display the last 50 used commands
```

2.2 Priority

Change the priority of a running process with `renice`. Negative numbers have a higher priority, the lowest is -20 and "nice" have a positive value.

```
# renice -5 586 # Stronger priority
```

```
586: old priority 0, new priority -5
```

Start the process with a defined priority with `nice`. Positive is "nice" or weak, negative is strong scheduling priority. Make sure you know if `/usr/bin/nice` or the shell built-in is used (check with `# which nice`).

```
# nice -n -5 top # Stronger priority (/usr/bin/nice)
```

```
# nice -n 5 top # Weaker priority (/usr/bin/nice)
```

```
# nice +5 top # tcsh builtin nice (same as above!)
```

2.3 Background/Foreground

When started from a shell, processes can be brought in the background and back to the foreground with `[Ctrl]-[Z]` (^Z), `bg` and `fg`. For example start two processes, bring them in the background, list the processes with `jobs` and bring one in the foreground.

```
# ping cb.vu > ping.log
```

```
^Z # ping is suspended (stopped) with [Ctrl]-[Z]
```

```
# bg # put in background and continues running
```

```
# jobs -l # List processes in background
```

```
[1] - 36232 Running ping cb.vu > ping.log
```

```
[2] + 36233 Suspended (tty output) top
```

```
# fg %2 # Bring process 2 back in foreground
```

Use `nohup` to start a process which has to keep running when the shell is closed (immune to hangups).

```
# nohup ping -i 60 > ping.log &
```

2.4 Top

The program `top` displays running information of processes.

```
# top
```

While `top` is running press the key `h` for a help overview. Useful keys are:

- `u` [user name] To display only the processes belonging to the user. Use `+` or blank to see all users
- `k` [pid] Kill the process with pid.

- 1 To display all processors statistics (Linux only)
- R Toggle normal/reverse sort.

2.5 Signals/Kill

Terminate or send a signal with `kill` or `killall`.

```
# ping -i 60 cb.vu > ping.log &
```

```
[1] 4712
```

```
# kill -s TERM 4712 # same as kill -15 4712
# killall -1 httpd # Kill HUP processes by exact name
# pkill -9 http # Kill TERM processes by (part of) name
# pkill -TERM -u www # Kill TERM processes owned by www
# fuser -k -TERM -m /home # Kill every process accessing /home (to umount)
```

Important signals are:

- 1 HUP (hang up)
- 2 INT (interrupt)
- 3 QUIT (quit)
- 9 KILL (non-catchable, non-ignorable kill)
- 15 TERM (software termination signal)

3. FILE SYSTEM

3.1 Permissions

Change permission and ownership with `chmod` and `chown`. The default umask can be changed for all users in `/etc/profile` for Linux or `/etc/login.conf` for FreeBSD. The default umask is usually 022. The umask is subtracted from 777, thus umask 022 results in a permission of 755.

```
1 --x execute # Mode 764 = exec/read/write | read/write | read
2 -w- write # For: |-- Owner --| |- Group-| |Oth|
4 r-- read
ugo=a u=user, g=group, o=others, a=everyone
# chmod [OPTION] MODE[,MODE] FILE # MODE is of the form [ugo]*([-+]=([rwxXst]))
# chmod 640 /var/log/maillog # Restrict the log -rw-r-----
# chmod u=rw,g=r,o= /var/log/maillog # Same as above
# chmod -R o-r /home/* # Recursive remove other readable for all users
# chmod u+s /path/to/prog # Set SUID bit on executable (know what you do!)
# find / -perm -u+s -print # Find all programs with the SUID bit
# chown user:group /path/to/file # Change the user and group ownership of a file
# chgrp group /path/to/file # Change the group ownership of a file
```

3.2 Disk information

```
# diskinfo -v /dev/ad2 # information about disk (sector/size) FreeBSD
# hdparm -I /dev/sda # information about the IDE/ATA disk (Linux)
# fdisk /dev/ad2 # Display and manipulate the partition table
# smartctl -a /dev/ad2 # Display the disk SMART info
```

3.3 Boot

FreeBSD

To boot an old kernel if the new kernel doesn't boot, stop the boot at during the count down.

```
# unload
# load kernel.old
# boot
```

3.4 System mount points/Disk usage

```
# mount | column -t # Show mounted file-systems on the system
# df # display free disk space and mounted devices
# cat /proc/partitions # Show all registered partitions (Linux)
```

Disk usage

```
# du -sh * # Directory sizes as listing
# du -csh # Total directory size of the current directory
# du -ks * | sort -n -r # Sort everything by size in kilobytes
# ls -lSr # Show files, biggest last
```

3.5 Who has which files opened

This is useful to find out which file is blocking a partition which has to be unmounted and gives a typical error of:

```
# umount /home/
```

```
umount: unmount of /home # umount impossible because a file is locking home
failed: Device busy
```

FreeBSD and most Unixes

```
# fstat -f /home # for a mount point
# fstat -p PID # for an application with PID
# fstat -u user # for a user name
```

Find opened log file (or other opened files), say for Xorg:

```
# ps ax | grep Xorg | awk '{print $1}'
```

```
1252
```

```
# fstat -p 1252
```

```
USER CMD PID FD MOUNT INUM MODE SZ|DV R/W
```

```
root Xorg 1252 root / 2 drwxr-xr-x 512 r
```

```
root Xorg 1252 text /usr 216016 -rws--x--x 1679848 r
```

```
root Xorg 1252 0 /var 212042 -rw-r--r-- 56987 w
```

The file with inum 212042 is the only file in /var:

```
# find -x /var -inum 212042
```

```
/var/log/Xorg.0.log
```

Linux

Find opened files on a mount point with `fuser` or `lsof`:

```
# fuser -m /home # List processes accessing /home
```

```
# lsof /home
```

```
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME
```

```
tcsh 29029 eedcoba cwd DIR 0,18 12288 1048587 /home/eedcoba (guam:/home)
```

```
lsof 29140 eedcoba cwd DIR 0,18 12288 1048587 /home/eedcoba (guam:/home)
```

About an application:

```
ps ax | grep Xorg | awk '{print $1}'
```

```
3324
```

```
# lsof -p 3324
```

```
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME
```

```
Xorg 3324 root 0w REG 8,6 56296 12492 /var/log/Xorg.0.log
```

About a single file:

```
# lsof /var/log/Xorg.0.log
```

```
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME
Xorg 3324 root 0w REG 8,6 56296 12492 /var/log/Xorg.0.log
```

3.6 Mount/remount a file system

For example the cdrom. If listed in /etc/fstab:

```
# mount /cdrom
```

Or find the device in /dev/ or with dmesg

FreeBSD

```
# mount -v -t cd9660 /dev/cd0c /mnt # cdrom
# mount_cd9660 /dev/wcd0c /cdrom # other method
# mount -v -t msdos /dev/fd0c /mnt # floppy
```

Entry in /etc/fstab:

```
# Device Mountpoint FStype Options Dump Pass#
/dev/acd0 /cdrom cd9660 ro,noauto 0 0
```

To let users do it:

```
# sysctl vfs.usermount=1 # Or insert the line "vfs.usermount=1" in
/etc/sysctl.conf
```

Linux

```
# mount -t auto /dev/cdrom /mnt/cdrom # typical cdrom mount command
# mount /dev/hdc -t iso9660 -r /cdrom # typical IDE
# mount /dev/sdc0 -t iso9660 -r /cdrom # typical SCSI
```

Entry in /etc/fstab:

```
/dev/cdrom /media/cdrom subfs noauto,fs=cdfss,ro,procuid,nosuid,nodev,exec 0 0
```

Mount a FreeBSD partition with Linux

Find the partition number containing with fdisk, this is usually the root partition, but it could be an other BSD slice too. If the FreeBSD has many slices, they are the one not listed in the fdisk table, but visible in /dev/sda* or /dev/hda*.

```
# fdisk /dev/sda # Find the FreeBSD partition
/dev/sda3 * 5357 7905 20474842+ a5 FreeBSD
# mount -t ufs -o ufstype=ufs2,ro /dev/sda3 /mnt
/dev/sda10 = /tmp; /dev/sda11 /usr # The other slices
```

Remount

Remount a device without unmounting it. Necessary for fsck for example

```
# mount -o remount,ro / # Linux
# mount -o ro / # FreeBSD
```

Copy the raw data from a cdrom into an iso image:

```
# dd if=/dev/cd0c of=file.iso
```

3.7 Mount an SMB share

Suppose we want to access the SMB share myshare on the computer smbserver, the address as typed on a Windows PC is \\smbserver\myshare\. We mount on /mnt/smbshare. Warning> cifs wants an IP or DNS name, not a Windows name.

Linux

```
# smbclient -U user -I 192.168.16.229 -L //smbshare/ # List the shares
# mount -t smbfs -o username=winuser //smbserver/myshare /mnt/smbshare
# mount -t cifs -o username=winuser,password=winpwd //192.168.16.229/myshare
/mnt/share
```

Additionally with the package mount.cifs it is possible to store the credentials in a file, for

```
example /home/user/.smb:
username=winuser
```

password=winpwd

And mount as follow:

```
# mount -t cifs -o credentials=/home/user/.smb //192.168.16.229/myshare /mnt/smbshare
```

FreeBSD

Use -I to give the IP (or DNS name); smbserver is the Windows name.

```
# smbutil view -I 192.168.16.229 //winuser@smbserver # List the shares
```

```
# mount_smbfs -I 192.168.16.229 //winuser@smbserver/myshare /mnt/smbshare
```

3.8 Mount an image

Linux loop-back

```
# mount -t iso9660 -o loop file.iso /mnt # Mount a CD image
```

```
# mount -t ext3 -o loop file.img /mnt # Mount an image with ext3 fs
```

FreeBSD

With memory device (do # kldload md.ko if necessary):

```
# mdconfig -a -t vnode -f file.iso -u 0
```

```
# mount -t cd9660 /dev/md0 /mnt
```

```
# umount /mnt; mdconfig -d -u 0 # Cleanup the md device
```

Or with virtual node:

```
# vnconfig /dev/vn0c file.iso; mount -t cd9660 /dev/vn0c /mnt
```

```
# umount /mnt; vnconfig -u /dev/vn0c # Cleanup the vn device
```

Solaris and FreeBSD

with loop-back file interface or lofi:

```
# lofiadm -a file.iso
```

```
# mount -F hsfs -o ro /dev/lofi/1 /mnt
```

```
# umount /mnt; lofiadm -d /dev/lofi/1 # Cleanup the lofi device
```

3.9 Create and burn an ISO image

This will copy the cd or DVD sector for sector. Without `conv=notrunc`, the image will be smaller if there is less content on the cd. See below and the `dd` examples (page 38).

```
# dd if=/dev/hdc of=/tmp/mycd.iso bs=2048 conv=notrunc
```

Use `mkisofs` to create a CD/DVD image from files in a directory. To overcome the file names restrictions: `-r` enables the Rock Ridge extensions common to UNIX systems, `-J` enables Joliet extensions used by Microsoft systems. `-L` allows ISO9660 filenames to begin with a period.

```
# mkisofs -J -L -r -V TITLE -o imagefile.iso /path/to/dir
```

On FreeBSD, `mkisofs` is found in the ports in `sysutils/cdrtools`.

Burn a CD/DVD ISO image

FreeBSD

FreeBSD does not enable DMA on ATAPI drives by default. DMA is enabled with the `sysctl` command and the arguments below, or with `/boot/loader.conf` with the following entries:

```
hw.ata.ata_dma="1"
```

```
hw.ata.ataapi_dma="1"
```

Use `burncd` with an ATAPI device (`burncd` is part of the base system) and `cdrecord` (in `sysutils/cdrtools`) with a SCSI drive.

```
# burncd -f /dev/acd0 data imagefile.iso fixate # For ATAPI drive
```

```
# cdrecord -scanbus # To find the burner device (like 1,0,0)
```



```
# cdrecord dev=1,0,0 imagefile.iso
```

Linux

Also use `cdrecord` with Linux as described above. Additionally it is possible to use the native ATAPI interface which is found with:

```
# cdrecord dev=ATAPI -scanbus
```

And burn the CD/DVD as above.

Convert a Nero .nrg file to .iso

Nero simply adds a 300Kb header to a normal iso image. This can be trimmed with `dd`.

```
# dd bs=1k if=imagefile.nrg of=imagefile.iso skip=300
```

Convert a bin/cue image to .iso

The little `bchunk` program² can do this. It is in the FreeBSD ports in `sysutils/bchunk`.

```
# bchunk imagefile.bin imagefile.cue imagefile.iso
```

3.10 Create a file based image

For example a partition of 1GB using the file `/usr/vdisk.img`.

FreeBSD

```
# dd if=/dev/random of=/usr/vdisk.img bs=1K count=1M
```

```
# mdconfig -a -t vnode -f /usr/vdisk.img -u 1 # Creates device /dev/md1
```

```
# bsdlabel -w /dev/md1
```

```
# newfs /dev/md1c
```

```
# mount /dev/md1c /mnt
```

```
# umount /mnt; mdconfig -d -u 1; rm /usr/vdisk.img # Cleanup the md device
```

Linux

```
# dd if=/dev/zero of=/usr/vdisk.img bs=1024k count=1024
```

```
# mkfs.ext3 /usr/vdisk.img
```

2.<http://freshmeat.net/projects/bchunk/>

```
# mount -o loop /usr/vdisk.img /mnt
```

```
# umount /mnt; rm /usr/vdisk.img # Cleanup
```

Linux with `losetup`

`/dev/zero` is much faster than `urandom`, but less secure for encryption.

```
# dd if=/dev/urandom of=/usr/vdisk.img bs=1024k count=1024
```

```
# losetup /dev/loop0 /usr/vdisk.img # Creates and associates /dev/loop0
```

```
# mkfs.ext3 /dev/loop0
```

```
# mount /dev/loop0 /mnt
```

```
# losetup -a # Check used loops
```

```
# umount /mnt
```

```
# losetup -d /dev/loop0 # Detach
```

```
# rm /usr/vdisk.img
```

3.11 Create a memory file system

A memory based file system is very fast for heavy IO application. How to create a 64 MB partition mounted on `/memdisk`:

FreeBSD

```
# mount_mfs -o rw -s 64M md /memdisk
```

```
# umount /memdisk; mdconfig -d -u 0 # Cleanup the md device
```

```
md /memdisk mfs rw,-s64M 0 0 # /etc/fstab entry
```

Linux

```
# mount -t tmpfs -osize=64m tmpfs /memdisk
```

3.12 Disk performance

Read and write a 1 GB file on partition ad4s3c (/home)

```
# time dd if=/dev/ad4s3c of=/dev/null bs=1024k count=1000
```

```
# time dd if=/dev/zero bs=1024k count=1000 of=/home/1Gb.file
```

```
# hdparm -tT /dev/hda # Linux only
```

4. NETWORK

4.1 Debugging (See also Traffic analysis) (page 18)

```
# mii-diag eth0 # Show the link status (Linux)
# ifconfig fxp0 # Check the "media" field on FreeBSD
# arp -a # Check the router (or host) ARP entry (all OS)
# ping cb.vu # The first thing to try...
# traceroute cb.vu # Print the route path to destination
# mii-diag -F 100baseTx-FD eth0 # Force 100Mbit Full duplex (Linux)
# ifconfig fxp0 media 100baseTX mediaopt full-duplex # Same for FreeBSD
# netstat -s # System-wide statistics for each network protocol
```

4.2 Routing

Print routing table

```
# route -n # Linux
```

```
# netstat -rn # Linux, BSD and UNIX
```

```
# route print # Windows
```

Add and delete a route

FreeBSD

```
# route add 212.117.0.0/16 192.168.1.1
```

```
# route delete 212.117.0.0/16
```

```
# route add default 192.168.1.1
```

Add the route permanently in /etc/rc.conf

```
static_routes="myroute"
```

```
route_myroute="-net 212.117.0.0/16 192.168.1.1"
```

Linux

```
# route add -net 192.168.20.0 netmask 255.255.255.0 gw 192.168.16.254
```

```
# ip route add 192.168.20.0/24 via 192.168.16.254 # same as above with ip route
```

```
# route add -net 192.168.20.0 netmask 255.255.255.0 dev eth0
```

```
# route add default gw 192.168.51.254
```

```
# ip route add default via 192.168.51.254 # same as above with ip route
```

```
# route delete -net 192.168.20.0 netmask 255.255.255.0
```

Windows

```
# Route add 192.168.50.0 mask 255.255.255.0 192.168.51.253
```

```
# Route add 0.0.0.0 mask 0.0.0.0 192.168.51.254
```

Use add -p to make the route persistent.

4.3 Configure additional IP addresses

Linux

```
# ifconfig eth0 192.168.50.254 netmask 255.255.255.0 # First IP
```

```
# ifconfig eth0:0 192.168.51.254 netmask 255.255.255.0 # Second IP
```

FreeBSD

```
# ifconfig fxp0 inet 192.168.50.254/24 # First IP
```

```
# ifconfig fxp0 alias 192.168.51.254 netmask 255.255.255.0 # Second IP
Permanent entries in /etc/rc.conf
ifconfig_fxp0="inet 192.168.50.254 netmask 255.255.255.0"
ifconfig_fxp0_alias0="192.168.51.254 netmask 255.255.255.0"
```

4.4 Change MAC address

```
# ifconfig eth0 hw ether 00:01:02:03:04:05 # Linux
# ifconfig fxp0 link 00:01:02:03:04:05 # FreeBSD
```

4.5 Ports in use

Listening open ports:

netstat -an | grep LISTEN

```
# lsof -i # Linux list all Internet connections
# socklist # Linux display list of open sockets
# sockstat -4 # FreeBSD application listing
# netstat -anp --udp --tcp | grep LISTEN # Linux
# netstat -tup # List active connections to/from system (Linux)
# netstat -tupl # List listening ports from system (Linux)
# netstat -ano # Windows
```

4.6 Firewall

Check if a firewall is running (typical configuration only):

Linux

```
# iptables -L -n -v # For status
```

Open the iptables firewall

```
# iptables -Z # Zero the packet and byte counters in all chains
# iptables -F # Flush all chains
# iptables -X # Delete all chains
# iptables -P INPUT ACCEPT # Open everything
```

```
# iptables -P FORWARD ACCEPT
```

```
# iptables -P OUTPUT ACCEPT
```

FreeBSD

```
# ipfw show # For status
# ipfw list 65535 # if answer is "65535 deny ip from any to any" the fw is
disabled
# sysctl net.inet.ip.fw.enable=0 # Disable
# sysctl net.inet.ip.fw.enable=1 # Enable
```

4.7 IP Forward for routing

Linux

Check and then enable IP forward with:

```
# cat /proc/sys/net/ipv4/ip_forward # Check IP forward 0=off, 1=on
```

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

or edit /etc/sysctl.conf with:

```
net.ipv4.ip_forward = 1
```

FreeBSD

Check and enable with:

```
# sysctl net.inet.ip.forwarding # Check IP forward 0=off, 1=on
```

```
# sysctl net.inet.ip.forwarding=1
```

```
# sysctl net.inet.ip.fastforwarding=1 # For dedicated router or firewall
```

Permanent with entry in /etc/rc.conf:

```
gateway_enable="YES" # Set to YES if this host will be a gateway.
```

4.8 NAT Network Address Translation

Linux

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE # to activate NAT
# iptables -t nat -A PREROUTING -p tcp -d 78.31.70.238 --dport 20022 -j DNAT \
--to 192.168.16.44:22 # Port forward 20022 to internal IP port ssh
# iptables -t nat -A PREROUTING -p tcp -d 78.31.70.238 --dport 993:995 -j DNAT \
--to 192.168.16.254:993:995 # Port forward of range 993-995
# ip route flush cache
# iptables -L -t nat # Check NAT status
Delete the port forward with -D instead of -A.
```

FreeBSD

```
# natd -s -m -u -dynamic -f /etc/natd.conf -n fxp0
```

Or edit /etc/rc.conf with:

```
firewall_enable="YES" # Set to YES to enable firewall functionality
firewall_type="open" # Firewall type (see /etc/rc.firewall)
natd_enable="YES" # Enable natd (if firewall_enable == YES).
natd_interface="tun0" # Public interface or IP address to use.
natd_flags="-s -m -u -dynamic -f /etc/natd.conf"
```

Port forward with:

```
# cat /etc/natd.conf
same_ports yes
use_sockets yes
unregistered_only
# redirect_port tcp insideIP:2300-2399 3300-3399 # port range
redirect_port udp 192.168.51.103:7777 7777
```

4.9 DNS

On Unix the DNS entries are valid for all interfaces and are stored in /etc/resolv.conf. The domain to which the host belongs is also stored in this file. A minimal configuration is:

```
nameserver 78.31.70.238
search sleepyowl.net intern.lab
domain sleepyowl.net
```

Check the system domain name with:

```
# hostname -d # Same as dnsdomainname
```

Windows

On Windows the DNS are configured per interface. To display the configured DNS and to flush the DNS cache use:

```
# ipconfig /? # Display help
# ipconfig /all # See all information including DNS
# ipconfig /flushdns # Flush the DNS cache
```

Forward queries

Dig is your friend to test the DNS settings. For example the public DNS server 213.133.105.2

ns.second-ns.de can be used for testing. See from which server the client receives the answer

(simplified answer).

```
# dig sleepyowl.net
```

```
sleepyowl.net. 600 IN A 78.31.70.238
```

```
:: SERVER: 192.168.51.254#53(192.168.51.254)
```

The router 192.168.51.254 answered and the response is the A entry. Any entry can be queried and the DNS server can be selected with @:

The program `host` is also powerful.

```
# host -t MX cb.vu # Get the mail MX entry
# host -t NS -T sun.com # Get the NS record over a TCP connection
# host -a sleepyowl.net # Get everything
```

Reverse queries

Find the name belonging to an IP address (in-addr.arpa.). This can be done with `dig`, `host` and

`nslookup`:

```
# dig -x 78.31.70.238
# host 78.31.70.238
# nslookup 78.31.70.238
```

/etc/hosts

Single hosts can be configured in the file `/etc/hosts` instead of running `named` locally to resolve the hostname queries. The format is simple, for example:

```
78.31.70.238 sleepyowl.net sleepyowl
```

The priority between hosts and a dns query, that is the name resolution order, can be configured in `/etc/nsswitch.conf` AND `/etc/host.conf`. The file also exists on

Windows, it is

usually in:

C:\WINDOWS\SYSTEM32\DRIVERS\ETC

4.10 DHCP

Linux

Some distributions (SuSE) use `dhcpcd` as client. The default interface is `eth0`.

```
# dhcpcd -n eth0 # Trigger a renew
# dhcpcd -k eth0 # release and shutdown
```

The lease with the full information is stored in:

`/var/lib/dhcpcd/dhcpcd-eth0.info`

FreeBSD

FreeBSD (and Debian) uses `dhclient`. To configure an interface (for example `bge0`) run:

```
# dhclient bge0
```

The lease with the full information is stored in:

`/var/db/dhclient.leases.bge0`

Use

`/etc/dhclient.conf`

to prepend options or force different options:

```
# cat /etc/dhclient.conf
```

```
interface "rl0" {
prepend domain-name-servers 127.0.0.1;
default domain-name "sleepyowl.net";
supersede domain-name "sleepyowl.net";
}
```

Windows

```
# ipconfig /renew # renew all adapters
# ipconfig /renew LAN # renew the adapter named "LAN"
# ipconfig /release WLAN # release the adapter named "WLAN"
```

Yes it is a good idea to rename you adapter with simple names!

4.11 Traffic analysis

`Bmon` is a small console bandwidth monitor and can display the flow on different

interfaces.

Sniff with tcpdump

```
# tcpdump -nl -i bge0 not port ssh and src \(192.168.16.121 or 192.168.16.54\)
```

```
# tcpdump -l > dump && tail -f dump # Buffered output
```

```
# tcpdump -i rl0 -w traffic.rl0 # Write traffic in binary file
```

```
# tcpdump -r traffic.rl0 # Read from file (also for ethereal)
```

```
# tcpdump port 80 # The two classic commands
```

tcpdump host google.com

```
# tcpdump -i eth0 -X port \(110 or 143\) # Check if pop or imap is secure
```

```
# tcpdump -n -i eth0 icmp # Only catch pings
```

```
# tcpdump -i eth0 -s 0 -A port 80 | grep GET # -s 0 for full packet -A for ASCII
```

Additional important options:

-A Print each packets in clear text (without header)

-X Print packets in hex and ASCII

-l Make stdout line buffered

-D Print all interfaces available

On Windows use windump from www.winpcap.org. Use windump -D to list the interfaces.

Scan with nmap

Nmap4 is a port scanner with OS detection, it is usually installed on most distributions and is also available for Windows. If you don't scan your servers, hackers do it for you...

```
# nmap cb.vu # scans all reserved TCP ports on the host
```

```
# nmap -sP 192.168.16.0/24 # Find out which IP are used and by which host on 0/24
```

```
# nmap -sS -sV -O cb.vu # Do a stealth SYN scan with version and OS detection
```

PORT STATE SERVICE VERSION

22/tcp open ssh OpenSSH 3.8.1p1 FreeBSD-20060930 (protocol 2.0)

25/tcp open smtp Sendmail smtpd 8.13.6/8.13.6

80/tcp open http Apache httpd 2.0.59 ((FreeBSD) DAV/2 PHP/4.

[...]

Running: FreeBSD 5.X

Uptime 33.120 days (since Fri Aug 31 11:41:04 2007)

4.12 Traffic control (QoS)

Traffic control manages the queuing, policing, scheduling, and other traffic parameters for a network. The following examples are simple practical uses of the Linux and FreeBSD capabilities to better use the available bandwidth.

Limit upload

DSL or cable modems have a long queue to improve the upload throughput. However filling the queue with a fast device (e.g. ethernet) will dramatically decrease the interactivity. It is therefore useful to limit the device upload rate to match the physical capacity of the modem, this should greatly improve the interactivity. Set to about 90% of the modem maximal (cable) speed.

3.<http://people.suug.ch/~tgr/bmon/>

4.<http://insecure.org/nmap/>

Linux

For a 512 Kbit upload modem.

```
# tc qdisc add dev eth0 root tbf rate 480kbit latency 50ms burst 1540
```

```
# tc -s qdisc ls dev eth0 # Status
```

```
# tc qdisc del dev eth0 root # Delete the queue
```

```
# tc qdisc change dev eth0 root tbf rate 220kbit latency 50ms burst 1540
```

FreeBSD

FreeBSD uses the `dummynet` traffic shaper which is configured with `ipfw`. Pipes are used to set limits the bandwidth in units of `[K|M]{bit/s|Byte/s}`, 0 means unlimited bandwidth. Using the same pipe number will reconfigure it. For example limit the upload bandwidth to 500 Kbit.

```
# kldload dummynet # load the module if necessary
# ipfw pipe 1 config bw 500Kbit/s # create a pipe with limited bandwidth
# ipfw add pipe 1 ip from me to any # divert the full upload into the pipe
```

Quality of service**Linux**

Priority queuing with `tc` to optimize VoIP. See the full example on voip-info.org or www.howtoforge.com. Suppose VoIP uses `udp` on ports 10000:11024 and device `eth0` (could also be `ppp0` or `so`). The following commands define the QoS to three queues and force the VoIP traffic to queue 1 with QoS `0x1e` (all bits set). The default traffic flows into queue 3 and QoS

Minimize-Delay flows into queue 2.

```
# tc qdisc add dev eth0 root handle 1: prio priomap 2 2 2 2 2 2 2 2 1 1 1 1 1 1 0
# tc qdisc add dev eth0 parent 1:1 handle 10: sfq
# tc qdisc add dev eth0 parent 1:2 handle 20: sfq
# tc qdisc add dev eth0 parent 1:3 handle 30: sfq
# tc filter add dev eth0 protocol ip parent 1: prio 1 u32 \
match ip dport 10000 0x3C00 flowid 1:1 # use server port range
match ip dst 123.23.0.1 flowid 1:1 # or/and use server IP
```

Status and remove with

```
# tc -s qdisc ls dev eth0 # queue status
# tc qdisc del dev eth0 root # delete all QoS
```

Calculate port range and mask

The `tc` filter defines the port range with port and mask which you have to calculate. Find the 2^N ending of the port range, deduce the range and convert to HEX. This is your mask.

Example for 10000 -> 11024, the range is 1024.

```
# 2^13 (8192) < 10000 < 2^14 (16384) # ending is 2^14 = 16384
# echo "obase=16; (2^14)-1024" | bc # mask is 0x3C00
```

FreeBSD

The max link bandwidth is 500Kbit/s and we define 3 queues with priority 100:10:1 for VoIP:ssh:all the rest.

```
# ipfw pipe 1 config bw 500Kbit/s
# ipfw queue 1 config pipe 1 weight 100
# ipfw queue 2 config pipe 1 weight 10
# ipfw queue 3 config pipe 1 weight 1
# ipfw add 10 queue 1 proto udp dst-port 10000-11024
# ipfw add 11 queue 1 proto udp dst-ip 123.23.0.1 # or/and use server IP
# ipfw add 20 queue 2 dsp-port ssh
# ipfw add 30 queue 3 from me to any # all the rest
```

Status and remove with

```
# ipfw list # rules status
# ipfw pipe list # pipe status
# ipfw flush # deletes all rules but default
```

4.13 NIS Debugging

Some commands which should work on a well configured NIS client:

```
# ypwhich # get the connected NIS server name
# domainname # The NIS domain name as configured
```



```
# ypcat group # should display the group from the NIS server
# cd /var/yp && make # Rebuild the yp database
Is ypbind running?
# ps auxww | grep ypbind
/usr/sbin/ypbind -s -m -S servername1,servername2 # FreeBSD
/usr/sbin/ypbind # Linux
# yppoll passwd.byname
Map passwd.byname has order number 1190635041. Mon Sep 24 13:57:21 2007
The master server is servername.domain.net.
Linux
# cat /etc/yp.conf
ypserver servername
domain domain.net broadcast
```

5. SSH SCP

5.1 Public key authentication

Connect to a host without password using public key authentication. The idea is to append your public key to the `authorized_keys2` file on the remote host. For this example let's connect host-client to host-server, the key is generated on the client.

- Use `ssh-keygen` to generate a key pair. `~/.ssh/id_dsa` is the private key, `~/.ssh/id_dsa.pub` is the public key.
- Copy only the public key to the server and append it to the file `~/.ssh/authorized_keys2` on your home on the server.

```
# ssh-keygen -t dsa -N "
```

```
# cat ~/.ssh/id_dsa.pub | ssh you@host-server "cat - >> ~/.ssh/authorized_keys2"
```

Using the Windows client from ssh.com

The non commercial version of the `ssh.com` client can be downloaded the main ftp site: <ftp.ssh.com/pub/ssh/>. Keys generated by the `ssh.com` client need to be converted for the OpenSSH server. This can be done with the `ssh-keygen` command.

- Create a key pair with the `ssh.com` client: Settings - User Authentication - Generate New....
- I use Key type DSA; key length 2048.
- Copy the public key generated by the `ssh.com` client to the server into the `~/.ssh` folder.
- The keys are in `C:\Documents and Settings\%USERNAME%\Application Data\SSH\UserKeys`.
- Use the `ssh-keygen` command on the server to convert the key:

```
# cd ~/.ssh
```

```
# ssh-keygen -i -f keyfilename.pub >> authorized_keys2
```

Notice: We used a DSA key, RSA is also possible. The key is not protected by a password.

Using putty for Windows

Putty⁵ is a simple and free ssh client for Windows.

- Create a key pair with the `puTTYgen` program.
- Save the public and private keys (for example into `C:\Documents and Settings\%USERNAME%\ssh`).
- Copy the public key to the server into the `~/.ssh` folder:


```
# scp .ssh/puttykey.pub root@192.168.51.254:.ssh/
```

- Use the ssh-keygen command on the server to convert the key for OpenSSH:

```
# cd ~/.ssh
```

```
# ssh-keygen -i -f puttykey.pub >> authorized_keys2
```

- Point the private key location in the putty settings: Connection - SSH - Auth

5.2 Check fingerprint

At the first login, ssh will ask if the unknown host with the fingerprint has to be stored in the known hosts. To avoid a man-in-the-middle attack the administrator of the server can send you the server fingerprint which is then compared on the first login.

Use ssh-keygen -l to get the fingerprint (on the server):

```
# ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub # For RSA key
2048 61:33:be:9b:ae:6c:36:31:fd:83:98:b7:99:2d:9f:cd /etc/ssh/ssh_host_rsa_key.pub
# ssh-keygen -l -f /etc/ssh/ssh_host_dsa_key.pub # For DSA key (default)
2048 14:4a:aa:d9:73:25:46:6d:0a:48:35:c7:f4:16:d4:ee /etc/ssh/ssh_host_dsa_key.pub
```

Now the client connecting to this server can verify that he is connecting to the right server:

```
# ssh linda
```

The authenticity of host 'linda (192.168.16.54)' can't be established.

DSA key fingerprint is 14:4a:aa:d9:73:25:46:6d:0a:48:35:c7:f4:16:d4:ee.

Are you sure you want to continue connecting (yes/no)? yes

5.3 Secure file transfer

Some simple commands:

```
# scp file.txt host-two:/tmp
# scp joe@host-two:/www/*.html /www/tmp
# scp -r joe@host-two:/www /www/tmp
```

In Konqueror or Midnight Commander it is possible to access a remote file system with the address fish://user@gate. However the implementation is very slow.

Furthermore it is possible to mount a remote folder with sshfs a file system client based on SCP. See [fuse sshfs](#).

5.4 Tunneling

SSH tunneling allows to forward or reverse forward a port over the SSH connection, thus securing the traffic and accessing ports which would otherwise be blocked. This only works with TCP. The general nomenclature for forward and reverse is (see also [ssh and NAT example](#)):

```
# ssh -L localport:desthost:destport user@gate # desthost as seen from the gate
# ssh -R destport:desthost:localport user@gate # forwards your localport to destination
```

```
# ssh -X user@gate # To force X forwarding
```

5.<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

6.<http://fuse.sourceforge.net/sshfs.html>

This will connect to gate and forward the local port to the host desthost:destport. Note desthost is the destination host as seen by the gate, so if the connection is to the gate, then desthost is localhost. More than one port forward is possible.

[Direct forward on the gate](#)

Let say we want to access the CVS (port 2401) and http (port 80) which are running on the gate. This is the simplest example, desthost is thus localhost, and we use the port 8080 locally instead of 80 so we don't need to be root. Once the ssh session is open,

both services are accessible on the local ports.

```
# ssh -L 2401:localhost:2401 -L 8080:localhost:80 user@gate
```

Netbios and remote desktop forward to a second server

Let say a Windows smb server is behind the gate and is not running ssh. We need access to the smb share and also remote desktop to the server.

```
# ssh -L 139:smbserver:139 -L 3388:smbserver:3389 user@gate
```

The smb share can now be accessed with \\127.0.0.1\, but only if the local share is disabled, because the local share is listening on port 139.

It is possible to keep the local share enabled, for this we need to create a new virtual device with a new IP address for the tunnel, the smb share will be connected over this address.

Furthermore the local RDP is already listening on 3389, so we choose 3388. For this example let's use a virtual IP of 10.1.1.1.

- With putty use Source port=10.1.1.1:139. It is possible to create multiple loop devices

and tunnel. On Windows 2000, only putty worked for me.

- With the ssh.com client, disable "Allow local connections only". Since ssh.com will bind

to all addresses, only a single share can be connected.

Now create the loopback interface with IP 10.1.1.1:

- # System->Control Panel->Add Hardware # Yes, Hardware is already connected # Add

a new hardware device (at bottom).

- # Install the hardware that I manually select # Network adapters # Microsoft , Microsoft

Loopback Adapter.

- Configure the IP address of the fake device to 10.1.1.1 mask 255.255.255.0, no gateway.

- advanced->WINS, Enable LMHosts Lookup; Disable NetBIOS over TCP/IP.

- # Enable Client for Microsoft Networks. # Disable File and Printer Sharing for Microsoft

Networks.

I HAD to reboot for this to work. Now connect to the smb share with \\10.1.1.1 and remote

desktop to 10.1.1.1:3388.

Debug

If it is not working:

- Are the ports forwarded: netstat -an? Look at 0.0.0.0:139 or 10.1.1.1:139
- Does telnet 10.1.1.1 139 connect?
- You need the checkbox "Local ports accept connections from other hosts".
- Is "File and Printer Sharing for Microsoft Networks" disabled on the loopback interface?

Connect two clients behind NAT

Suppose two clients are behind a NAT gateway and client cliadmin has to connect to client

cliuser (the destination), both can login to the gate with ssh and are running Linux with sshd.

You don't need root access anywhere as long as the ports on gate are above 1024. We use 2022

on gate. Also since the gate is used locally, the option GatewayPorts is not necessary.

On client cliuser (from destination to gate):

```
# ssh -R 2022:localhost:22 user@gate # forwards client 22 to gate:2022
```

On client cliadmin (from host to gate):

```
# ssh -L 3022:localhost:2022 admin@gate # forwards client 3022 to gate:2022
```

Now the admin can connect directly to the client cliuser with:

```
# ssh -p 3022 admin@localhost # local:3022 -> gate:2022 -> client:22
```

Connect to VNC behind NAT

Suppose a Windows client with VNC listening on port 5900 has to be accessed from behind NAT.

On client cliwin to gate:

```
# ssh -R 15900:localhost:5900 user@gate
```

On client cliadmin (from host to gate):

```
# ssh -L 5900:localhost:15900 admin@gate
```

Now the admin can connect directly to the client VNC with:

```
# vncconnect -display :0 localhost
```

6. VPN WITH SSH

As of version 4.3, OpenSSH can use the tun/tap device to encrypt a tunnel. This is very similar to other TLS based VPN solutions like OpenVPN. One advantage with SSH is that there is no need to install and configure additional software. Additionally the tunnel uses the SSH authentication like pre shared keys. The drawback is that the encapsulation is done over TCP which might result in poor performance on a slow link. Also the tunnel is relying on a single (fragile) TCP connection. This technique is very useful for a quick IP based VPN setup. There is no limitation as with the single TCP port forward, all layer 3/4 protocols like ICMP, TCP/UDP, etc. are forwarded over the VPN. In any case, the following options are needed in the sshd_conf file:

```
PermitRootLogin yes
```

```
PermitTunnel yes
```

6.1 Single P2P connection

Here we are connecting two hosts, hclient and hserver with a peer to peer tunnel. The connection is started from hclient to hserver and is done as root. The tunnel end points are 10.0.1.1 (server) and 10.0.1.2 (client) and we create a device tun5 (this could also be an other number). The procedure is very simple:

- Connect with SSH using the tunnel option -w
- Configure the IP addresses of the tunnel. Once on the server and once on the client.

Connect to the server

Connection started on the client and commands are executed on the server.

Server is on Linux

```
cli># ssh -w5:5 root@hserver
```

```
srv># ifconfig tun5 10.0.1.1 netmask 255.255.255.252 # Executed on the server shell
```

Server is on FreeBSD

```
cli># ssh -w5:5 root@hserver
```

```
srv># ifconfig tun5 10.0.1.1 10.0.1.2 # Executed on the server shell
```

Configure the client

Commands executed on the client:

```
cli># ifconfig tun5 10.0.1.2 netmask 255.255.255.252 # Client is on Linux
```

```
cli># ifconfig tun5 10.0.1.2 10.0.1.1 # Client is on FreeBSD
```

The two hosts are now connected and can transparently communicate with any layer 3/4

protocol using the tunnel IP addresses.

6.2 Connect two networks

In addition to the p2p setup above, it is more useful to connect two private networks with an SSH VPN using two gates. Suppose for the example, netA is 192.168.51.0/24 and netB 192.168.16.0/24. The procedure is similar as above, we only need to add the routing. NAT must be activated on the private interface only if the gates are not the same as the default gateway of their network.

192.168.51.0/24 (netA)|gateA <-> gateB|192.168.16.0/24 (netB)

- Connect with SSH using the tunnel option -w.
- Configure the IP addresses of the tunnel. Once on the server and once on the client.
- Add the routing for the two networks.
- If necessary, activate NAT on the private interface of the gate.

The setup is started from gateA in netA.

Connect from gateA to gateB

Connection is started from gateA and commands are executed on gateB.

gateB is on Linux

```
gateA># ssh -w5:5 root@gateB
gateB># ifconfig tun5 10.0.1.1 netmask 255.255.255.252 # Executed on the gateB shell
gateB># route add -net 192.168.51.0 netmask 255.255.255.0 dev tun5
gateB># echo 1 > /proc/sys/net/ipv4/ip_forward # Only needed if not default gw
gateB># iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

gateB is on FreeBSD

```
gateA># ssh -w5:5 root@gateB # Creates the tun5 devices
gateB># ifconfig tun5 10.0.1.1 10.0.1.2 # Executed on the gateB shell
gateB># route add 192.168.51.0/24 10.0.1.2
gateB># sysctl net.inet.ip.forwarding=1 # Only needed if not default gw
gateB># natd -s -m -u -dynamic -n fxp0 # see NAT (page 15)
gateA># sysctl net.inet.ip.fw.enable=1
```

Configure gateA

Commands executed on gateA:

gateA is on Linux

```
gateA># ifconfig tun5 10.0.1.2 netmask 255.255.255.252
gateA># route add -net 192.168.16.0 netmask 255.255.255.0 dev tun5
gateA># echo 1 > /proc/sys/net/ipv4/ip_forward
gateA># iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

gateA is on FreeBSD

```
gateA># ifconfig tun5 10.0.1.2 10.0.1.1
gateA># route add 192.168.16.0/24 10.0.1.2
gateA># sysctl net.inet.ip.forwarding=1
gateA># natd -s -m -u -dynamic -n fxp0 # see NAT (page 15)
gateA># sysctl net.inet.ip.fw.enable=1
```

The two private networks are now transparently connected via the SSH VPN. The IP forward and NAT settings are only necessary if the gates are not the default gateways. In this case the clients would not know where to forward the response, and nat must be activated.

7. RSYNC

Rsync can almost completely replace cp and scp, furthermore interrupted transfers are

efficiently restarted. A trailing slash (and the absence thereof) has different meanings, the man page is good... Here some examples:

Copy the directories with full content:

```
# rsync -a /home/colin/ /backup/colin/
```

```
# rsync -a /var/ /var_bak/
```

```
# rsync -aR --delete-during /home/user/ /backup/ # use relative (see below)
```

Same as before but over the network and with compression. Rsync uses SSH for the transport per default and will use the ssh key if they are set. Use ":" as with SCP. A typical remote copy:

```
# rsync -axSRzv /home/user/ user@server:/backup/user/
```

Exclude any directory tmp within /home/user/ and keep the relative folders hierarchy, that is the remote directory will have the structure /backup/home/user/. This is typically used for backups.

```
# rsync -azR --exclude /tmp/ /home/user/ user@server:/backup/
```

Use port 20022 for the ssh connection:

```
# rsync -az -e 'ssh -p 20022' /home/colin/ user@server:/backup/colin/
```

Using the rsync daemon (used with "::") is much faster, but not encrypted over ssh. The

location of /backup is defined by the configuration in /etc/rsyncd.conf. The variable RSYNC_PASSWORD can be set to avoid the need to enter the password manually.

```
# rsync -axSRz /home/ ruser@hostname::rmodule/backup/
```

```
# rsync -axSRz ruser@hostname::rmodule/backup/ /home/ # To copy back
```

Some important options:

-a, --archive archive mode; same as -rlptgoD (no -H)

-r, --recursive recurse into directories

-R, --relative use relative path names

-H, --hard-links preserve hard links

-S, --sparse handle sparse files efficiently

-x, --one-file-system don't cross file system boundaries

--exclude=PATTERN exclude files matching PATTERN

--delete-during receiver deletes during xfer, not before

--delete-after receiver deletes after transfer, not before

7.1 Rsync on Windows

Rsync is available for Windows through cygwin or as stand-alone packaged in [cwrsync7](#). This is very convenient for automated backups. Install one of them (not both) and add the path to the Windows system variables: # Control Panel -> System -> tab

Advanced, button Environment

Variables. Edit the "Path" system variable and add the full path to the installed rsync, e.g.

C:\Program Files\cwRsync\bin or C:\cygwin\bin. This way the commands `rsync` and `ssh` are available in a Windows command shell.

Public key authentication

Rsync is automatically tunneled over SSH and thus uses the SSH authentication on the server.

Automatic backups have to avoid a user interaction, for this the SSH public key authentication can be used and the rsync command will run without a password.

All the following commands are executed within a Windows console. In a console (Start -> Run -> cmd) create and upload the key as described in [SSH](#), change "user" and "server" as appropriate. If the file `authorized_keys2` does not exist yet, simply copy `id_dsa.pub` to `authorized_keys2` and upload it.

```
# ssh-keygen -t dsa -N '' # Creates a public and a private key
# rsync user@server:~/.ssh/authorized_keys2 . # Copy the file locally from the
server
# cat id_dsa.pub >> authorized_keys2 # Or use an editor to add the key
# rsync authorized_keys2 user@server:~/.ssh/ # Copy the file back to the server
# del authorized_keys2 # Remove the local copy
```

Now test it with (in one line):

```
rsync -rv "/cygdrive/c/Documents and Settings/%USERNAME%/My Documents/" \
'user@server:My\ Documents'
```

Automatic backup

Use a batch file to automate the backup and add the file in the scheduled tasks (Programs -> Accessories -> System Tools -> Scheduled Tasks). For example create the file backup.bat and

replace user@server.

```
@ECHO OFF
```

```
REM rsync the directory My Documents
```

```
SETLOCAL
```

```
SET CWRSYNCHOME=C:\PROGRAM FILES\CWRSYNC
```

```
SET CYGWIN=nontsec
```

```
SET CWOLDPATH=%PATH%
```

```
REM uncomment the next line when using cygwin
```

```
SET PATH=%CWRSYNCHOME%\BIN;%PATH%
```

```
echo Press Control-C to abort
```

```
rsync -av "/cygdrive/c/Documents and Settings/%USERNAME%/My Documents/" \
'user@server:My\ Documents'
```

```
pause
```

8. SUDO

Sudo is a standard way to give users some administrative rights without giving out the root password. Sudo is very useful in a multi user environment with a mix of server and workstations. Simply call the command with sudo:

```
# sudo /etc/init.d/dhcpd restart # Run the rc script as root
# sudo -u sysadmin whoami # Run cmd as an other user
```

8.1 Configuration

Sudo is configured in `/etc/sudoers` and must only be edited with `visudo`. The basic syntax is

(the lists are comma separated):

```
user hosts = (runas) commands # In /etc/sudoers
```

`users` one or more users or %group (like %wheel) to gain the rights

`hosts` list of hosts (or ALL)

`runas` list of users (or ALL) that the command rule can be run as. It is enclosed in ()!

`commands` list of commands (or ALL) that will be run as root or as (runas)

Additionally those keywords can be defined as alias, they are called User_Alias, Host_Alias, Runas_Alias and Cmnd_Alias. This is useful for larger setups. Here a sudoers example:

```
# cat /etc/sudoers
```

```
# Host aliases are subnets or hostnames.
```

```
Host_Alias DMZ = 212.118.81.40/28
```

```
User_Alias ADMINS = colin, luca, admin
```

```
User_Alias DEVEL = joe, jack, julia
```

```

Runas_Alias DBA = oracle,pgsql
# Command aliases define the full path of a list of commands
Cmd_Alias SYSTEM = /sbin/reboot,/usr/bin/kill,/sbin/halt,/sbin/shutdown,/etc/init.d/
Cmd_Alias PW = /usr/bin/passwd [A-z]*, !/usr/bin/passwd root # Not root pwd!
Cmd_Alias DEBUG = /usr/sbin/tcpdump,/usr/bin/wireshark,/usr/bin/nmap
# The actual rules
root,ADMINS ALL = (ALL) NOPASSWD: ALL # ADMINS can do anything w/o a password.
DEVEL DESKTOP = (ALL) NOPASSWD: ALL # Developers have full right on desktops
DEVEL DMZ = (ALL) NOPASSWD: DEBUG # Developers can debug the DMZ servers.
# User sysadmin can mess around in the DMZ servers with some commands.
sysadmin DMZ = (ALL) NOPASSWD: SYSTEM,PW,DEBUG
sysadmin ALL,!DMZ = (ALL) NOPASSWD: ALL # Can do anything outside the DMZ.
%dba ALL = (DBA) ALL # Group dba can run as database user.
# anyone can mount/unmount a cd-rom on the desktop machines
ALL DESKTOP = NOPASSWD: /sbin/mount /cdrom,/sbin/umount /cdrom

```

9. ENCRYPT FILES

9.1 A single file

Encrypt and decrypt:

```

# openssl des -salt -in file -out file.des
# openssl des -d -salt -in file.des -out file

```

Note that the file can of course be a tar archive.

9.2 tar and encrypt a whole directory

```

# tar -cf - directory | openssl des -salt -out directory.tar.des # Encrypt
# openssl des -d -salt -in directory.tar.des | tar -x # Decrypt

```

9.3 tar zip and encrypt a whole directory

```

# tar -zcf - directory | openssl des -salt -out directory.tar.gz.des # Encrypt
# openssl des -d -salt -in directory.tar.gz.des | tar -xz # Decrypt

```

- Use `-k mysecretpassword` after `des` to avoid the interactive password request. However note that this is highly insecure.
- Use `des3` instead of `des` to get even stronger encryption (Triple-DES Cipher). This uses also more CPU.

10. ENCRYPT PARTITIONS

There are (many) other alternative methods to encrypt disks, I only show here the methods I know and use. Keep in mind that the security is only good as long the OS has not been tempered with. An intruder could easily record the password from the keyboard events.

Furthermore the data is freely accessible when the partition is attached and will not prevent an intruder to have access to it in this state.

10.1 Linux

Those instructions use the Linux `dm-crypt` (device-mapper) facility available on the 2.6 kernel.

In this example, lets encrypt the partition `/dev/sdc1`, it could be however any other

partition or disk, or USB or a file based partition created with `losetup`. In this case we would use `/dev/loop0`. See [file image partition](#). The device mapper uses labels to identify a partition. We use `sdcl` in this example, but it could be any string.

dm-crypt with LUKS

LUKS with dm-crypt has better encryption and makes it possible to have multiple passphrase for the same partition or to change the password easily. To test if LUKS is available, simply type #

`cryptsetup --help`, if nothing about LUKS shows up, use the instructions below

Without LUKS.

First create a partition if necessary: `fdisk /dev/sdc`.

Create encrypted partition

```
# dd if=/dev/urandom of=/dev/sdc1 # Optional. For paranoids only (takes days)
# cryptsetup -y luksFormat /dev/sdc1 # This destroys any data on sdc1
# cryptsetup luksOpen /dev/sdc1 sdc1
# mkfs.ext3 /dev/mapper/sdc1 # create ext3 file system
# mount -t ext3 /dev/mapper/sdc1 /mnt
# umount /mnt
# cryptsetup luksClose sdc1 # Detach the encrypted partition
```

Attach

```
# cryptsetup luksOpen /dev/sdc1 sdc1
# mount -t ext3 /dev/mapper/sdc1 /mnt
```

Detach

```
# umount /mnt
# cryptsetup luksClose sdc1
```

dm-crypt without LUKS

```
# cryptsetup -y create sdc1 /dev/sdc1 # or any other partition like /dev/loop0
# dmsetup ls # check it, will display: sdc1 (254, 0)
# mkfs.ext3 /dev/mapper/sdc1 # This is done only the first time!
# mount -t ext3 /dev/mapper/sdc1 /mnt
# umount /mnt/
# cryptsetup remove sdc1 # Detach the encrypted partition
```

Do exactly the same (without the mkfs part!) to re-attach the partition. If the password is not correct, the mount command will fail. In this case simply remove the map `sdcl` (`cryptsetup remove sdc1`) and create it again.

10.2 FreeBSD

The two popular FreeBSD disk encryption modules are `gbde` and `geli`. I now use `geli` because it is faster and also uses the crypto device for hardware acceleration. See [The FreeBSD handbook Chapter 18.6](#) for all the details. The `geli` module must be loaded or compiled into the kernel:

options GEOM_ELI

```
device crypto # or as module:
# echo 'geom_eli_load="YES"' >> /boot/loader.conf # or do: kldload geom_eli
8.http://www.freebsd.org/handbook/disks-encrypting.html
```

Use password and key

I use those settings for a typical disk encryption, it uses a passphrase AND a key to encrypt the master key. That is you need both the password and the generated key `/root/ad1.key` to attach the partition. The master key is stored inside the partition and is not visible. See below for typical USB or file based image.

Create encrypted partition

```
# dd if=/dev/random of=/root/ad1.key bs=64 count=1 # this key encrypts the mater
```



```
key
# geli init -s 4096 -K /root/ad1.key /dev/ad1 # -s 8192 is also OK for disks
# geli attach -k /root/ad1.key /dev/ad1 # DO make a backup of /root/ad1.key
# dd if=/dev/random of=/dev/ad1.eli bs=1m # Optional and takes a long time
# newfs /dev/ad1.eli # Create file system
```

```
# mount /dev/ad1.eli /mnt
```

Attach

```
# geli attach -k /root/ad1.key /dev/ad1
```

```
# fsck -ny -t ffs /dev/ad1.eli # In doubt check the file system
```

```
# mount /dev/ad1.eli /mnt
```

Detach

The detach procedure is done automatically on shutdown.

```
# umount /mnt
```

```
# geli detach /dev/ad1.eli
```

/etc/fstab

The encrypted partition can be configured to be mounted with /etc/fstab. The password will be

prompted when booting. The following settings are required for this example:

```
# grep geli /etc/rc.conf
```

```
geli_devices="ad1"
```

```
geli_ad1_flags="-k /root/ad1.key"
```

```
# grep geli /etc/fstab
```

```
/dev/ad1.eli /home/private ufs rw 0 0
```

Use password only

It is more convenient to encrypt a USB stick or file based image with a passphrase only and no key. In this case it is not necessary to carry the additional key file around. The procedure is very much the same as above, simply without the key file. Let's encrypt a file based image

/cryptedfile of 1 GB.

```
# dd if=/dev/zero of=/cryptedfile bs=1M count=1000 # 1 GB file
```

```
# mdconfig -at vnode -f /cryptedfile
```

```
# geli init /dev/md0 # encrypts with password only
```

```
# geli attach /dev/md0
```

```
# newfs -U -m 0 /dev/md0.eli
```

```
# mount /dev/md0.eli /mnt
```

```
# umount /dev/md0.eli
```

```
# geli detach md0.eli
```

It is now possible to mount this image on an other system with the password only.

```
# mdconfig -at vnode -f /cryptedfile
```

```
# geli attach /dev/md0
```

```
# mount /dev/md0.eli /mnt
```

— Encrypt Partitions —

29

11. SSL CERTIFICATES

So called SSL/TLS certificates are cryptographic public key certificates and are composed of a public and a private key. The certificates are used to authenticate the endpoints and encrypt the data. They are used for example on a web server (https) or mail server (imaps).

11.1 Procedure

- We need a certificate authority to sign our certificate. This step is usually provided by a vendor like Thawte, Verisign, etc., however we can also create our own.
- Create a certificate signing request. This request is like an unsigned certificate (the public part) and already contains all necessary information. The certificate request is normally sent to the authority vendor for signing. This step also creates the private key on the local machine.
- Sign the certificate with the certificate authority.
- If necessary join the certificate and the key in a single file to be used by the application (web server, mail server etc.).

11.2 Configure OpenSSL

We use /usr/local/certs as directory for this example check or edit /etc/ssl/openssl.cnf accordingly to your settings so you know where the files will be created. Here are the relevant part of openssl.cnf:

```
[ CA_default ]
```

```
dir = /usr/local/certs/CA # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/index.txt # database index file.
```

Make sure the directories exist or create them

```
# mkdir -p /usr/local/certs/CA
# cd /usr/local/certs/CA
# mkdir certs crl newcerts private
# echo "01" > serial # Only if serial does not exist
# touch index.txt
```

11.3 Create a certificate authority

If you do not have a certificate authority from a vendor, you'll have to create your own. This

step is not necessary if one intend to use a vendor to sign the request. To make a certificate

authority (CA):

```
# openssl req -new -x509 -days 730 -config /etc/ssl/openssl.cnf \
-keyout CA/private/cakey.pem -out CA/cacert.pem
```

11.4 Create a certificate signing request

To make a new certificate (for mail server or web server for example), first create a request certificate with its private key. If your application do not support encrypted private key (for example UW-IMAP does not), then disable encryption with `-nodes`.

```
# openssl req -new -keyout newkey.pem -out newreq.pem \
-config /etc/ssl/openssl.cnf
# openssl req -nodes -new -keyout newkey.pem -out newreq.pem \
-config /etc/ssl/openssl.cnf # No encryption for the key
```

— SSL Certificates —

30

11.5 Sign the certificate

The certificate request has to be signed by the CA to be valid, this step is usually done by the vendor. Note: replace "servername" with the name of your server in the next commands.

```
# cat newreq.pem newkey.pem > new.pem
# openssl ca -policy policy_anything -out servernamecert.pem \
-config /etc/ssl/openssl.cnf -infile new.pem
# mv newkey.pem servernamekey.pem
```

Now servernamekey.pem is the private key and servernamecert.pem is the server certificate.

11.6 Create united certificate

The IMAP server wants to have both private key and server certificate in the same file. And in general, this is also easier to handle, but the file has to be kept securely!. Apache also can deal with it well. Create a file servername.pem containing both the certificate and key.

- Open the private key (servernamekey.pem) with a text editor and copy the private key into the "servername.pem" file.

- Do the same with the server certificate (servernamecert.pem).

The final servername.pem file should look like this:

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDutWy+o/XZ/[...]qK5LqQgT3c9dU6fcR+WuSs6aejdEDDqBRQ
-----END RSA PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIERzCCA7CgAwIBAgIBBDANB[...]iG9w0BAQQFADCBxTELMakGA1UEBhMCREUx
-----END CERTIFICATE-----
```

What we have now in the directory /usr/local/certs/:

```
CA/private/cakey.pem (CA server private key)
CA/cacert.pem (CA server public key)
certs/servernamekey.pem (server private key)
certs/servernamecert.pem (server signed certificate)
certs/servername.pem (server certificate with private key)
Keep the private key secure!
```

11.7 View certificate information

To view the certificate information simply do:

```
# openssl x509 -text -in servernamecert.pem # View the certificate info
# openssl req -noout -text -in server.csr # View the request info
```

12. CVS

12.1 Server setup

Initiate the CVS

Decide where the main repository will rest and create a root cvs. For example /usr/local/cvs (as root):

```
# mkdir -p /usr/local/cvs
```

```
# setenv CVSROOT /usr/local/cvs # Set CVSROOT to the new location (local)
# cvs init # Creates all internal CVS config files
# cvs checkout CVSROOT # Checkout the config files to modify them
```

cd CVSROOT

edit config (fine as it is)

cvs commit config

```
cat >> writers # Create a writers file (optionally also readers)
```

colin

```
^D # Use [Control][D] to quit the edit
```

```
# cvs add writers # Add the file writers into the repository
```

cvs edit checkoutlist

cat >> checkoutlist

writers

```
^D # Use [Control][D] to quit the edit
```

```
# cvs commit # Commit all the configuration changes
```

Add a readers file if you want to differentiate read and write permissions Note: Do not (ever) edit files directly into the main cvs, but rather checkout the file, modify it and check it in. We did this with the file writers to define the write access.

There are three popular ways to access the CVS at this point. The first two don't need any

further configuration. See the examples on [CVSROOT](#) below for how to use them:

- Direct local access to the file system. The user(s) need sufficient file permission to access the CS directly and there is no further authentication in addition to the OS login. However this is only useful if the repository is local.
- Remote access with ssh with the ext protocol. Any use with an ssh shell account and read/write permissions on the CVS server can access the CVS directly with ext over ssh without any additional tunnel. There is no server process running on the CVS for this to work. The ssh login does the authentication.
- Remote access with pserver. This is the preferred use for larger user base as the users

are authenticated by the CVS pserver with a dedicated password database, there is therefore no need for local users accounts. This setup is explained below.

Network setup with inetd

The CVS can be run locally only if a network access is not needed. For a remote access, the

daemon inetd can start the pserver with the following line in /etc/inetd.conf

(/etc/xinetd.d/cvs

on SuSE):

```
cvspserver stream tcp nowait cvs /usr/bin/cvs cvs \
```

```
--allow-root=/usr/local/cvs pserver
```

It is a good idea to block the cvs port from the Internet with the firewall and use an ssh tunnel

to access the repository remotely.

Separate authentication

It is possible to have cvs users which are not part of the OS (no local users). This is actually probably wanted too from the security point of view. Simply add a file named passwd (in the CVSROOT directory) containing the users login and password in the crypt format. This is can be done with the apache htpasswd tool.

Note: This passwd file is the only file which has to be edited directly in the CVSROOT directory.

Also it won't be checked out. More info with htpasswd --help

```
# htpasswd -cb passwd user1 password1 # -c creates the file
```

```
# htpasswd -b passwd user2 password2
```

Now add `:cvs` at the end of each line to tell the cvs server to change the user to cvs (or whatever your cvs server is running under). It looks like this:

```
# cat passwd
user1:xsFjhU22u8Fuo:cvs
```

12.2 Test it

Test the login as normal user (for example here me)

```
# cvs -d :pserver:colin@192.168.50.254:/usr/local/cvs login
Logging in to :pserver:colin@192.168.50.254:2401/usr/local/cvs
CVS password:
```

CVSROOT variable

This is an environment variable used to specify the location of the repository we're doing operations on. For local use, it can be just set to the directory of the repository. For use over the network, the transport protocol must be specified. Set the CVSROOT variable with `setenv`

CVSROOT string on a csh, tcsh shell, or with `export CVSROOT=string` on a sh, bash shell.

```
# setenv CVSROOT :pserver:<username>@<host>:/cvsdirectory
```

For example:

```
# setenv CVSROOT /usr/local/cvs # Used locally only
# setenv CVSROOT :local:/usr/local/cvs # Same as above
# setenv CVSROOT :ext:user@cvsserver:/usr/local/cvs # Direct access with SSH
# setenv CVS_RSH ssh # for the ext access
# setenv CVSROOT :pserver:user@cvsserver.254:/usr/local/cvs # network with pserver
```

When the login succeeded one can import a new project into the repository: cd into your

project root directory

```
cvs import <module name> <vendor tag> <initial tag>
```

```
cvs -d :pserver:colin@192.168.50.254:/usr/local/cvs import MyProject MyCompany
START
```

Where MyProject is the name of the new project in the repository (used later to checkout). Cvs

will import the current directory content into the new project.

To checkout:

```
# cvs -d :pserver:colin@192.168.50.254:/usr/local/cvs checkout MyProject
or
```

```
# setenv CVSROOT :pserver:colin@192.168.50.254:/usr/local/cvs
```

```
# cvs checkout MyProject
```

12.3 SSH tunneling for CVS

We need 2 shells for this. On the first shell we connect to the cvs server with ssh and port-forward the cvs connection. On the second shell we use the cvs normally as if it where running locally.

on shell 1:

```
# ssh -L2401:localhost:2401 colin@cvs_server # Connect directly to the CVS server.
```

Or:

```
# ssh -L2401:cvs_server:2401 colin@gateway # Use a gateway to reach the CVS
```

on shell 2:

```
# setenv CVSROOT :pserver:colin@localhost:/usr/local/cvs
# cvs login
Logging in to :pserver:colin@localhost:2401/usr/local/cvs
CVS password:
# cvs checkout MyProject/src
```

12.4 CVS commands and usage

Import

The import command is used to add a whole directory, it must be run from within the directory to be imported. Say the directory /devel/ contains all files and subdirectories to be imported.

The directory name on the CVS (the module) will be called "myapp".

```
# cvs import [options] directory-name vendor-tag release-tag
```

```
# cd /devel # Must be inside the project to import it
```

```
# cvs import myapp Company R1_0 # Release tag can be anything in one word
```

After a while a new directory "/devel/tools/" was added and it has to be imported too.

```
# cd /devel/tools
```

```
# cvs import myapp/tools Company R1_0
```

Checkout update add commit

```
# cvs co myapp/tools # Will only checkout the directory tools
```

```
# cvs co -r R1_1 myapp # Checkout myapp at release R1_1 (is sticky)
```

```
# cvs -q -d update -P # A typical CVS update
```

```
# cvs update -A # Reset any sticky tag (or date, option)
```

```
# cvs add newfile # Add a new file
```

```
# cvs add -kb newfile # Add a new binary file
```

```
# cvs commit file1 file2 # Commit the two files only
```

```
# cvs commit -m "message" # Commit all changes done with a message
```

Create a patch

It is best to create and apply a patch from the working development directory related to the project, or from within the source directory.

```
# cd /devel/project
```

```
# diff -Naur olddir newdir > patchfile # Create a patch from a directory or a file
```

```
# diff -Naur oldfile newfile > patchfile
```

Apply a patch

Sometimes it is necessary to strip a directory level from the patch, depending how it was created. In case of difficulties, simply look at the first lines of the patch and try -p0, -p1 or -p2.

```
# cd /devel/project
```

```
# patch --dry-run -p0 < patchfile # Test the path without applying it
```

```
# patch -p0 < patchfile
```

```
# patch -p1 < patchfile # strip off the 1st level from the path
```

13. SVN

Subversion (SVN)⁹ is a version control system designed to be the successor of CVS (Concurrent Versions System). The concept is similar to CVS, but many shortcomings were improved. See also the [SVN book](#)¹⁰.

13.1 Server setup

The initiation of the repository is fairly simple (here for example /home/svn/ must

exist):

```
# svnadmin create --fs-type fsfs /home/svn/project1
```

9.<http://subversion.tigris.org/>

10.<http://svnbook.red-bean.com/en/1.4/>

Now the access to the repository is made possible with:

- `file://` Direct file system access with the svn client with. This requires local permissions on the file system.
- `svn://` or `svn+ssh://` Remote access with the svnserve server (also over SSH). This requires local permissions on the file system.
- `http://` Remote access with webdav using apache. No local users are necessary for this method.

Using the local file system, it is now possible to import and then check out an existing project.

Unlike with CVS it is not necessary to cd into the project directory, simply give the full path:

```
# svn import /project1/ file:///home/svn/project1/trunk -m 'Initial import'
```

```
# svn checkout file:///home/svn/project1
```

The new directory "trunk" is only a convention, this is not required.

Remote access with ssh

No special setup is required to access the repository via ssh, simply replace `file://` with

`svn+ssh/hostname`. For example:

```
# svn checkout svn+ssh://hostname/home/svn/project1
```

As with the local file access, every user needs an ssh access to the server (with a local account) and also read/write access. This method might be suitable for a small group. All users could belong to a subversion group which owns the repository, for example:

```
# groupadd subversion
```

```
# groupmod -A user1 subversion
```

```
# chown -R root:subversion /home/svn
```

```
# chmod -R 770 /home/svn
```

Remote access with http (apache)

Remote access over http (https) is the only good solution for a larger user group. This method

uses the apache authentication, not the local accounts. This is a typical but small apache

configuration:

```
LoadModule dav_module modules/mod_dav.so
```

```
LoadModule dav_svn_module modules/mod_dav_svn.so
```

```
LoadModule authz_svn_module modules/mod_authz_svn.so # Only for access control
```

```
<Location /svn>
```

```
DAV svn
```

```
# any "/svn/foo" URL will map to a repository /home/svn/foo
```

```
SVNParentPath /home/svn
```

```
AuthType Basic
```

```
AuthName "Subversion repository"
```

```
AuthzSVNAccessFile /etc/apache2/svn.acl
```

```
AuthUserFile /etc/apache2/svn-passwd
```

```
Require valid-user
```

```
</Location>
```

The apache server needs full access to the repository:

```
# chown -R www:www /home/svn
```

Create a user with htpasswd2:

```
# htpasswd -c /etc/svn-passwd user1 # -c creates the file
```

Access control svn.acl example

```
# Default it read access. "*" = " would be default no access
```

```
[/]
```

```
* = r
```

```
project1-developers = joe, jack, jane
```

```
# Give write access to the developers
```

```
[project1:]
```

```
@project1-developers = rw
```

13.2 SVN commands and usage

See also the [Subversion Quick Reference Card](#)¹¹. Tortoise SVN¹² is a nice Windows interface.

Import

A new project, that is a directory with some files, is imported into the repository with the `import` command. Import is also used to add a directory with its content to an existing project.

```
# svn help import # Get help for any command
```

```
# Add a new directory (with content) into the src dir on project1
```

```
# svn import /project1/newdir http://host.url/svn/project1/trunk/src -m 'add newdir'
```

Typical SVN commands

```
# svn co http://host.url/svn/project1/trunk # Checkout the most recent version
```

```
# Tags and branches are created by copying
```

```
# svn mkdir http://host.url/svn/project1/tags/ # Create the tags directory
```

```
# svn copy -m "Tag rc1 rel." http://host.url/svn/project1/trunk \
```

```
http://host.url/svn/project1/tags/1.0rc1
```

```
# svn status [--verbose] # Check files status into working dir
```

```
# svn add src/file.h src/file.cpp # Add two files
```

```
# svn commit -m 'Added new class file' # Commit the changes with a message
```

```
# svn ls http://host.url/svn/project1/tags/ # List all tags
```

```
# svn move foo.c bar.c # Move (rename) files
```

```
# svn delete some_old_file # Delete files
```

14. USEFUL COMMANDS

14.1 less

The `less` command displays a text document on the console. It is present on most installation.

```
# less unixtoolbox.xhtml
```

Some important commands are (^N stands for [control]-[N]):

h H good help on display

f ^F ^V SPACE Forward one window (or N lines).

b ^B ESC-v Backward one window (or N lines).

F Forward forever; like "tail -f".

/pattern Search forward for (N-th) matching line.

?pattern Search backward for (N-th) matching line.

n Repeat previous search (for N-th occurrence).

N Repeat previous search in reverse direction.

q quit

11.<http://www.cs.put.poznan.pl/csobaniec/Papers/svn-refcard.pdf>

12.<http://tortoisesvn.tigris.org>

14.2 vi

Vi is present on ANY Linux/Unix installation and it is therefore useful to know some basic commands. There are two modes: command mode and insertion mode. The commands mode is accessed with [ESC], the insertion mode with i.

Quit

:w newfilename save the file to newfilename

:wq or :x save and quit

:q! quit without saving

Search and move

/string Search forward for string

?string Search back for string

n Search for next instance of string

N Search for previous instance of string

{ Move a paragraph back

} Move a paragraph forward

1G Move to the first line of the file

nG Move to the n th line of the file

G Move to the last line of the file

:%s/OLD/NEW/g Search and replace every occurrence

Delete text

dd delete current line

D Delete to the end of the line

dw Delete word

x Delete character

u Undo last

U Undo all changes to current line

14.3 mail

The `mail` command is a basic application to read and send email, it is usually installed.

To send

an email simply type "mail user@domain". The first line is the subject, then the mail content.

Terminate and send the email with a single dot (.) in a new line. Example:

```
# mail c@cb.vu
```

```
Subject: Your text is full of typos
```

```
"For a moment, nothing happened. Then, after a second or so,
nothing continued to happen."
```

```
.
```

```
EOT
```

```
#
```

This is also working with a pipe:

```
# echo "This is the mail body" | mail c@cb.vu
```

This is also a simple way to test the mail server.

14.4 tar

The command `tar` (tape archive) creates and extracts archives of file and directories. The archive `.tar` is uncompressed, a compressed archive has the extension `.tgz` or `.tar.gz` (zip) or `.tbz` (bzip2). Do not use absolute path when creating an archive, you probably want to unpack it

Create

```
# cd /
```

```
# tar -cf home.tar home/ # archive the whole /home directory (c for create)
# tar -czf home.tgz home/ # same with zip compression
# tar -cjf home.tbz home/ # same with bzip2 compression
```

Only include one (or two) directories from a tree, but keep the relative structure. For example

archive `/usr/local/etc` and `/usr/local/www` and the first directory in the archive should be `local/`.

```
# tar -C /usr -czf local.tgz local/etc local/www
```

```
# tar -C /usr -xzf local.tgz # To untar the local dir into /usr
# cd /usr; tar -xzf local.tgz # Is the same as above
```

Extract

```
# tar -tzf home.tgz # look inside the archive without extracting (list)
# tar -xf home.tar # extract the archive here (x for extract)
# tar -xzf home.tgz # same with zip compression
# tar -xjf home.tgz # same with bzip2 compression
# tar -xjf home.tgz home/colin/file.txt # Restore a single file
```

More advanced

```
# tar c dir/ | gzip | ssh user@remote 'dd of=dir.tgz' # arch dir/ and store remotely.
# tar cvf - `find . -print` > backup.tar # arch the current directory.
# tar -cf - -C /etc . | tar xpf - -C /backup/etc # Copy directories
# tar -cf - -C /etc . | ssh user@remote tar xpf - -C /backup/etc # Remote copy.
# tar -czf home.tgz --exclude '*.o' --exclude 'tmp/' home/
```

14.5 dd

The program `dd` (disk dump) is used to copy partitions and disks and for other copy tricks.

Typical usage:

```
# dd if=<source> of=<target> bs=<byte size> conv=<conversion>
```

Important conv options:

`notrunc` do not truncate the output file, all zeros will be written as zeros.

`noerror` continue after read errors (e.g. bad blocks)

`sync` pad every input block with Nulls to `ibs-size`

The default byte size is 512 (one block). The MBR, where the partition table is located, is on the first block, the first 63 blocks of a disk are empty. Larger byte sizes are faster to copy but require also more memory.

Backup and restore

```
# dd if=/dev/hda of=/dev/hdc bs=16065b # Copy disk to disk (same size)
# dd if=/dev/sda7 of=/home/root.img bs=4096 conv=notrunc,noerror # Backup /
# dd if=/home/root.img of=/dev/sda7 bs=4096 conv=notrunc,noerror # Restore /
# dd bs=1M if=/dev/ad4s3e | gzip -c > ad4s3e.gz # Zip the backup
# gunzip -dc ad4s3e.gz | dd of=/dev/ad0s3e bs=1M # Restore the zip
# dd bs=1M if=/dev/ad4s3e | gzip | ssh eedcoba@fry 'dd of=ad4s3e.gz' # also remote
# gunzip -dc ad4s3e.gz | ssh eedcoba@host 'dd of=/dev/ad0s3e bs=1M'
# dd if=/dev/ad0 of=/dev/ad2 skip=1 seek=1 bs=4k conv=noerror # Skip MBR
# This is necessary if the destination (ad2) is smaller.
```

Recover

The command `dd` will read every single block of the partition, even the bad blocks. In case of problems it is better to use the option `conv=sync,noerror` so `dd` will skip the bad block and write zeros at the destination. Accordingly it is important to set the block size equal or smaller than the disk block size. A 1k size seems safe, set it with `bs=1k`. If a disk has bad sectors and the data should be recovered from a partition, create an image file with `dd`, mount the image and copy the content to a new disk. With the option `noerror`, `dd` will skip the bad sectors and write zeros instead, thus only the data contained in the bad sectors will be lost.

```
# dd if=/dev/hda of=/dev/null bs=1m # Check for bad blocks
# dd bs=1k if=/dev/hda1 conv=sync,noerror,notrunc | gzip | ssh \ # Send to remote
root@fry 'dd of=hda1.gz bs=1k'
# dd bs=1k if=/dev/hda1 conv=sync,noerror,notrunc of=hda1.img # Store into an
image
# mount -o loop /hda1.img /mnt # Mount the image (page 12)
# rsync -ax /mnt/ /newdisk/ # Copy on a new disk
# dd if=/dev/hda of=/dev/hda # Refresh the magnetic state
```

The above is useful to refresh a disk. It is perfectly safe, but must be unmounted.

Delete

```
# dd if=/dev/zero of=/dev/hdc count=1 # Delete MBR and partition table
# dd if=/dev/zero of=/dev/hdc # Delete full disk
# dd if=/dev/urandom of=/dev/hdc # Delete full disk better
# kill -USR1 PID # View dd progress (Linux only!)
```

14.6 screen

Screen has two main functionalities:

- Run multiple terminal session within a single terminal.
- A started program is decoupled from the real terminal and can thus run in the background. The real terminal can be closed and reattached later.

Short start example

start screen with:

```
# screen
```

Within the screen session we can start a long lasting program (like `top`). Detach the terminal

and reattach the same terminal from an other machine (over `ssh` for example).

```
# top
```

Now detach with `Ctrl-a Ctrl-d`. Reattach the terminal with

```
# screen -r
```

or better:

```
# screen -R -D
```

Attach here and now. In detail this means: If a session is running, then reattach. If necessary detach and logout remotely first. If it was not running create it and notify the user.

Screen commands (within screen)

All screen commands start with `Ctrl-a`.

- `Ctrl-a ?` help and summary of functions
- `Ctrl-a c` create a new window (terminal)
- `Ctrl-a Ctrl-n` and `Ctrl-a Ctrl-p` to switch to the next or previous window in the list, by number.
- `Ctrl-a Ctrl-N` where `N` is a number from 0 to 9, to switch to the corresponding window.
- `Ctrl-a "` to get a navigable list of running windows
- `Ctrl-a a` to clear a missed `Ctrl-a`
- `Ctrl-a Ctrl-d` to disconnect and leave the session running in the background

- Ctrl-a x lock the screen terminal with a password

— Useful Commands —

39

The screen session is terminated when the program within the running terminal is closed and you logout from the terminal.

14.7 Find

Some important options:

```
-x (on BSD) -xdev (on Linux) Stay on the same file system (dev in fstab).
-exec cmd {} \; Execute the command and replace {} with the full path
-iname Like -name but is case insensitive
-ls Display information about the file (like ls -la)
-size n n is +-n (k M G T P)
-cmin n File's status was last changed n minutes ago.
# find . -type f ! -perm -444 # Find files not readable by all
# find . -type d ! -perm -111 # Find dirs not accessible by all
# find /home/user/ -cmin 10 -print # Files created or modified in the last 10 min.
# find . -name '*.ch' | xargs grep -E 'expr' # Search 'expr' in this dir and
below.
# find / -name "*.core" | xargs rm # Find core dumps and delete them
# find / -name "*.core" -print -exec rm {} \; # Other syntax
# find . \( -name "*.png" -o -name "*.jpg" \) -print
# iname is not case sensitive
# find . \( -iname "*.png" -o -iname "*.jpg" \) -print -exec tar -rf images.tar {} \;
# find . -type f -name "*.txt" ! -name README.txt -print # Exclude README.txt
files
# find /var/ -size +1M -exec ls -lh {} \;
# find /var/ -size +1M -ls # This is simpler
# find . -size +10M -size -50M -print
# find /usr/ports/ -name work -type d -print -exec rm -rf {} \; # Clean the ports
Find files with SUID; those file have to be kept secure
# find / -type f -user root -perm -4000 -exec ls -l {} \;
```

14.8 Miscellaneous

```
# which command # Show full path name of command
# time command # See how long a command takes to execute
# time cat # Use time as stopwatch. Ctrl-c to stop
# set | grep $USER # List the current environment
# cal -3 # Display a three month calendar
# date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]
# date 10022155 # Set date and time
# whatis grep # Display a short info on the command or word
# whereis java # Search path and standard directories for word
# setenv varname value # Set env. variable varname to value (csh/tcsh)
# export varname="value" # set env. variable varname to value (sh/ksh/bash)
# pwd # Print working directory
# mkdir -p /path/to/dir # no error if existing, make parent dirs as needed
# rmdir /path/to/dir # Remove directory
# rm -rf /path/to/dir # Remove directory and its content (force)
# cp -la /dir1 /dir2 # Archive and hard link files instead of copy
# cp -lpR /dir1 /dir2 # Same for FreeBSD
# mv /dir1 /dir2 # Rename a directory
```

15. INSTALL SOFTWARE

15.1 List installed packages

```
# rpm -qa # List installed packages (RH, SuSE, RPM based)
# dpkg -l # Debian, Ubuntu
# pkg_info -W smbd # FreeBSD show which package smbd belongs to
# pkginfo # Solaris
```

15.2 Add/remove software

Front ends: `yast2/yast` for SuSE, `redhat-config-packages` for Red Hat.

```
# rpm -i pkgname.rpm # install the package (RH, SuSE, RPM based)
# rpm -e pkgname # Remove package
```

Debian

```
# apt-get update # First update the package lists
# apt-get install emacs # Install the package emacs
# dpkg --remove emacs # Remove the package emacs
```

FreeBSD

```
# pkg_add -r rsync # Fetch and install rsync.
# pkg_delete /var/db/pkg/rsync-xx # Delete the rsync package
```

Set where the packages are fetched from with the `PACKAGESITE` variable. For example:

```
# export PACKAGESITE=ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages/Latest/
# or ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6-stable/Latest/
```

FreeBSD ports. The port tree `/usr/ports/` is a collection of software ready to compile and install. The ports are updated with the program `portsnap`.

```
# portsnap fetch extract # Create the tree when running the first time
# portsnap fetch update # Update the port tree
# cd /usr/ports/net/rsync/ # Select the package to install
# make install distclean # Install and cleanup (also see man ports)
# make package # Make a binary package for the port
```

15.3 Library path

Due to complex dependencies and runtime linking, programs are difficult to copy to an other system or distribution. However for small programs with little dependencies, the missing libraries can be copied over. The runtime libraries (and the missing one) are checked with `ldd` and managed with `ldconfig`.

```
# ldd /usr/bin/rsync # List all needed runtime libraries
# ldconfig -n /path/to/libs/ # Add a path to the shared libraries directories
# ldconfig -m /path/to/libs/ # FreeBSD
# LD_LIBRARY_PATH # The variable set the link library path
```

16. CONVERT MEDIA

Sometimes one simply need to convert a video, audio file or document to another format.

16.1 Text encoding

Text encoding can get totally wrong, specially when the language requires special characters like `аѓз`. The command `iconv` can convert from one encoding to an other.

```
# iconv -f <from_encoding> -t <to_encoding> <input_file>
# iconv -f ISO8859-1 -t UTF-8 -o file.input > file_utf8
# iconv -l # List known coded character sets
```

Without the `-f` option, `iconv` will use the local char-set, which is usually fine if the document displays well.

16.2 Unix - DOS newlines

Convert DOS (CR/LF) to Unix (LF) newlines within a Unix shell. See also `dos2unix` and `unix2dos`

if you have them.

```
# sed 's/.$//' dosfile.txt > unixfile.txt
```

Convert Unix to DOS newlines within a Windows environment. Use `sed` from mingw or cygwin.

```
# sed -n p unixfile.txt > dosfile.txt
```

16.3 PDF to Jpeg and concatenate PDF files

Convert a PDF document with `gs` (GhostScript) to jpeg (or png) images for each page. Also

much shorter with `convert` (from ImageMagick or GraphicsMagick).

```
# gs -dBATCH -dNOPAUSE -sDEVICE=jpeg -r150 -dTextAlphaBits=4 \
-dGraphicsAlphaBits=4 \
-dMaxStripSize=8192 -sOutputFile=unixtoolbox_%d.jpg unixtoolbox.pdf
```

```
# convert unixtoolbox.pdf unixtoolbox-%03d.png
```

```
# convert *.jpeg images.pdf # Create a simple PDF with all pictures
```

Ghostscript can also concatenate multiple pdf files into a single one.

```
# gs -q -sPAPERSIZE=a4 -dNOPAUSE -dBATCH -sDEVICE=pdfwrite \
-sOutputFile=all.pdf \
```

```
file1.pdf file2.pdf ... # On Windows use '#' instead of '='
```

16.4 Convert video

Compress the Canon digicam video with an mpeg4 codec and repair the crappy sound.

```
# mencoder -o videoout.avi -oac mp3lame -ovc lavc -srate 11025 \
-channels 1 -af-adv force=1 -lameopts preset=medium -lavcopts \
vcodec=msmpeg4v2:vbitrate=600 -mc 0 vidoein.AVI
```

16.5 Copy an audio cd

The program `cdparanoia13` can save the audio tracks (FreeBSD port in `audio/cdparanoia/`),

`oggenc` can encode in Ogg Vorbis format, `lame` converts to mp3.

```
# cdparanoia -B # Copy the tracks to wav files in current dir
```

```
# lame -b 256 in.wav out.mp3 # Encode in mp3 256 kb/s
```

```
# for i in *.wav; do lame -b 256 $i `basename $i .wav`.mp3; done
```

```
# oggenc in.wav -b 256 out.ogg # Encode in Ogg Vorbis 256 kb/s
```

13.<http://xiph.org/paranoia/>

17. PRINTING

17.1 Print with lpr

```
# lpr unixtoolbox.ps # Print on default printer
```

```
# export PRINTER=hp4600 # Change the default printer
```

```
# lpr -Php4500 #2 unixtoolbox.ps # Use printer hp4500 and print 2 copies
```

```
# lpr -o Duplex=DuplexNoTumble ... # Print duplex along the long side
# lpr -o PageSize=A4,Duplex=DuplexNoTumble ...
# lpq # Check the queue on default printer
# lpq -l -Php4500 # Queue on printer hp4500 with verbose
# lprm - # Remove all users jobs on default printer
# lprm -Php4500 3186 # Remove job 3186. Find job nbr with lpq
# lpc status # List all available printers
# lpc status hp4500 # Check if printer is online and queue length
```

18. DATABASES

18.1 PostgreSQL

Change root or a username password

```
# psql -d template1 -U pgsql
```

```
> alter user pgsql with password 'pgsql_password'; # Use username instead of
"pgsql"
```

Create user and database

The commands `createuser`, `dropuser`, `createdb` and `dropdb` are convenient shortcuts equivalent to the SQL commands. The new user is bob with database bobdb ; use as root with

pgsql the database super user:

```
# createuser -U pgsql -P bob # -P will ask for password
# createdb -U pgsql -O bob bobdb # new bobdb is owned by bob
# dropdb bobdb # Delete database bobdb
# dropuser bob # Delete user bob
```

The general database authentication mechanism is configured in `pg_hba.conf`

Grant remote access

The file `$PGSQL_DATA_D/postgresql.conf` specifies the address to bind to. Typically `listen_addresses = '*'` for Postgres 8.x.

The file `$PGSQL_DATA_D/pg_hba.conf` defines the access control. Examples:

```
# TYPE DATABASE USER IP-ADDRESS IP-MASK METHOD
```

```
host bobdb bob 212.117.81.42 255.255.255.255 password
```

```
host all all 0.0.0.0/0 password
```

Backup and restore

The backups and restore are done with the user pgsql or postgres. Backup and restore a single database:

```
# pg_dump --clean dbname > dbname_sql.dump
```

```
# psql dbname < dbname_sql.dump
```

Backup and restore all databases (including users):

```
# pg_dumpall --clean > full.dump
```

```
# psql -f full.dump postgres
```

In this case the restore is started with the database postgres which is better when reloading an empty cluster.

18.2 MySQL

Change mysql root or username password

Method 1

```
# /etc/init.d/mysql stop
```

or

```
# killall mysqld
```

```
# mysqld --skip-grant-tables
# mysqladmin -u root password 'newpasswd'
# /etc/init.d/mysql start
```

Method 2

```
# mysql -u root mysql
mysql> UPDATE USER SET PASSWORD=PASSWORD("newpassword") where user='root';
mysql> FLUSH PRIVILEGES; # Use username instead of "root"
mysql> quit
```

Create user and database

```
# mysql -u root mysql
mysql> CREATE DATABASE bobdb;
mysql> GRANT ALL ON *.* TO 'bob'@'%' IDENTIFIED BY 'pwd'; # Use localhost instead
of %
```

to restrict the network access

```
mysql> DROP DATABASE bobdb; # Delete database
mysql> DROP USER bob; # Delete user
mysql> DELETE FROM mysql.user WHERE user='bob and host='hostname'; # Alt. command
mysql> FLUSH PRIVILEGES;
```

Grant remote access

Remote access is typically permitted for a database, and not all databases. The file `/etc/my.cnf`

contains the IP address to bind to. Typically comment the line `bind-address = out.`

mysql -u root mysql

```
mysql> GRANT ALL ON bobdb.* TO bob@'xxx.xxx.xxx.xxx' IDENTIFIED BY 'PASSWORD';
mysql> REVOKE GRANT OPTION ON foo.* FROM bar@'xxx.xxx.xxx.xxx';
mysql> FLUSH PRIVILEGES; # Use 'hostname' or also '%' for full access
```

Backup and restore

Backup and restore a single database:

```
# mysqldump -u root -psecret --add-drop-database dbname > dbname_sql.dump
# mysql -u root -psecret -D dbname < dbname_sql.dump
```

Backup and restore all databases:

```
# mysqldump -u root -psecret --add-drop-database --all-databases > full.dump
# mysql -u root -psecret < full.dump
```

Here is "secret" the mysql root password, there is no space after -p. When the -p option is used

alone (w/o password), the password is asked at the command prompt.

18.1 SQLite

SQLite¹⁴ is a small powerfull self-contained, serverless, zero-configuration SQL database.

14.<http://www.sqlite.org>

— Databases —

44

Dump and restore

It can be useful to dump and restore an SQLite database. For example you can edit the dump

file to change a column attribute or type and then restore the database. This is easier than

messing with SQL commands. Use the command `sqlite3` for a 3.x database.

```
# sqlite database.db .dump > dump.sql # dump
# sqlite database.db < dump.sql # restore
```

Convert 2.x to 3.x database

```
sqlite database_v2.db .dump | sqlite3 database_v3.db
```


19. DISK QUOTA

A disk quota allows to limit the amount of disk space and/or the number of files a user or (or member of group) can use. The quotas are allocated on a per-file system basis and are enforced by the kernel.

19.1 Linux setup

The quota tools package usually needs to be installed, it contains the command line tools. Activate the user quota in the fstab and remount the partition. If the partition is busy, either all locked files must be closed, or the system must be rebooted. Add `usrquota` to the fstab mount options, for example:

```
/dev/sda2 /home reiserfs rw,acl,user_xattr,usrquota 1 1
```

```
# mount -o remount /home
```

```
# mount # Check if usrquota is active, otherwise reboot
```

Initialize the quota.user file with `quotacheck`.

```
# quotacheck -vum /home
```

```
# chmod 644 /home/aquota.user # To let the users check their own quota
```

Activate the quota either with the provided script (e.g. `/etc/init.d/quotad` on SuSE) or with

```
quotaon:
```

```
quotaon -vu /home
```

Check that the quota is active with:

```
quota -v
```

19.2 FreeBSD setup

The quota tools are part of the base system, however the kernel needs the option `quota`. If it is

not there, add it and `recompile` the kernel.

options `QUOTA`

As with Linux, add the quota to the fstab options (userquota, not usrquota):

```
/dev/ad0s1d /home ufs rw,noatime,userquota 2 2
```

```
# mount /home # To remount the partition
```

Enable disk quotas in `/etc/rc.conf` and start the quota.

```
# grep quotas /etc/rc.conf
```

```
enable_quotas="YES" # turn on quotas on startup (or NO).
```

```
check_quotas="YES" # Check quotas on startup (or NO).
```

```
# /etc/rc.d/quota start
```

19.3 Assign quota limits

The quotas are not limited per default (set to 0). The limits are set with `edquota` for single

users. A quota can be also duplicated to many users. The file structure is different between the

quota implementations, but the principle is the same: the values of blocks and inodes can be

limited. Only change the values of soft and hard. If not specified, the blocks are 1k. The grace

period is set with `edquota -t`. For example:

```
# edquota -u colin
```

Linux

Disk quotas for user colin (uid 1007):

Filesystem blocks soft hard inodes soft hard

/dev/sda8 108 1000 2000 1 0 0

FreeBSD

Quotas for user colin:

/home: kbytes in use: 504184, limits (soft = 700000, hard = 800000)

inodes in use: 1792, limits (soft = 0, hard = 0)

For many users

The command `edquota -p` is used to duplicate a quota to other users. For example to duplicate

a reference quota to all users:

```
# edquota -p refuser `awk -F: ' $3 > 499 {print $1}' /etc/passwd`
```

```
# edquota -p refuser user1 user2 # Duplicate to 2 users
```

Checks

Users can check their quota by simply typing `quota` (the file `quota.user` must be readable). Root

can check all quotas.

```
# quota -u colin # Check quota for a user
```

```
# repquota /home # Full report for the partition for all users
```

20. SHELLS

Most Linux distributions use the bash shell while the BSDs use tcsh, the bourne shell is only

used for scripts. Filters are very useful and can be piped:

`grep` Pattern matching

`sed` Search and Replace strings or characters

`cut` Print specific columns from a marker

`sort` Sort alphabetically or numerically

`uniq` Remove duplicate lines from a file

For example used all at once:

```
# ifconfig | sed 's/ /g' | cut -d" " -f1 | uniq | grep -E "[a-z0-9]+" | sort -r
```

```
# ifconfig | sed '/.*inet addr:!/d;s///s/ .*//'|sort -t. -k1,1n -k2,2n -k3,3n -k4,4n
```

The first character in the sed pattern is a tab. To write a tab on the console, use `ctrl-v` `ctrl-tab`.

20.1 bash

Redirects and pipes for bash and sh:

```
# cmd 1> file # Redirect stdout to file.
```

```
# cmd 2> file # Redirect stderr to file.
```

```
# cmd 1>> file # Redirect and append stdout to file.
```

```
# cmd &> file # Redirect both stdout and stderr to file.
```

```
# cmd >file 2>&1 # Redirects stderr to stdout and then to file.
```

```
# cmd1 | cmd2 # pipe stdout to cmd2
```

```
# cmd1 2>&1 | cmd2 # pipe stdout and stderr to cmd2
```

Modify your configuration in `~/.bashrc` (it can also be `~/.bash_profile`). The following entries are useful, reload with `". .bashrc"`.

in .bashrc

```
bind '"\e[A":history-search-backward # Use up and down arrow to search
```

```
bind '"\e[B":history-search-forward # the history. Invaluable!
```

```
set -o emacs # Set emacs mode in bash (see below)
```

```
set bell-style visible # Do not beep, inverse colors
```

```
# Set a nice prompt like [user@host]/path/todir>
PS1="\[\033[1;30m\]\[\033[1;34m\]\u\[\033[1;30m\]"
PS1="$PS1@\[\033[0;33m\]\h\[\033[1;30m\]\[\033[0;37m\]"
PS1="$PS1\w\[\033[1;30m\]>\[\033[0m\]"
# To check the currently active aliases, simply type alias
alias ls='ls -aF' # Append indicator (one of */=>@|)
alias ll='ls -aFls' # Listing
alias la='ls -all'
alias ..='cd ..'
alias ...='cd ../../'
export HISTFILESIZE=5000 # Larger history
export CLICOLOR=1 # Use colors (if possible)
export LSCOLORS=ExGxFxdxCxDxBxBxExEx
```

20.2 tcsh

Redirects and pipes for tcsh and csh (simple > and >> are the same as sh):

```
# cmd >& file # Redirect both stdout and stderr to file.
# cmd >>& file # Append both stdout and stderr to file.
# cmd1 | cmd2 # pipe stdout to cmd2
# cmd1 |& cmd2 # pipe stdout and stderr to cmd2
```

The settings for csh/tcsh are set in ~/.cshrc, reload with "source .cshrc". Examples:

```
# in .cshrc
alias ls 'ls -aF'
alias ll 'ls -aFls'
alias la 'ls -all'
alias .. 'cd ..'
alias ... 'cd ../../'
set prompt = "%B%n%b@%B%m%b%> " # like user@host/path/todir>
set history = 5000
set savehist = ( 6000 merge )
set autolist # Report possible completions with tab
set visiblebell # Do not beep, inverse colors
# Bindkey and colors
bindkey -e Select Emacs bindings # Use emacs keys to edit the command prompt
bindkey -k up history-search-backward # Use up and down arrow to search
bindkey -k down history-search-forward
setenv CLICOLOR 1 # Use colors (if possible)
setenv LSCOLORS ExGxFxdxCxDxBxBxExEx
```

The emacs mode enables to use the emacs keys shortcuts to modify the command prompt line.

This is extremely useful (not only for emacs users). The most used commands are:

C-a Move cursor to beginning of line

C-e Move cursor to end of line

— Shells —

47

M-b Move cursor back one word

M-f Move cursor forward one word

M-d Cut the next word

C-w Cut the last word

C-u Cut everything before the cursor

C-k Cut everything after the cursor (rest of the line)

C-y Paste the last thing to be cut (simply paste)

C-_ Undo

Note: C- = hold control, M- = hold meta (which is usually the alt or escape key).

21. SCRIPTING

The Bourne shell (/bin/sh) is present on all Unix installations and scripts written in this language are (quite) portable; `man 1 sh` is a good reference.

21.1 Basics

Variables and arguments

Assign with `variable=value` and get content with `$variable`

```
MESSAGE="Hello World" # Assign a string
```

```
PI=3.1415 # Assign a decimal number
```

```
N=8
```

```
TWON=`expr $N * 2` # Arithmetic expression (only integers)
```

```
TWON=$(( $N * 2 )) # Other syntax
```

```
TWOPI=`echo "$PI * 2" | bc -l` # Use bc for floating point operations
```

```
ZERO=`echo "c($PI/4)-sqrt(2)/2" | bc -l`
```

The command line arguments are

```
$0, $1, $2, ... # $0 is the command itself
```

```
$# # The number of arguments
```

```
$* # All arguments (also $@)
```

Special Variables

```
$$ # The current process ID
```

```
$? # exit status of last command
```

command

```
if [ $? != 0 ]; then
```

```
echo "command failed"
```

```
fi
```

```
mypath=`pwd`
```

```
mypath=${mypath}/file.txt
```

```
echo ${mypath##*/} # Display the filename only
```

```
echo ${mypath%*.} # Full path without extension
```

```
var2=${var:=string} # Use var if set, otherwise use string
```

assign string to var and then to var2.

Constructs

```
for file in `ls`
```

```
do
```

```
echo $file
```

```
done
```

```
count=0
```

```
while [ $count -lt 5 ]; do
```

```
echo $count
```

```
sleep 1
```

```
count=$(( $count + 1 ))
```

```
done
```

```
myfunction() {
```

```
find . -type f -name "*.1" -print # $1 is first argument of the function
```

```
}
```

```
myfunction "txt"
```

Generate a file

```
MYHOME=/home/colin
```

```
cat > testhome.sh << _EOF
```

```
# All of this goes into the file testhome.sh
```

```
if [ -d "$MYHOME" ] ; then
echo $MYHOME exists
else
echo $MYHOME does not exist
fi
_EOF
sh testhome.sh
```

21.2 Bourne script example

As a small example, the script used to create a PDF booklet from this xhtml document:

```
#!/bin/sh
# This script creates a book in pdf format ready to print on a duplex printer
if [ $# -ne 1 ]; then # Check the argument
echo 1>&2 "Usage: $0 HtmlFile"
exit 1 # non zero exit if error
fi
file=$1 # Assign the filename
fname=${file%.*} # Get the name of the file only
fext=${file#*.} # Get the extension of the file
prince $file -o $fname.pdf # from www.princexml.com
pdftops -paper A4 -noshrink $fname.pdf $fname.ps # create postscript booklet
cat $fname.ps |psbook|psnup -Pa4 -2 |pstops -b "2:0,1U(21cm,29.7cm)" >
$fname.book.ps
ps2pdf13 -sPAPERSIZE=a4 -sAutoRotatePages=None $fname.book.ps $fname.book.pdf
# use #a4 and #None on Windows!
exit 0 # exit 0 means successful
```

21.3 Some sed commands

```
sed 's/string1/string2/g' # Replace string1 with string2
sed -i 's/wroong/wrong/g' *.txt # Replace a recurring word with g
sed 's/\(.*\)1/\2/g' # Modify anystring1 to anystring2
sed '/<p>/,/<\p>/d' t.xhtml # Delete lines that start with <p>
# and end with </p>
sed '/ *#/d; /^ */d' # Remove comments and blank lines
sed 's/[ \t]*$//' # Remove trailing spaces (use tab as \t)
sed 's/^[ \t]*//;s/[ \t]*$//' # Remove leading and trailing spaces
sed 's/[^\*]/[&]/' # Enclose first char with [] top->[t]op
```

21.4 Some useful commands

```
sort -t. -k1,1n -k2,2n -k3,3n -k4,4n # Sort IPv4 ip addresses
echo 'Test' | tr '[:lower:]' '[:upper:]' # Case conversion
echo foo.bar | cut -d . -f 1 # Returns foo
PID=$(ps | grep script.sh | grep bin | awk '{print $1}') # PID of a running script
PID=$(ps axww | grep [p]ing | awk '{print $1}') # PID of ping (w/o grep pid)
IP=$(ifconfig $INTERFACE | sed '/.*inet addr:!/d;s///;s/ .*//') # Linux
IP=$(ifconfig $INTERFACE | sed '/.*inet /!d;s///;s/ .*//') # FreeBSD
if [ `diff file1 file2 | wc -l` != 0 ]; then [...] fi # File changed?
cat /etc/master.passwd | grep -v root | grep -v \*: | awk -F":" ' \ # Create http
passwd
'{ printf("%s:%s\n", $1, $2) }' > /usr/local/etc/apache2/passwd
testuser=$(cat /usr/local/etc/apache2/passwd | grep -v \ # Check user in passwd
root | grep -v \*: | awk -F":" '{ printf("%s\n", $1) }' | grep ^user$)
```

22. PROGRAMMING

22.1 C basics

```
strcpy(newstr, str) /* copy str to newstr */
expr1 ? expr2 : expr3 /* if (expr1) expr2 else expr3 */
x = (y > z) ? y : z; /* if (y > z) x = y; else x = z; */
int a[]={0,1,2}; /* Initialized array (or a[3]={0,1,2}; */
int a[2][3]={{{1,2,3},{4,5,6}}}; /* Array of array of ints */
int i = 12345; /* Convert in i to char str */
char str[10];
sprintf(str, "%d", i);
```

22.2 C example

A minimal c program simple.c:

```
#include <stdio.h>
main() {
    int number=42;
    printf("The answer is %i\n", number);
}
```

Compile with:

```
# gcc simple.c -o simple
```

```
# ./simple
```

The answer is 42

22.3 C++ basics

```
*pointer // Object pointed to by pointer
&obj // Address of object obj
obj.x // Member x of class obj (object obj)
pobj->x // Member x of class pointed to by pobj
// (*pobj).x and pobj->x are the same
```

22.4 C++ example

As a slightly more realistic program in C++, let's create a class in its own header (IPv4.h) and

implementation (IPv4.cpp) and create a program which uses the class functionality. The class has a member to convert an IP address in integer format to the known quad format. This is a minimal c++ program with a class and multi-source compile.

IPv4 class

IPv4.h:

```
#ifndef IPV4_H
#define IPV4_H
#include <string>
namespace GenericUtils { // create a namespace
class IPv4 { // class definition
public:
    IPv4();
    ~IPv4();
    std::string IPint_to_IPquad(unsigned long ip); // member interface
};
} //namespace GenericUtils
#endif // IPV4_H
```

IPv4.cpp:

```
#include "IPv4.h"
#include <string>
#include <sstream>
using namespace std; // use the namespaces
using namespace GenericUtils;
IPv4::IPv4() {} // default constructor/destructor
IPv4::~IPv4() {}
string IPv4::IPint_to_IPquad(unsigned long ip) { // member implementation
ostringstream ipstr; // use a stringstream
ipstr << ((ip &0xff000000) >> 24) // Bitwise right shift
<< "." << ((ip &0x00ff0000) >> 16)
<< "." << ((ip &0x0000ff00) >> 8)
<< "." << ((ip &0x000000ff));
return ipstr.str();
}
```

The program simplecpp.cpp

```
#include "IPv4.h"
#include <iostream>
#include <string>
using namespace std;
int main (int argc, char* argv[]) {
string ipstr; // define variables
unsigned long ipint = 1347861486; // The IP in integer form
GenericUtils::IPv4 iputils; // create an object of the class
ipstr = iputils.IPint_to_IPquad(ipint); // call the class member
cout << ipint << " = " << ipstr << endl; // print the result
return 0;
}
```

Compile and execute with:

```
# g++ -c IPv4.cpp simplecpp.cpp # Compile in objects
# g++ IPv4.o simplecpp.o -o simplecpp.exe # Link the objects to final executable
```

./simplecpp.exe

1347861486 = 80.86.187.238

Use `ldd` to check which libraries are used by the executable and where they are located. This command is also used to check if a shared library is missing or if the executable is static.

```
# ldd /sbin/ifconfig
```

22.5 Simple Makefile

The corresponding minimal Makefile for the multi-source program is shown below. The lines with

instructions must begin with a tab! The back slash "\" can be used to cut long lines.

```
CC = g++
CFLAGS = -O
OBJS = IPv4.o simplecpp.o
simplecpp: ${OBJS}
${CC} -o simplecpp ${CFLAGS} ${OBJS}
clean:
rm -f ${TARGET} ${OBJS}
```

23. ONLINE HELP

23.1 Documentation

Linux Documentation en.tldp.org
Linux Man Pages www.linuxmanpages.com
Linux commands directory www.oreillynet.com/linux/cmd
Linux doc man howtos linux.die.net
FreeBSD Handbook www.freebsd.org/handbook
FreeBSD Man Pages www.freebsd.org/cgi/man.cgi
FreeBSD user wiki www.freebsdwiki.net
Solaris Man Pages docs.sun.com/app/docs/coll/40.10

23.2 Other Unix/Linux references

Rosetta Stone for Unix bhami.com/rosetta.html (a Unix command translator)
Unix guide cross reference unixguide.net/unixguide.shtml
Linux commands line list www.linuxguide.it/commands_list.php
Short Linux reference www.pixelbeat.org/cmdline.html
That's all folks!
This document: "Unix Toolbox revision 11" is licensed under a [Creative Commons](http://creativecommons.org/licenses/by/4.0/)
Licence

Полезные утилиты Gentoo Linux

- Пакет **gentoolkit** (для установки наберите: **emerge gentoolkit**) содержит:
 - а) **euse**, которая существенно упрощает управление флагами USE. Выполните **euse -i flag** для получения информации о флаге "flag". Так же просмотрите man-страницу euse (не волнуйтесь. она короткая :-)).
 - б) **eclean**. Она поможет удалить не нужные файлы. Эта утилита может работать как с исходниками, так и с бинарными пакетами. (просто вызовите её как **eclean-dist** или **eclean-pkg** соответственно). Прочитайте страницу man для получения полного списка опций.
- Утилита **genlop** (для установки наберите: **emerge genlop**) позволяет извлечь множество различной информации из этого файла. Например genlop -с покажет вам какой пакет собирается в данный момент и попытается предсказать сколько времени осталось до конца сборки.
- Утилита **ufed** (для установки наберите: **emerge ufed**) позволяет просматривать и устанавливать USE флаги в более дружелюбном интерфейсе.
- Скрипт **update-world** (ссылка: <http://www.gentoo.org/news/ru/gwn/20061204-newsletter.xml> Пункт 3. Полезные советы / Простейший путь обновления своей системы) позволяет оставлять обновляющуюся систему не боясь, что обновление прервется на каком-либо пакете. Данный скрипт просто продолжит обновление со следующего пакета.
- Оконные приложения:
 - **kuroo** (написана на **Qt**, установка: **emerge kuroo**);
 - **porthole** (написана на **GTK**, установка: **emerge porthole**);
 - **portagemaster** (написана на **Java**, установка: **emerge portagemaster**);

предлагают доступ к **portage** в оконном режиме. Выбирайте на свой вкус! :)
- Общая Linux утилита **lshw – hardware list (emerge lshw)** показывает информацию о аппаратном обеспечении компьютера.
- Если необходимо заново сгенерировать файл world используйте команду **regenworld**.
- Если необходимо пересобрать модули ядра после компиляции нового ядра, то воспользуйтесь утилитой **module-rebuild (emerge module-rebuild)**

Ресурсы с документацией

- <http://gentoo.ru>
- <http://gentoo.org>
- <http://ru.gentoo-wiki.com>
- <http://gentoo.blog.ru>
- <http://www.rugentoo.org>
- <http://linuxforum.ru>
- <http://myfotomx.com/dalth/linuxbook.odt>
- <http://linuxshop.ru>
- <http://citkit.ru>
- <http://rus-linux.net>
- <http://lxj.endofinternet.net>
- <http://www.habrahabr.ru/blog/linux>
- <http://perlovka.homelinux.net>

Ресурсы с программным обеспечением

1. <http://packages.gentoo.org>
2. <http://gentoo-portage.com>

Каналы IRC (freenode.net)

1. *#gentoo.*
2. *#gentoo-amd64*
3. *#gentoo-kde*
4. *#gentoo-ru.*
5. *#rugentoo.*
6. *#gentoo-doc-ru*
7. *#gentoo-laptop*