Секреты Мастера

Использование антивирусов AVP и Dr.Web под Linux

Игровой сервер для компьютерного клуба. Запуск Windows-игр под Linux

Система обнаружения и защиты от атак LIDS. Сканер портов SATAN

Настройка и перекомпилирование ядра MRTG-программа подсчета трафика

Настройка "обратного звонка" Связка Apache+PHP+MySQL Настройка массивов RAID

Д.Н. Колисниченко

Linux сервер своими руками

Все типы серверов от сервера локальной сети до Интернет-сервера и сервера провайдера на основе дистрибутивов Red Hat и Mandrake.

CD прилагается

НиТ

Д.Н. Колисниченко

Linux-сервер
своими руками

Наука и Техника
Санкт-Петербург
2002

Колисниченко Д.Н. Linux-сервер своими руками. — СПб: Наука и Техника, 2002. — 576 стр. с ил. Прилагается CD.

Под редакцией Финкова М.В.

ISBN 5-94387-063-6

Серия «Секреты мастерства»

В книге подробно рассмотрены настройки сетевых сервисов, позволяющих создать сервер требуемой конфигурации и функциональности на основе ОС Linux. Вы сможете настроить сервер любого типа: от сервера локальной сети до Интернет-сервера и сервера удаленного доступа. Детально описано администрирование Linux.

Изложение материала построено на основе дистрибутивов Red Hat и Mandrake. Много уникальной информации: запуск Windows-игр под Linux и создание Linux-сервера для игрового зала, настройка антивирусов Dr. Web и AVP под Linux, программа учета трафика MRTG, система защиты и обнаружения атак LIDS, а также многое другое. Особое внимание уделено безопасности Linux-серверов. Достаточно подробно описана сама ОС Linux и приведен справочник ее команд. Прочитав книгу, вы станете обладателями знаний по настройке и компилированию ядра, созданию собственных rpm-пакетов, командному интерпретатору bash, использованию массивов RAID. Вы узнаете внутренний мир Linux. Книга подойдет как для профессиональных, так и для начинающих администраторов, поскольку изложение материала начинается с установки ОС Linux, а в первой главе дано описание основных сетевых технологий и протоколов (Курс Молодого Администратора).

Все приведенные в книге листинги проверены на практике и размещены на прилагаемом CD. Помимо этого на нем содержится много справочной информации (HOWTO, RFC), а также статей, посвященных Linux. Размещен богатый набор вспомогательных утилит и программного обеспечения для сервера (Apache, MySQL, MRTG и др.).

ISBN 5-94387-063-6

Контактные телефоны издательства
(812) 567-70-25, 567-70-26
(044) 516-38-66, 518-56-47

www.pubnit.com
Интернет-магазин www.nit.com.ru

ООО «Наука и Техника».
Лицензия №000350 от 23 декабря 1999 года.
198097, г. Санкт-Петербург, ул. Маршала Говорова, д. 29.
Подписано в печать 08.10.02. Формат 70×100 1/16.
Бумага газетная. Печать офсетная. Объем 36 п. л.
Тираж 3000 экз. Заказ № 851

Отпечатано с готовых диапозитивов в ФГУП ордена Трудового Красного Знамени «Техническая книга» Министерства Российской Федерации по делам печати, телерадиовещания и средств массовых коммуникаций.
198005, Санкт-Петербург, Измайловский пр., 29.

Автор благодарит Марка Финкова, редактора и самого внимательного читателя этой книги, за его советы относительно содержания книги (например, описать программу MRTG в книге посоветовал именно Марк), а также исправление множества всевозможных опечаток в тексте и листингах. Особая благодарность Федору Сорексу, редактору раздела Unix-систем портала СофтТерра и руководителю проекта Linux RSP за предложение в 2000 году написать мою первую статью. Главная благодарность Линусу Торвальдсу за создание замечательной операционной системы и всем тем, кто участвует в процессе создания и сопровождения данной системы, а также всем моим родным за проявленное ими терпение во время работы над книгой.

Содержание

1

Введение в Linux

1.1. Вступительное слово

Цель данной книги состоит в том, чтобы развеять миф о «неподъемности» UNIX или Linux-систем, заложить прочный фундамент для построения вашего Интернет-сервера. Излагая материал, я старался объяснять все как можно более понятным языком. Все примеры, приведенные в книге, являются на 100% рабочими. Поэтому эта книга окажется полезной тем, кому нужно в сжатые сроки освоить Linux или настроить Linux-сервер. Я следовал одному простому правилу: не излагать те факты, которые заведомо окажутся невостребованными. Я наоборот, я старался приводить как можно больше примеров из реальной жизни, а в гл. 17 я полностью описал настройку шлюза и сервера для входящих звонков, а также довольно интересную вещь — настройку «обратного звонка». Вам может показаться, что сведения в этой главе повторяются, но с ее помощью читатель сможет настроить шлюз и сервер для входящих звонков «с нуля», не читая предыдущих глав, при условии, что он обладает хоть какими-нибудь навыками работы в ОС Linux.

Данная книга предназначена как для начинающих, так и для опытных системных администраторов. В книге рассматриваются все этапы настройки Linux-сервера, особое внимание уделено вопросам безопасности. Читатель также найдет подробное описание настройки сетевых служб, включая DNS, HTTP, FTP, IpChains, SQUID. В книге собраны самые актуальные, на мой взгляд, темы.

1.2. О Linux

История операционной системы Linux началась, естественно, с создания операционной системы Unix. В конце 60-х годов завершился проект **Multics,** над которым работали сотрудники компаний General Electrics, AT&T Bell Laboratories и Массачусетского института. Результатом этого проекта стала одноименная операционная система. Операционная система Multics была многозадачной, имела высокоэффективную на то время файловую систему, а также предоставляла пользователям относительно удобный интерфейс.

В 1969 году Кен Томпсон разработал операционную систему **Unix,** в основе которой были принципы, которых придерживались разработчики операционной системы Multics. Новая операционная система, в отличие от Multics, могла работать на мини-ЭВМ. При этом с самого начала новая система была многозадачной и многопользовательской.

Вскоре операционная система Unix стала настолько популярной, что Кен Томпсон и Деннис Ритчи решили переписать код системы на языке С. До этого операционная система была написана на ассемблере. Это обеспечило огромною мобильность операционной системы — ОС Unix могла быть перенесена практически на любую платформу без перепрограммирования. Нужно было только модифицировать небольшую часть ядра, написанную на ассемблере.

Через определенное время операционная система Unix стала стандартным программным продуктом, который распространялся многими компаниями, включая IBM и Novell.

В 1972 году началась массовая продажа лицензий на эту операционную систему различным пользователям. С этого момента ОС Unix неофициально стала коммерческим программным продуктом.

Калифорнийский университет в Беркли также приобрел лицензию на ОС Unix. Специалисты этого университета внесли много изменений, которые вскоре стали стандартными. В 1975 году Калифорнийский университет выпустил свою версию Unix -- Berkeley Software Distribution **(BSD).** Эта операционная система стала основным конкурентом операционной системы, разработанной компанией AT&T.

Постепенно другие компании, последовав примеру Калифорнийского университета, начали выпускать свои версии Unix. Например, в 1980 году компания Microsoft выпустила ОС **Xenix.** Правда, данная операционная система не могла составить конкуренцию Unix, так как не поддерживала многопользовательский режим, а была предназначена для одного пользователя.

В 1982 году компания AT&T выпустила версию **Unix System 3.** Это была первая официальная коммерческая версия ОС Unix. Следующей версией стала **Unix System V.** Помимо различных нововведений, эта версия отличалась серьезной технической поддержкой.

Разработчики BSD Unix также не сидели сложа руки и в 1983 году Калифорнийский университет выпустил версию **Unix BSD 4.2.** Эта операционная система содержала довольно мощные средства управления памятью, файлами, печатью, а также в ней был реализован протокол TCP/IP, который сейчас во всю применяется в сети Интернет. Многие фирмы-производители выбрали именно Unix BSD 4.2.

Широкое распространение различных версий Unix привело к необходимости создания стандарта на эту операционную систему. В середине 80-х годов выделились два основных стандарта — Unix System V и BSD Unix. Компания AT&T Labs передала права на разработку версии Unix System V компании Unix System Labs. Эта компания в 1991 году представила операционную систему **System** V 4, в которой были реализованы практически все возможности операционных систем System V 3, BSD 4.2, Xenix.

Четыре компании, в том числе IBM и Hewlett-Packard, создали фонд открытого программного обеспечения (Open Software Foundation, OSF). Целью этого фонда было создание собственной версии Unix. В результате появился еще один стандарт — **OSF Unix.**

В 1993 году компания AT&T продала свою часть прав на ОС Unix компании Novell. После этого были выпущены версии Unix компании Novell, которые были основаны на версии System V 4. Новая операционная система была названа **UnixWare.**

Параллельно с развитием операционных систем развивались графические интерфейсы. До начала 90-х годов выделились два основных графических интерфейса пользователя: **Motiff и OpenLook.** Впоследствии эти два интерфейса были определены в один, получивший название **Common Desktop Environment (CDE).**

Операционная система Unix с самого начала была требовательна к аппаратным ресурсам компьютера. Для нормальной работы нужна была довольно мощная рабочая станция. Некоторые версии были рассчитаны только на определенные платформы. Например, SunOS была предназначена только для рабочих станций Sun, AIX — для рабочих станций IBM, а AUX — для компьютеров Macintosh.

Прообразом Linux стала операционная система **Minix,** разработанная Эндрю Таннебаумом. ОС Minix являлась небольшой UNIX-системой, которая была предназначена даже не для реального использования, а для демонстрации возможностей системы Unix. Вдохновленный идеей создать собственную Minix, Линус Торвальдс начал работу над операционной системой Linux. Впервые ОС Linux обсуждалась в конференции USENET **comp.os.minix.**

5-го октября 1991 года Линус Торвальдс объявил о выходе первой «официальной» версии **Linux 0.02.** Тогда в этой операционной системе работали только интерпретатор **bash** (Bourne Again Shell) и gcc (GNU C compiler). Основное внимание уделялось созданию ядра. Никакие вопросы поддержки пользователей, тиражирования и документирования даже не обсуждались.

Постепенно к разработке ядра и другого программного обеспечения присоединились тысячи других разработчиков со всего мира. На сегодняшний день Linux считается единственным примером столь масштабного сотрудничества программистов. Linux сегодня — это полноценная операционная система семейства UNIX, поддерживающая широкий спектр аппаратных средств, протокол TCP/IP, графический интерфейс пользователя, что позволяет использовать ее не только как сервер, но и как высокопродуктивную рабочую станцию.

С самого начала ОС Linux разрабатывалась для персональных компьютеров на платформе Intel. Со временем некоторые компании начали разрабатывать версии Linux для своей платформы, например, Sun Microsystems. Многие компании, в том числе и отечественные, разработали свои версии Linux.

Прочитав много книг, посвященных Linux, я заметил, что практически во всех, особенно в книгах зарубежных авторов, есть раздел «Где достать Linux?». В этой книге такого раздела не будет благодаря самой ОС Linux.

Повсеместное распространение этой операционной системы привело к тому, что дистрибутив Linux можно купить практически в любом магазине, торгующем компакт-дисками, а на рынке даже появились полиэтиленовые пакеты с изображением пингвина и надписью «Linux».

1.3. Почему именно Linux?

Каждая операционная система имеет свое «призвание». Операционную систему Windows NT Server предпочтительнее использовать как сервер рабочих групп сетей Microsoft. Система Novell Netware лучше «смотрится» в роли файлового сервера и сервера печати. ОС UNIX первоначально разрабатывалась как Интернет-сервер. Средства для работы с Сетью встроены непосредственно в ядро этой операционной системы, а все необходимое программное обеспечение для организации сервера входит в состав дистрибутива. UNIX-система работает со всеми сетевыми протоколами (особенно с TCP/IP) лучше, чем любая другая операционная система для платформы Intel. Все перечисленные выше качества касаются также и ОС Linux.

Устанавливая Linux, вы получаете также множество других преимуществ. Во-первых, вам становятся доступны исходные тексты ядра и вы можете модифицировать систему так, как вам нужно. Такое можно встретить далеко не в каждой операционной системе, особенно в ОС семейства Microsoft. Вы видели где-нибудь исходные тексты хотя бы Блокнота Windows? Мне, например, очень не хватает функции замены текста в этом редакторе. Для решения этой проблемы я написал собственный редактор, в котором и реализовал эту функцию. А если мне нужно сделать небольшое изменение в ядре? Не буду же я полностью переписывать Windows? Или ожидать новую версию «монстра», пожирающего системные ресурсы, в котором есть всего одна нужная мне функция?

Во-вторых, ОС Linux абсолютно бесплатна. Конечно, существуют коммерческие версии Linux, но в этом случае вы платите за некоторые дополнительные функции и техническую поддержку. Купив однажды компакт-диск с Linux, вы можете установить эту операционную систему на неограниченном числе компьютеров. Вам не нужно ничего доплачивать, вам не нужно платить за каждый дополнительный процессор — ОС Linux поддерживает SMP и при этом тоже бесплатно. Кроме того, поскольку Linux является UNIX-подобной системой, в состав ее дистрибутива входит все программное обеспечение, необходимое для организации сервера.

В последнее время появилась тенденция выпускать многодисковые дистрибутивы или дистрибутивы, содержащие программное обеспечение отдельно для рабочей станции и отдельно для сервера. В этом случае вам нужно купить только первый и, возможно, второй компакт-диск. Если дистрибутивы разделены на категории «сервер» и «рабочая станция», как это сделали разработчики ASP Linux, покупайте, естественно, серверную версию.

В любом случае стоимость всего программного обеспечения составит несколько долларов. Я не буду сравнивать стоимость построения Linux-

сервера со стоимостью аналогичного сервера на платформе Microsoft. Вы сами это можете сделать на сайте компании Microsoft. К тому же, если вам необходим сервер Windows NT(2000) Server, вы можете спокойно заменить его SMB-сервером на базе Linux. Кстати, решению именно этого вопроса посвящена глава 9 данной книги.

В-третьих, ОС Linux легка в освоении и сопровождении. Для облегчения перехода с ОС Windows NT(2000) Server, где вы для настройки сервисов в основном используете графический интерфейс, создано множество графических конфигураторов. Эти конфигураторы значительно упрощают процесс настройки системы. Для большего понимания я старался в книге излагать материал, не прибегая к помощи конфигураторов. Зная расположение и формат системных файлов, вы сможете настроить практически любой дистрибутив, в котором нет графических конфигураторов или они недоступны. Что касается самих конфигураторов, то работа с ними интуитивно понятна и, если вы будете представлять себе суть дела, не должна вызвать затруднений.

В-четвертых, операционная система Linux не так требовательна к системным ресурсам как другие операционные системы. Например, для организации Интернет-сервера вам вполне хватит старенького компьютера с процессором Intel 80486DX и 32 мегабайтами ОЗУ. Конечно, системные требования зависят от версии ядра и выбранного вами дистрибутива. Организовать сервер на вышеупомянутой машине можно, используя дистрибутив Red Hat Linux версии 5.2. Возможно, он не будет удовлетворять всем требованиям безопасности, нд при правильной настройке вам подойдет и этот дистрибутив. И в самом деле, вы же не собираетесь строить систему электронных платежей, основанную на процессоре Intel 80486.

ОС Linux, как и большинство программного обеспечения для этой операционной системы, распространяется по лицензии GPL. В двух словах лицензия GPL означает, что вы можете свободно использовать и распространять программное обеспечение, лицензируемое GPL, а также использовать его для создания другого свободно распространяемого программного обеспечения.

1.4. Область применения Linux-серверов

Операционная система Linux получает все большее распространение. В настоящее время Linux все чаще можно увидеть установленной на компьютерах домашних пользователей. Этому способствует дружественный интерфейс, надежность и быстродействие ОС Linux. Определенную роль, конечно, сыграла и лицензионная политика корпорации Microsoft — домашнему пользователю дешевле купить ОС Linux, чем покупать новый компьютер за $400 и Windows 98 за $60.

В этой главе мы не будем обсуждать преимущества Linux в качестве настольной системы, а поговорим о Linux-серверах. Microsoft Windows NT Server (и Windows 2000 Server), на мой взгляд, больше подходит как сервер для небольшой рабочей группы. По своим параметрам Windows NT(2000)

Server довольно надежна и быстра, но все же ей не хватает масштабируемости, несмотря на заявления Microsoft о своих серверах.

Чтобы понять, почему Linux целесообразно использовать именно в качестве Интернет-сервера, немного вспомним историю. Давайте сначала обратимся к тем далеким 80-м годам прошлого столетия, когда в кабинетах стояли «калькуляторы» под управлением DOS: неприятный интерфейс, однозадачность, отсутствие поддержки мультимедиа, а под термином «сеть» понималось соединение двух компьютеров через последовательный или параллельный порт с помощью Norton Commander. В начале 90-х годов (а именно в 1993 году) ситуация несколько изменилась: Microsoft выпустила рабочий вариант ОС Windows — Windows 3.1. Что же было новым в Windows 3.1? Во-первых, это многозадачность, во-вторых, поддержка виртуальной памяти, и, конечно же, относительно удобный графический интерфейс. В операционной оболочке (системой ее назвать трудно) Windows 3.1 не были реализованы сетевые функции. Поддержка сети появилась в следующей редакции Windows — **Windows for Workgroups.**

В 1995 году появилась операционная *система* Windows 95, представленная Microsoft чуть ли не самой совершенной системой. Однако эта операционная система также не была сетевой, *а лишь с поддержкой сетевых функций.* Принципиальным отличием от Windows 3.1 была 32-разрядность этой системы. В следующем году Microsoft выпускает настоящую сетевую операционную систему — Windows NT 4 Server. В этой системе был реализован (и нормально функционировал) протокол TCP/IP, который является стандартом сети Интернет, но протоколом по умолчанию он не являлся, а устанавливался опционально, то есть по требованию администратора.

В 2000 году мы стали свидетелями появления новой сетевой серверной операционной системы от Microsoft — Windows 2000 Server. Помимо прочих достоинств Windows 2000 по сравнению с Windows NT 4 Server, нужно отметить службу управления каталогами **Active Directory,** поддержку по умолчанию протокола TCP/IP, а также средства для квотирования (ограничения дискового пространства).

Все кажется просто прекрасным, однако, если разобраться более детально, то в 2000 году Microsoft достигла того, что уже существовало в Unix еще в 80-х годах. Попробую сейчас все разъяснить. С самого своего начала (с 1979 года), операционная система Unix была:

1. 32-разрядной.
2. Многозадачной.
3. Многопользовательской, а значит *сетевой.*

Достижения инженерной мысли, о которых мы узнали только в 1993 году — многозадачность и поддержка виртуальной памяти, были реализованы в Unix еще в далеком 1979 году. К тому же, протокол TCP/IP включен в состав ядра системы по умолчанию, а это говорит о многом. Хотя бы о том, что благодаря поддержке ядром протокола TCP/IP достигается высокое быстродействие программ, использующих этот протокол. Существует даже выражение: «Unix создан для сети, как птица для полета».

Квотированием, которое появилось в Windows совсем недавно, настоящих юниксоидов тоже не удивишь. А службу Active Directory можно заменить на **Network Information Service.** Может быть, это и не полноценная замена, но, учитывая, что этой службе уже пошел второй десяток лет...

Не нужно думать, что Unix всегда был неподъемным монстром с DOS-образным интерфейсом. Графическая система X Window создана достаточно давно и существует множество Linux-приложений, использующих графический интерфейс. А благодаря таким графическим средам как **KDE** и **Gnome,** Linux стал еще более дружелюбным. Но в нашем случае графический интерфейс не столь существенен -- мы же будем с вами настраивать сервер. Например, компания Novell поначалу вообще отказалась от использования графического интерфейса в своей операционной системе, чтобы не задействовать дополнительные системные ресурсы. А поскольку Linux является прямым потомком Unix, то ей присущи все вышеописанные качества. К тому же, операционная система Linux совершенно бесплатна — об этом я уже упоминал немного раньше. Если вы считаете себя серьезным администратором, то выбор между надежностью, производительностью Linux и дружелюбным интерфейсом Windows NT (2000) Server, я думаю, очевиден.

Еще один важный аспект — документация системы. Все без исключения Unix-подобные системы очень хорошо документированы и поэтому вся необходимая информация для настройки сервера по сути уже есть в вашем компьютере. Моя же задача при этом сводится к тому, чтобы научить вас использовать эту документацию, а также на практике рассмотреть ее применение.

Итак, что же получается, что мы, используя программное обеспечение от Microsoft, отстали на двадцать лет в развитии, использовали, мягко говоря, не совсем надежное программное обеспечение и еще платили за это деньги? Ребятам из маркетингового отдела Microsoft нужно памятник поставить за их профессиональные качества.

Где же применяются Linux-серверы? Прежде всего, это Интернет-серверы. Вы можете спросить, почему именно Linux (Unix)? Почему не какая-нибудь другая операционная система, например, Windows NT (2000)? Давайте подумаем вместе. В начале 60-х годов по приказу Министерства обороны США была создана сеть Arpanet, которая и послужила в дальнейшем прототипом для создания Интернет. Как можно использовать NT-сервер в качестве Интернет-сервера, если он был выпущен в 1996 году? А Интернет-то существовал с 70-х годов. И существовал именно благодаря Unix-системам. Так почему же не использовать для предоставления Интернет-услуг родную операционную систему? В самом деле, не будете же вы покупать для своего BMW запчасти от Hond'ы? В случае с Интернет это равноценное сравнение: Linux (Unix) для Интернет — абсолютно родная система. Использование других систем допустимо и кому-то может показаться более удобным, но только на том уровне, как будто бы вы действительно в автомобиль одной хорошей марки вставили бы деталь от автомобиля другой хорошей марки. Обе марки автомобилей хороши, но детали одной из них не предназначены для другой. Для справки: относительно недавно был открыт сайт президента

России. Так вот этот сайт, к которому предъявляются повышенные требования надежности, безопасности и производительности, базируется именно на основе Red Hat Linux.

Многие правительственные и финансовые организации всего мира, например, Министерство иностранных дел Германии, используют Linux (SuSE Linux), а немецкий Dresdner Bank совместно с американской компанией CollabNet объявил о новой банковской информационной системе, построенной на основе Linux. И тут, как вы видите, дело не в деньгах — платить или не платить за Linux, а в заботе организаций о своей информационной безопасности и надежности своих серверов. Как объяснить клиенту, что его счет «будет закрыт», поскольку «программа выполнила недопустимую операцию»? Тут даже созданный журнал ошибок не поможет.

Второй отраслью применения Linux-серверов является создание кластеров для произведения параллельных вычислений. По определению кластер — это несколько компьютеров, объединенных вместе для совместного решения одной задачи. Объединение компьютеров, как правило, производится с помощью высокоскоростной сети. На сегодняшний день создано специальное программное обеспечение, позволяющее собрать кластер даже в домашних условиях, например, PVM (Parallel Virtual Machine). Более подробно останавливаться на этом вопросе мы не будем, так как параллельные вычисления и компьютерное моделирование — это уже тема для другой книги.

Помимо всего вышеуказанного, существует еще множество направлений, где используются Linux-серверы: Web-серверы, FTP-серверы, почтовики, шлюзы, X-серверы, можно даже эмулировать домен NT с помощью пакета Samba. Все эти вопросы будут рассмотрены в данной книге.

1.5. Как устроена данная книга

«Не сразу Москва строилась», поэтому и настройку нашего сервера будем производить с малого, а потом постепенно будем его наращивать. Сразу нужно оговориться, что, возможно, при настройке вашего сервера вам не потребуется читать все главы. Например, если вы хотите настроить почтовик, то вам незачем разбираться с настройкой сервера для удаленного доступа. Хотя прочитать всю книгу все же стоит — для общего развития.

Во второй главе достаточно подробно рассмотрена установка операционной системы Linux на примере дистрибутивов Linux Mandrake и Linux Red Hat. В этой же главе рассматривается постинсталляционная настройка пока что настольной версии Linux.

В третьей и четвертой главах обсуждаются общие принципы работы с операционной системой Linux. В третьей главе рассматриваются учетные записи, а в четвертой — работа с файловой системой Linux. Очень советую вам разобраться с правами доступа к файлам и каталогам. Большое значение для организации сервера имеет настройка дисков SCSI и массивов RAID. Особое внимание обратите на создание резервных копий, если вы, конечно, настраиваете «серьезный» Linux-сервер.

В пятой главе рассматриваются основы управления процессами в Linux. Эта глава имеет больше теоретическое, чем практическое значение, но какая же практика без теории?

Шестая глава посвящена русификации дистрибутива. Так как современные дистрибутивы практически не имеют проблем с русификацией, вы можете без особых угрызений совести прочитать данную главу «по диагонали».

В седьмой главе рассмотрена базовая настройка сети — установка сетевой платы, настройка протокола TCP/IP, подключение к Интернет по модемной и выделенной линиям. Особое внимание уделено настройке ADSL-соединения, которое сейчас получает все большее распространение.

В восьмой главе обсуждается непосредственная настройка сервера. Рассматриваются суперсерверы inetd и xinetd. Первый использовался в старых дистрибутивах, но вы все же можете еще встретиться с ним, а xinetd является стандартом для современных Linux-серверов. Получение и предоставление «безопасного» удаленного доступа с помощью ssh, настройка протокола динамической настройки хоста — DHCP, сетевая файловая система NFS, маршрутизация, подсчет трафика, поисковый сервер ht:/Dig — обо всем этом вы можете прочитать в восьмой главе. И это еще не все: последними пунктами в восьмой главе приведены описания настроек прокси-сервера Socks5, а также системы обнаружения и защиты от вторжения LIDS. Особенный интерес представляет программа подсчета трафика MRTG, также описанная в данной главе.

Девятая глава посвящена настройке популярного пакета Samba, который используется для получения доступа к ресурсам Windows-сети. С помощью этого пакета вы также сможете настроить свой Linux-сервер для предоставления ресурсов пользователям сети Windows, которые даже и не заметят, что вместо Windows-сервера у вас установлен Linux-сервер.

В десятой главе рассматривается настройка службы доменных имен — DNS. Для настройки практически любого Интернет-сервера вам придется конфигурировать эту службу. Немного внимания я уделил средствам безопасности, которые предоставляет сервер DNS, например, передаче зоны только определенным узлам. Подробно описано конфигурирование основного и дополнительного серверов DNS, обновление файлов конфигурации, создание кэширующего DNS-сервера.

Популярная служба передачи файлов — FTP — рассматривается в **одиннадцатой главе.** В этой главе обсуждаются серверы wu-ftpd и ProFTPD, а также организация виртуальных FTP-узлов.

Web-сервер Apache, практически ставший стандартом в Интернет, обсуждается в **двенадцатой главе.** Рассматриваются: базовая настройка, каталоги пользователей, настройка виртуальных серверов, установка и настройка SSL, а в конце главы приведен полный листинг конфигурационного файла.

Тринадцатая глава посвящена настройке сервисов POP и SMTP, а также SMTP-аутентификации. Для непосвященных: SMTP — это протокол для отправления сообщений, а POP — для их приема. Кроме этого, приведено описание создания своей собственной службы рассылки. В конце главы

объясняется, как настроить популярные клиенты электронной почты The Bat!, Netscape Messenger, Outlook Express.

Безопасность при работе в сети Интернет — вот основная тема **четырнадцатой главы** «Бастионы», в которой рассматривается фильтрация пакетов. Здесь описывается, как закрыть доступ к вашей внутренней сети для нежеланных гостей, а также как предотвратить доступ пользователей к нежелательным ресурсам сети.

В пятнадцатой главе подробно рассматривается настройка прокси-сервера SQUID, разделение канала, программы учета трафика. Вы хотите достичь существенного прироста в работе с Web или вам надоело тратить свое драгоценное время и деньги на загрузку рекламных баннеров? Тогда эта глава для вас.

Небольшая **шестнадцатая глава** посвящена настройке тоже небольшого, но с большими возможностями сервера баз данных MySQL. Сначала описывается настройка сервера, а потом клиента MySQL. Последним пунктом главы приведено описание настроек связки Apache+PHP+MySQL.

Очень большое значение для всей книги имеет **семнадцатая глава** — «Практические примеры». В ней рассматривается настройка шлюза для доступа в Интернет и сервера удаленного доступа, а также приводится пример настройки «обратного звонка». Эта глава является как бы кульминацией всей книги. На протяжении всех предыдущих глав мы накапливаем знания, а потом настраиваем сразу два реальных сервера (или два в одном), плюс настройка полезной технологии. Примеры конфигурационных файлов, приведенные в гл. 17, являются полностью рабочими. Если у вас что-то не заработало, особенно при настройке сервера удаленного доступа, то у вас просто неправильно настроены модемы. Для их правильной настройки прочтите руководство по модему. Конфигурации для некоторых модемов приведены в этой же главе.

Оптимизация работы ядра операционной системы Linux и его компилирование рассматривается в **восемнадцатой главе.** Иногда полезно исключить из ядра лишний код для повышения производительности всей системы. Да и ядро при этом становится более компактным.

В девятнадцатой главе приведены некоторые полезные команды при работе с операционной системой Linux. Эти команды полезны скорее для пользователя, чем для администратора, но администратор ведь должен знать, чем может заниматься пользователь.

Двадцатая глава посвящена установке, настройке и использованию графической системы X Window, а также оконных сред KDE и GNOME.

На сегодняшний момент времени очень большое распространение и популярность приобрели компьютерные клубы. Однако ни для кого не является большим секретом то, что абсолютное большинство из них использует нелицензионное программное обеспечение, в том числе и продукты компании Microsoft. Тому, как перевести компьютерный клуб на бесплатную операционную систему Linux, посвящена **глава** под номером **двадцать один.** В этой главе рассмотрены также вопросы администрирования компьютерно-

го клуба (управление пользователями, вопросы безопасности, запись журналов, отслеживание времени работы и т.п.), а также приведен листинг несложного управляющего модуля (Louncher'a). Экспериментируя с ним, вы сможете добиться самых различных результатов. Большое внимание в двадцать первой главе уделено описанию настроек Windows-игр под Linux и использованию Windows-эмулятора. Благодаря информации, которю вы почерпнете в данной главе, вы сможете спокойно играть в Quake 2 и 3, Counter Strike, Unreal Tournament, Diablo 2 и многие другие.

Двадцать вторая глава целиком посвящена антивирусной защите. Подробно расмотрены настройка и использование под Linux лучших антивирусов: DrWeb и AVP. Отдельно написано о проверке входящей и исходящей почты на предмет вирусов.

В двадцать третьей главе приводится описание вопросов, которые не столь существенны, чтобы под каждый из них выделять отдельную главу, но которые являются очень полезными: сканер портов SATAN. защита от спама, ограничение системных ресурсов.

Своего рода напутствие и личные рекомендации я разместил в заключительной, **двадцать четвертой главе.**

В приложениях приведено назначение и расположение системных файлов Linux **(приложение** А) и общие параметры программ, предназначенных для работы с системой X Window **(приложение** Б). Очень полезным может оказаться рассмотрение моего компактного ядра Linux, приведенное в **приложении В.** Помимо самого листинга даны рекомендации как с ним работать и как его использовать. В **приложении Г** приведен список наиболее интересных и полезных ссылок (URL-адресов), по которым вы можете найти самую различную информацию, касающуюся ОС Linux. Описание прилагаемого компакт-диска вы найдете в последнем приложении.

1.6. Какие сервера бывают и для чего они нужны

Для полноты картины рассмотрим несколько различных типов серверов:

1. Сервер локальной сети.
2. Шлюз.
3. Сервер удаленного доступа.

Сервер любого типа вы сможете настроить с помощью данной книги.

1.6.1. Сервер локальной сети

Сервер локальной сети (рис. 1.1) — это сервер, оказывающий услуги пользователям сети Интранет (Intranet). Не путайте «Интранет» с «Интернет». Сеть Intranet — это внутренняя корпоративная сеть, как правило, без выхода в Интернет.

Представьте себе небольшую офисную сеть. Если у вас развито воображение, вы даже можете себе представить большую локальную сеть, размещенную в многоэтажном здании. В такой сети сервер может выполнять самые разнообразные функции, например, быть сервером печати

или файловым сервером. Обычно во внутренних сетях серверы выполняют только эти функции. В зависимости от вида деятельности организации, в которой установлен сервер, довольно часто он используется в качестве сервера баз данных.

Для сервера баз данных и файлового сервера вообще желательно выделить по одному компьютеру, потому что если нагрузка на сервер баз данных довольно большая, а это в большинстве случаев именно так, то это будет снижать производительность файлового сервера, и наоборот. Если производительность сервера будет низкой, то, как всегда, страдает пользователь. В свою очередь, «страдания» пользователей отразятся ни на ком другом, кроме как на вас — администраторе.

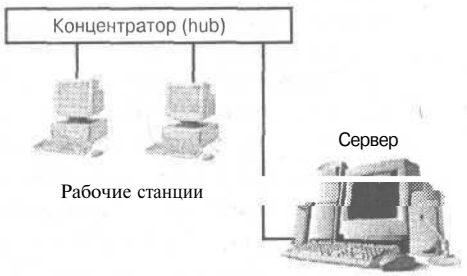


Рис. 1.1. *Сервер локальной сети*

На рис. 1.1 изображена небольшая локальная сеть без выхода в Интернет. Все рабочие станции и сам сервер подключены к центральному устройству сети - - концентратору. Вместо концентратора можно использовать (и предпочтительнее использовать) коммутатор (switch). В отличие от коммутатора, концентратор «не знает», к какому порту подключен тот или иной компьютер и когда один из компьютеров передает пакет данных, то концентратор повторяет его на все свои порты. Каждый компьютер сети получает этот пакет и проверяет наличие своего IP-адреса в его заголовке. Если IP-адрес назначения не совпадает с IP-адресом компьютера, который принял этот пакет, то пакет просто игнорируется. Коммутатор же передает пакет только на тот порт, к которому подключен адресат. Благодаря этому, помимо повышения безопасности, снижается нагрузка на сеть. Вот так пакеты данных доставляютсяя от источника к месту назначения.

Кроме концентратора или коммутатора, в большинстве случаев не нужно никакого другого сетевого оборудования, за исключением разве что сетевых плат. В случае большой протяженности сети используются повторители, которые, пропуская через себя сигнал, усиливают его. По своей сути повторитель представляет собой простейший концентратор. Концентратор же, за вычетом некоторых сервисных функций, представляет собой многопортовый повторитель.

О выборе сетевого адаптера и о другом «железе» для сервера написано в п. 7.6 «Несколько слов перед настройкой сервера». Здесь же нужно сказать пару слов о выборе коммутатора. Раньше основным камнем преткновения для использования коммутаторов была их высокая стоимость. Сейчас можно купить довольно производительные коммутаторы за относительно небольшие деньги. Для домашнего или небольшого офиса можно порекомендовать коммутатор LUCENT CAJUN P115G. Он имеет 24 порта 10/100 Мбит, а

также 1 оптоволоконный порт 100FX. По последним данным его стоимость составляет около 230 долларов США.

Более дорогой и более производительный вариант на 48 портов 10/100 Mbit LUCENT-CAJUN P334T. Этот коммутатор обладает внутренней шиной на 8 Гбит, а также модульной конструкцией, что позволяет устанавливать дополнительные модули. Стоимость модели P334T -- около $900. При установке дополнительных оптоволоконных портов 100FX стоимость увеличится примерно на $150...250 в зависимости от количества портов.

1.6.2. Шлюз — сервер для доступа в Интернет

В том случае, если в вашей внутренней сети необходим доступ к Интернет, имеет смысл установить сервер для доступа к всемирной Сети (см. рис. 1.2). Это и есть шлюз. Шлюзом может быть и отдельное устройство, но в локальных сетях с выходом в Интернет обычно устанавливается целый сервер для доступа к Интернет. Это намного удобнее, так как кроме шлюза можно также настроить Web, FTP, SMTP/POP-сервер. В этом случае вы сможете размещать информацию о вашей компании в Интернет и настроить собственный почтовый сервер. Для этих же целей существуют аппаратные преднастроенные решения различных компаний, например, компании Intel. Но возможности администрирования таких устройств довольно ограниченные. Ощутить свободу действий как администратор вы сможете только в случае, когда у вас будет полноценный сервер.

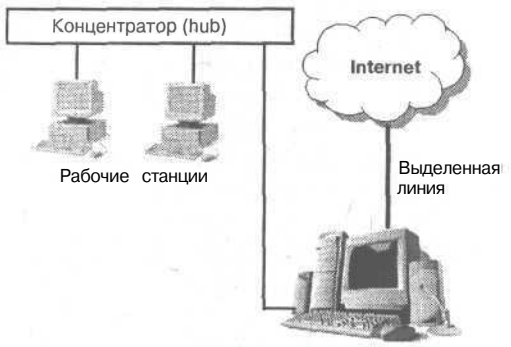


Рис. 1.2. *Сервер для доступа к Интернет (шлюз)*

Что же изменилось по сравнению с рис. 1.1? Вы правы, появилась выделенная линия. Обычно подключение сервера (читайте: локальной сети) к Интернет происходит по выделенной линии. В простейшем случае для организации выделенной линии необходим модем, позволяющий работать на выделенных линиях. Я бы порекомендовал использовать ZyXEL U336S. Этот модем поддерживает двух- и четырехпроводные линии, а также синхронную и асинхронную передачу данных. Максимальная скорость передачи 300...480 Кбит/с. Это уже будет считаться рекламой модемов ZyXEL, но я все-таки напишу, что данные модемы работают практически на любых (даже на самых зашумленных) линиях. При организации модемного пула, о котором поговорим немного позже,также желательно установить модемы ZyXEL.

В большинстве случаев скорость передачи 300...480 Кбит/с вас не будет устраивать. В этом случае могут помочь DSL-модемы, обеспечивающие передачу данных со скоростью до 1 Мбит/с. Если позволяют условия расположения вашего офиса, можно купить оборудование Radio Ethernet.

1.6.3. Сервер удаленного доступа

Теперь представьте, что ваша компания немного разрослась и кроме нескольких соседних зданий, соединенных между собой оптоволоконным кабелем, появилось еще несколько филиалов в разных концах города. Нужно обеспечить пользователям удаленных филиалов возможность работать в сети компании. При этом вам не нужно, чтобы эти пользователи находились в сети постоянно. В этой ситуации на сцене появляется сервер удаленного доступа (см. рис. 1.3).

Работа с сервером удаленного доступа выглядит следующим образом. Удаленный пользователь звонит по определенному номеру модемного пула компании. Сервер удаленного доступа аутентифицирует пользователя и предоставляет ему доступ, если аутентификация прошла успешно. При этом удаленному пользователю кажется, что он работает непосредственно в сети компании, если не считать медленного канала передачи данных. Другим (не удаленным) пользователям сети будет казаться, что удаленный пользователь находится где-то рядом — в этом же здании.

Однако, если нужно обеспечить постоянную работу удаленных пользователей в сети компании, то для этого существуют более эффективные решения, например, технология Radio Ethernet.

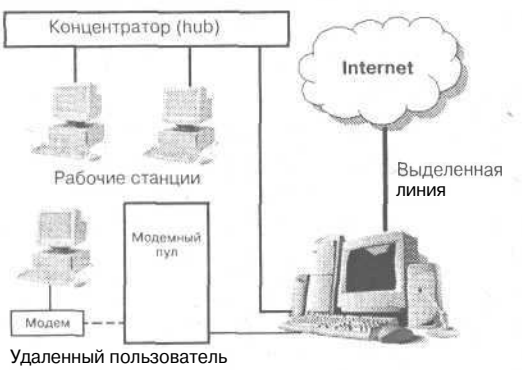


Рис. 1.3. Сервер удаленного доступа

Обратите внимание, что в сети появилось новое устройство — модемный пул. Это просто совокупность модемов, подключенных к серверу. Обычно все модемы устанавливаются в специальные стойки, но это не обязательное требование — просто ради удобства. У вас может возникнуть вопрос: как все эти модемы подключаются к серверу, если максимально можно подключить четыре последовательных устройства (а без установки дополнительных контроллеров — только два)? Для этого используется мультипортовая плата, которая обеспечивает подсоединение большого количества модемов (и других устройств), подключаемых к последовательному порту компьютера. При этом все модемы подключаются к мультипортовой плате, а она, в свою очередь, к компьютеру.

Пунктирной линией на рис. 1.3 обозначено непостоянное (dial-up) подключение пользователя.

Как я уже писал, для модемного пула я рекомендую устанавливать модемы ZyXEL. Модемы Robotics тоже хорошо работают, но ZyXEL — это мой субъективный выбор. В случае с выделенной линией вероятность возникновения каких-либо помех на линии значительно меньше, чем при работе с обыкновенными телефонными линиями. Чтобы обеспечить надеж-

ное соединение пользователя, нужно установить профессиональные модемы и правильно настроить их. О том, как настраивать модемы, вы прочитаете в документации по модему. Лучший модем в этом случае - - это модем, который лучше остальных работает на данной линии. Сейчас мы не рассматриваем случай, когда вы являетесь Интернет-провайдером и у вас нет возможности проконтролировать, какой модем установил пользователь. Перед покупкой модема протестируйте его работу на линии — обычно фирма-продавец разрешает сделать это. Когда вы найдете оптимальный вариант по цене и качеству — покупайте.

Тестировать подключение нужно непосредственно на той АТС, к которой подключен пользователь. Не повторяйте распространенной ошибки, когда администратор, настроив сервер удаленного доступа, пытается протестировать соединение, позвонив с другого номера той же АТС. Учитывая качество наших телефонных линий, результаты тестирования на разных АТС будут отличаться. Лучше всего тестировать подключение, находясь на рабочем месте удаленного пользователя.

Обратите внимание на «карьеру» нашего сервера: мы постепенно наращиваем его, добавляя новые функции. Сначала он был рядовым сервером локальной сети, затем он начал предоставлять доступ пользователей к Интернет, а потом мы построили сервер удаленного доступа.

1.7. Что такое сервер? (или Курс Молодого Администратора)

Эта глава предназначена для начинающих системных администраторов, которые вообще с трудом представляют, что такое сервер и с чем его едят. Я принципиально не буду называть таких читателей «чайниками», поскольку сам когда-то таким был. В этой главе я попытаюсь вкратце объяснить некоторые термины, которые связаны с сетью (я имею в виду компьютерной!), а также с протоколом TCP/IP. Эту главу можно рассматривать как своеобразный курс молодого бойца (администратора). Сразу нужно отметить, что углубляться в технические подробности мы не станем — для этого существует множество другой литературы, например, «Компьютерные сети. Принципы, технологии, протоколы» В. Г. Олифера.

1.7.1. Архитектура сети: одноранговая и клиент/сервер

Начнем с самого главного — архитектуры сети. Существуют две основные архитектуры сети: одноранговая (peer-to-peer) и клиент/сервер (client/server), причем вторая практически вытеснила первую. В одноранговой сети все компьютеры равны — имеют один ранг. Любой компьютер может выступать как в роли сервера, то есть предоставлять свои ресурсы (файлы, принтеры) другому компьютеру, так и в роли клиента, другими словами – использовать предоставленные ему ресурсы. Одноранговые сети преимущественно распространены в домашних сетях или небольших офисах. В самом простом случае для организации такой сети нужно всего лишь пара ком-

пьютеров, снабженных сетевыми платами и коаксиальный кабель (нужна еще пара терминаторов (заглушек), но я обещал сильно не углубляться).

Когда сеть создана физически (компьютеры связаны с помощью коаксиального кабеля), нужно настроить сеть программно. Для этого необходимо, чтобы на компьютерах были установлены сетевые операционные системы (Linux, FreeBSD, Windows NT, Windows 98) или сетевые системы с поддержкой сетевых функций (Windows 95, Windows for Workgroups).

Компьютеры в одноранговой сети объединяются в **рабочие группы.** Каждая рабочая группа имеет свой идентификатор — имя рабочей группы. Если вы сейчас работаете в Windows 9x, узнать имя рабочей группы вы можете, запустив апплет **Сеть** с **Панели управления** (см. рис. 1.4).

Для примера допустим, что в вашей одноранговой сети три компьютера A, B, C. Первые два входят в рабочую группу WG1, а компьютер C — в рабочую группу WG2 (см. рис. 1.5).

Даже несмотря на то, что компьютеры входят в один сегмент сети (физически подключены к одному кабелю), компьютеры A и B не будут «видеть» компьютер C, а компьютер C не будет видеть компьютеры A и B. Если выполнить команду поиска компьютера в Windows 9x **(Пуск → Поиск → Найти компьютер),** компьютер «увидит» компьютеры A и B, но будет сообщено, что они находятся в другой рабочей группе — WG1.

Единственное ограничение доступа, которое возможно в одноранговой сети, это использование пароля для доступа к какому-нибудь ресурсу. Для того, чтобы получить доступ к этому ресурсу, например, принтеру, нужно знать пароль. Это называется управлением доступом на уровне ресурсов. В сети клиент/сервер используется другой способ управления доступом — на уровне пользователей. В этом случае можно разрешить доступ к ресурсу только определенным пользователям. Например, ваш компьютер A через сеть могут использовать два пользователя: Иванов и Петров. К этому компьютеру подключен принтер, который можно использовать по сети. Но вы не хотите, чтобы кто угодно печатал на вашем принтере, и установили пароль для доступа к это-

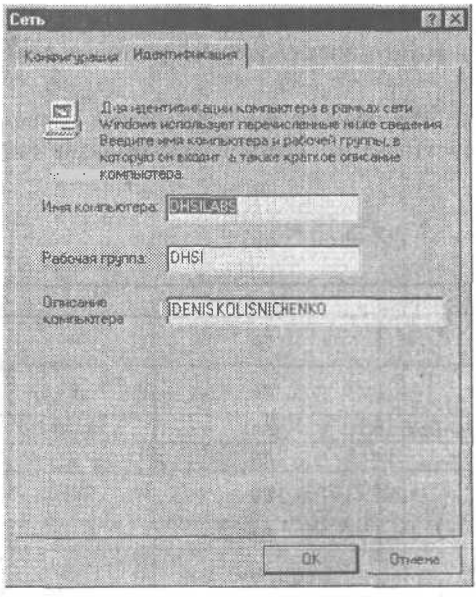


Рис. 1.4. *Идентификатор рабочей группы в ОС Windows 9x*

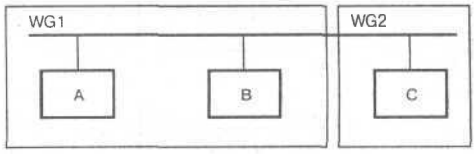


Рис. 1.5. *Схема одноранговой сети*

му ресурсу. Если у вас одноранговая сеть, то любой, кто узнает этот пароль, сможет использовать ваш принтер. В случае с сетью клиент/сервер вы можете разрешить использовать ваш принтер только Иванову или только Петрову (можно и обоим).

Для получения доступа к ресурсу в сети клиент/сервер пользователь должен ввести свой уникальный идентификатор — имя пользователя (login — логин) и пароль (password). Логин пользователя является общедоступной информацией и это правильно: возможно, если кто-нибудь захочет отправить пользователю сообщение по электронной почте, то для этого ему достаточно знать его логин (естественно, и имя сервера электронной почты, который «знает» этого пользователя).

Использование логина и пароля для доступа к ресурсам называется *аутентификацией пользователя* (user authentication). Существуют и другие виды аутентификации, например, аутентификация источника данных или одноранговой объекта, но сейчас мы рассматривать их не будем. В любом случае, *аутентификация* — это проверка подлинности.

После рассмотрения архитектуры одноранговой сети можно прийти к выводу, что единственное преимущество этой архитектуры — это ее простота и дешевизна. Сети клиент/сервер обеспечивают более высокий уровень производительности и безопасности.

В отличие от одноранговой сети, в сети клиент/сервер существует один или несколько главных компьютеров — серверов. Все остальные компьютеры сети называются клиентами или рабочими станциями (workstations). Как я уже писал выше, *сервер* — это специальный компьютер, который предоставляет определенные услуги другим компьютерам. Существуют различные виды серверов (в зависимости от предоставляемых ими услуг): серверы баз данных, файловые серверы, серверы печати (принт-серверы), почтовые серверы, Web-серверы и т.д.

В табл. 1.1 перечислены лишь немногие функции, выполняемые сервером, и рекомендуемое программное обеспечение, которое необходимо для реализации этих функций.

Функции и программное обеспечение сервера *Таблица 1.1*

Функция	Программное обеспечение	Дистрибутив	Глава
Авторизация удаленных пользователей (dialup)	Пакет ppp	Да	17
Автоматическое конфигурирование узлов сети	dhcp	Да	8
Доступ (совместный) к файлам	NFS, FTPd (ProFTPD, wu-ftpd)	Да	8, 13
Доступ к сети Microsoft	Пакет samba	Да	9
Кэширование передаваемой информации	Пакет Squid	Да	15
Маршрутизация	route(d)	Да	8, 14
Обмен сообщениями электронной почты	Пакеты sendmail (или postfix/qmail) и imap	Да (Да/Нет), Да	13
Подсчет передаваемого по сети трафика	ядро Linux, IPChains	Да	8, 14

Продолжение табл. 1.1

Функция	Программное обеспечение	Дистрибутив	Глава
Передача секретной информации	*modSSL*	Да (не во всех дистрибутивах)	12
Разрешение (резолвинг) имени компьютера в IP-адрес	Пакет bind	Да	10
Сетевая печать	Lpd, Samba, CUPS	Да	6, 9
Функции Web-сервера	Пакет apache	Да	12
Фильтрация пакетов	IPChains (IPTables в новых версиях Linux)	Да	14
Управление базой данных	MySQL/ PostgreSQL/ InterBase	Да/Да/Не во всех	16
IP-Маскарадинг	IPChains	Да	14

В графе Дистрибутив отмечено, входит ли указанное программное обеспечение в состав распространенных дистрибутивов, а в графе Глава указывается глава книги, в которой описана настройка интересующей вас функции.

Для экономии средств, как правило, один сервер сочетает в себе функции нескольких серверов, например, почтовик может быть также и Web-сервером. Услуги, которые может предоставлять сервер, ограничиваются только его физическими возможностями — чем мощнее сервер, тем больше услуг и с большим качеством он может предоставлять, поэтому в качестве сервера выбирается довольно мощный компьютер. Хотя эта формула (чем мощнее, тем лучше) не всегда оправдана, например, если ваш сервер используется для предоставления доступа к Интернет небольшой сети, то в этом случае с поставленной задачей прекрасно справится старенький 486DX/ 66 — 32 Мб ОЗУ. Однако, если вы являетесь Интернет-провайдером, то есть предоставляете коммерческий доступ к сети Интернет, такой конфигурации будет явно недостаточно.

Хотя установке Linux посвящена вторая глава, в которой подробно расписаны все рекомендуемые конфигурации, уже сейчас отмечу, что в случае с Linux-сервером объем оперативной памяти более критичен, чем частота процессора. Поэтому, если у вас есть возможность установить больше оперативной памяти, установите — не пожалеете.

Примечание.

Иногда увеличивать объем памяти нецелесообразно, потому что машина будет работать медленнее, чем до модернизации. Это может произойти, если вы используете некоторые старые чипсеты, которые не кэшируют объемы оперативной памяти более 64 Мб (или 128 Мб), а так как операционная система загружается в старшие адреса, освобождая младшие адреса для прикладных программ, общая производительность системы будет снижена. Перед модернизацией рекомендую ознакомиться с документацией на материнскую плату.

1.7,2. Протокол и интерфейс

Теперь пора уже перейти к протоколам, в частности, к протоколу TCP/IP, который лежит в основе сети Интернет. Протокол -- это совокупность правил, определяющая взаимодействие абонентов вычислительной системы

(в нашем случае — сети) и описывающая способ выполнения определенного класса функций. Еще один термин, который мы будем часто употреблять - интерфейс. Интерфейс — это средства и правила взаимодействия компонент системы между собой. Чтобы лучше понять значения этих терминов, обратите внимание на рис. 1.6. На этом рисунке изображены две системы (компьютера) — A и B.

Из рис. 1.6 видно, что средства, которые обеспечивают взаимодействие модулей разных уровней в рамках *одной системы* (например, B1 и B2), называются интерфейсом, а средства, обеспечивающие взаимодействие компонент одного уровня *разных систем* (например, A1 и B1), называются протоколом. Протокол и интерфейс можно сравнить еще и так: разговор двух директоров разных предприятий можно назвать протоколом, а разговор директора и подчиненного одного предприятия можно считать интерфейсом. Как вы уже догадались, разговор сотрудников разных предприятий будет протоколом.

Теперь, когда мы уже знаем, что означает слово «протокол», перейдем к рассмотрению основных протоколов.

Самым главным — святыней всех святынь — является протокол TCP/IP. TCP/IP (Transmission Control Protocol/Internet Protocol — Протокол Управления Передачей/Интернет-протокол) — это базовый транспортный сетевой протокол. На этом протоколе основана вся сеть Интернет.

Следующий важный протокол — это RIP (Routing Information Protocol). Протокол RIP используется для маршрутизации пакетов в компьютерных сетях. Для маршрутизации также используется протокол OSPF (Open Shortest Path First), который является брлее эффективным, чем RIP.

ICMP (Internet Control Message Protocol) — протокол межсетевых управляющих сообщений. Существует несколько типов данного протокола, которые используются для определенных целей (установление соединения, проверка доступности узла).

FTP (File Transfer Protocol) — протокол передачи файлов. Служит для обмена файлами между системами. Например, вам нужно передать файл на сервер или, наоборот, скачать файл с сервера. Для этого вам нужно подключиться к файловому серверу (он же FTP-сервер) и выполнить необходимую вам

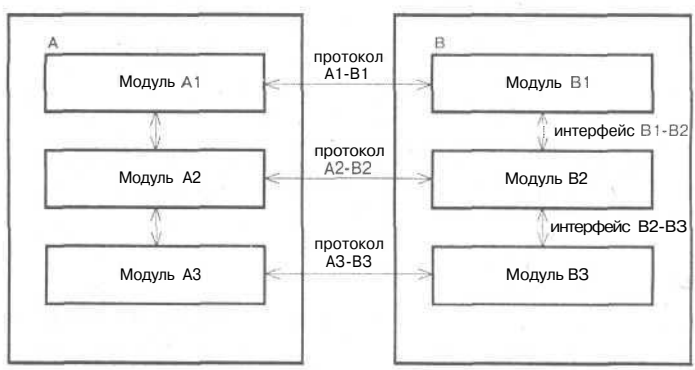


Рис. 1.6. *Протоколы и интерфейсы*

операцию. Подключение осуществляется с помощью FTP-клиента. Простейший FTP-клиент входит в состав практически любой операционной системы. Обычно для запуска FTP-клиента нужно ввести команду **ftp.**

HTTP (Hyper Text Transfer Protocol) — протокол обмена гипертекстовой информацией, то есть документами HTML. Протокол HTTP используется Web-серверами. HTTP-клиенты называются браузерами.

POP (Post Office Protocol) -- протокол почтового отделения. Этот протокол используется для получения электронной почты с почтовых серверов. А для передачи электронной почты служит протокол **SMTP** (Simple Mail Transfer Protocol) — протокол передачи сообщений электронной почты.

Раз уж затронули тему почтовых протоколов, давайте разберемся, как происходит чтение и отправление почты. Для получения сообщений пользователь соединяется с POP-сервером, сообщает ему свое *имя пользователя* и *пароль,* и, если аутентификация прошла успешно, получает сообщения. Обратите внимание, что пользователь именно *получает сообщения,* а не просматривает их. Прочитать сообщение пользователь может только после его загрузки на свой компьютер. Обычно полученные сообщения удаляются на сервере, но это зависит от настроек почтового клиента пользователя. Почтовый клиент — это программа, которая выполняет все операции по работе с электронной почтой. Самыми распространенными почтовыми клиентами являются The Bat!, Outlook, Outlook Express, Netscape Messenger, KMail.

Для передачи сообщения пользователь просто соединяется с SMTP-сервером и передает сообщение. При этом никакой аутентификации не происходит, хотя можно настроить сервер так, что он будет запрашивать имя пользователя и пароль перед началом передачи сообщения на сервер. Настройке SMTP-аутентификации посвящен п. 13.2 этой книги. После передачи сообщения на сервер SMTP оно становится в очередь. Через определенное время это сообщение передается нужному POP-серверу, который принимает сообщение. Потом сообщение может получить пользователь, для которого оно предназначено. Если же сервер SMTP не может отправить сообщение (например, нужный POP-сервер не существует или недоступен, или же адресат не зарегистрирован на этом сервере POP), письмо возвращается отправителю.

Для чтения почты существует и другой протокол — ШАР. Его отличие от протокола POP состоит в том, что пользователь читает сообщения электронной почты, не загружая их на свой компьютер. Все сообщения хранятся на сервере. При удалении сообщения оно удаляется с сервера.

SLIP (Serial Line Internet Protocol) — протокол подключения к сети Интернет по последовательной линии. Используется для установления связи с удаленными узлами через низкоскоростные последовательные интерфейсы. В настоящее время вытеснен протоколом PPP и практически не используется.

PPP (Point-to-Point Protocol) обеспечивает управление конфигурацией, обнаружение ошибок и повышенную безопасность при передаче данных на более высоком уровне, чем протокол SLIP. Поэтому при настройке сервера рекомендуется использовать именно этот протокол. Протокол PPP рассмотрен в RFC 1547 и RFC 1661.

Прежде чем перейти к рассмотрению протокола TCP/IP, рассмотрим семиуровневую модель взаимодействия открытых систем. Под открытой системой понимается любая система, построенная в соответствии с открытыми спецификациями. Протокол также можно рассматривать как определенное соглашение, принятое взаимодействующими объектами, в нашем случае — это компьютеры, работающие в сети. Соглашение (протокол) не обязательно должно быть стандартным, но на практике стараются использовать именно стандартные протоколы.

В начале 80-х годов международной организацией по стандартизации (ISO — International Organization for Standardization) была разработана *модель взаимодействия открытых систем* (OSI -- Open System Interconnection). В другой литературе вы можете встретить и другие названия этой модели: сокращенное - - модель OSI или более полное — семиуровневая модель взаимодействия открытых систем OSI. Средства взаимодействия (см. рис. 1.7) в модели OSI делятся на семь уровней:

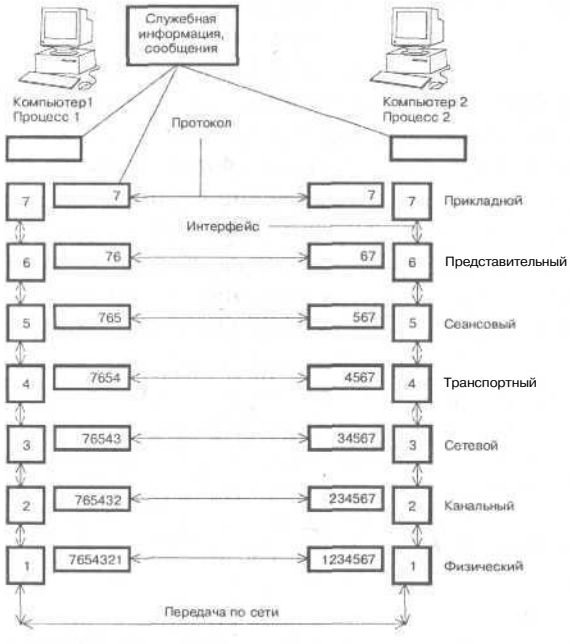


Рис. 1.7. *Модель OSI*

1. Физический.
2. Канальный.
3. Сетевой.
4. Транспортный.
5. Сеансовый.
6. Представительный.
7. Прикладной.

Благодаря этому задача сетевого взаимодействия разбивается на несколько более мелких задач. Это позволяет при разработке новых способов и инструментов сетевого взаимодействия не разрабатывать их заново целиком и полностью, а использовать уже готовые решения, заменив только некоторые его части. Непосредственно друг с другом взаимодействуют только физические уровни. Все остальные уровни напрямую взаимодействуют только с выше- и нижележащими уровнями: пользуются услугами нижележащего и предоставляют услуги вышележащему. Друг с другом такие уровни контактируют косвенным образом, через посредство нижележащих уровней.

Примечание.

6 *некоторых случаях сетевого взаимодействия физический уровень как таковой отсутствует, при этом его функции выполняет самый низлежащий уровень.*

Из рис. 1.7 видно, что по мере прохождения сообщения через уровни модели OSI к пересылаемым данным добавляется служебная информация, свидетельствующая о прохождении данных через определенный уровень.

Рассмотрим взаимодействие двух компьютеров более подробно на примере файловой службы. Допустим, нам (компьютер 1) нужно записать какую-нибудь информацию в файл на удаленном компьютере 2. Обычное сообщение состоит из заголовка и поля данных. В заголовке содержится различная служебная информация. Как изменяется заголовок видно из рис. 1.7. Например, в заголовке может содержаться информация о нашем компьютере (его адрес), компьютере получателя, а также имя и расположение файла, в который нужно записать информацию. Поле данных может быть и пустым, но в нашем случае, очевидно, содержит информацию, которую нужно записать в файл.

Приложение (процесс 1) формирует стандартное сообщение, которое передается прикладному уровню. Точнее, процесс 1 работает на прикладном уровне.

После формирования сообщения прикладной уровень передает его представительному уровню. На этом уровне в заголовок добавляются указания для представительного уровня компьютера-адресата. Потом сообщение передается сеансовому уровню, который добавляет свою информацию и т.д. Процесс вложения одного протокола в другой называется *инкапсуляцией*.

Когда сообщение поступает на компьютер-адресат, оно принимается физическим уровнем и передается вверх с уровня на уровень. Каждый уровень анализирует содержимое заголовка своего уровня, выполняет содержащиеся в нем указания, затем удаляет относящуюся к себе информацию из заголовка и передает сообщение далее вышележащему уровню. Этот процесс называется *декапсуляцией*. Далее приведено описание уровней взаимодействия.

Физический уровень (Physical Layer)

Физический уровень передает биты по физическим каналам связи, например, коаксиальному кабелю или витой паре. На этом уровне определяются характеристики электрических сигналов, которые передают дискретную информацию, например: тип кодирования, скорость передачи сигналов. К этому уровню также относятся характеристики физических сред передачи данных: полоса пропускания, волновое сопротивление, помехозащищенность.

Функции физического уровня реализуются сетевым адаптером или последовательным портом. Примером протокола физического уровня может послужить спецификация 100Base-TX (технология Ethernet).

Канальный уровень (Data link Layer)

Канальный уровень отвечает за передачу данных между узлами в рамках одной локальной сети. Узлом будем считать любое устройство, подключенное к сети.

Этот уровень выполняет адресацию по физическим адресам (MAC-адресам), «вшитым» в сетевые адаптеры предприятием-изготовителем. Каждый сетевой адаптер имеет свой *уникальный* MAC-адрес, то есть вы не найдете две сетевые платы с одним и тем же MAC-адресом.

Канальный уровень переводит поступившую с верхнего уровня информацию в биты, которые потом будут переданы физическим уровнем по сети. Он разбивает пересылаемую информацию на фрагменты данных — кадры (frames).

На этом уровне открытые системы обмениваются именно кадрами. Процесс пересылки выглядит примерно так: канальный уровень отправляет кадр физическому уровню, который отправляет кадр в сеть. Этот кадр получает каждый узел сети и проверяет, соответствует ли адрес пункта назначения адресу этого узла. Если адреса совпадают, канальный уровень принимает кадр и передает наверх вышележащим уровням. Если же адреса не совпадают, то он просто игнорирует кадр.

В используемых протоколах канального уровня заложена определенная топология. Топологией называется способ организации физических связей и способы их адресации. Канальный уровень обеспечивает доставку данных между узлами в сети с определенной топологией, то есть для которой он разработан. К основным топологиям (см. рис. 1.8) относятся:

- Общая шина.
- Кольцо.
- Звезда.

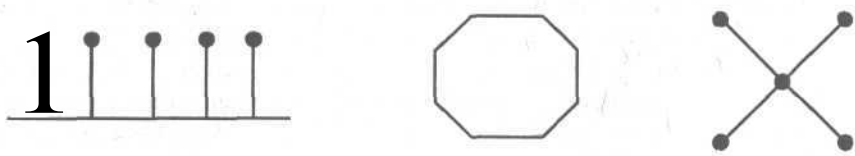


Рис. 1.8. *Основные топологии локальных компьютерных сетей*

Протоколы канального уровня используются компьютерами, мостами, маршрутизаторами. Глобальные сети (в том числе и Интернет) редко обладают регулярной топологией, поэтому канальный уровень обеспечивает связь только между компьютерами, соединенными индивидуальной линией связи. При этом для доставки данных через всю глобальную сеть используются средства сетевого уровня (протоколы «точка-точка»). Примерами протоколов «точка-точка» могут послужить PPP, LAP-B.

Сетевой уровень (Network Layer)

Данный уровень служит для образования единой транспортной системы, которая объединяет несколько сетей. Другими словами, сетевой уровень обеспечивает межсетевое взаимодействие.

Протоколы канального уровня передают кадры между узлами только в рамках сети с соответствующей топологией. Проще говоря - - в рамках одной сети.

Нельзя передать кадр канального уровня узлу, который находится в другой сети. Данное ограничение не позволяет строить сети с развитой структурой или сети с избыточностью связей. Построить одну *большую* сеть также невозможно из-за физических ограничений. К тому же даже если

построить довольно большую сеть (например, спецификация 10Base-T позволяет использовать 1024 узла в одном сегменте), производительность данной сети не будет вас радовать. Более подробно о причинах разделения сети на подсети и возникающих при этом трудностях мы поговорим немного позже, а сейчас продолжим рассматривать сетевой уровень.

На сетевом уровне термин *сеть* следует понимать как совокупность компьютеров, которые соединены в соответствии с одной из основных топологий и использующих для передачи данных один из протоколов канального уровня.

Сети соединяются специальными устройствами — *маршрутизаторами*. Маршрутизатор собирает информацию о топологии межсетевых соединений и на основании этой информации пересылает пакеты сетевого уровня в сеть назначения. Чтобы передать сообщение от компьютера-отправителя компьютеру-адресату, который находится в другой сети, нужно совершить некоторое количество *транзитных передач между сетями*. Иногда их еще называют хопами (от англ. hop — прыжок). При этом каждый раз выбирается подходящий маршрут.

Сообщения на сетевом уровне называются пакетами. На сетевом уровне работают несколько видов протоколов. Прежде всего — это сетевые протоколы, которые обеспечивают передвижение пакетов по сети, в том числе в другую сеть. Поэтому довольно часто к сетевому уровню относят протоколы маршрутизации (routing protocols) — RIP и OSPF.

Еще одним видом протоколов, работающих на сетевом уровне, являются протоколы разрешения адреса — Address Resolution Protocol (ARP). Хотя эти протоколы иногда относят и к канальному уровню.

Классические примеры протоколов сетевого уровня: IP (стек TCP/IP), IPX (стек Novell).

Транспортный уровень (Transport Layer)

На пути от отправителя к получателю пакеты могут быть искажены или утеряны. Некоторые приложения самостоятельно выполняют обработку ошибок при передаче данных, но большинство все же предпочитают иметь дело с надежным соединением, которое как раз и призван обеспечить транспортный уровень. Этот уровень обеспечивает требуемую приложению или верхнему уровню (сеансовому или прикладному) надежность доставки пакетов. На транспортном уровне определены пять классов сервиса:

1. Срочность.
2. Восстановление прерванной связи.
3. Наличие средств мультиплексирования нескольких соединений.
4. Обнаружение ошибок.
5. Исправление ошибок.

Обычно уровни модели OSI, начиная с транспортного уровня и выше, реализуются на программном уровне соответствующими компонентами операционных систем.

Примеры протоколов транспортного уровня: TCP и UDP (стек TCP/IP), SPX (стек Novell).

Сеансовый уровень (Session Layer)

Сеансовый уровень устанавливает и разрывает соединения между компьютерами, управляет диалогом между ними, а также предоставляет средства синхронизации. Средства синхронизации позволяют вставлять определенную контрольную информацию в длинные передачи (точки), чтобы в случае обрыва связи можно было вернуться назад (к последней точке) и продолжить передачу.

Сеанс - - это логическое соединение между компьютерами. Каждый сеанс имеет три фазы:

1. Установление соединения. Здесь узлы «договариваются» между собой о протоколах и параметрах связи.
2. Передача информации.
3. Разрыв связи.

Не нужно путать сеанс сетевого уровня с сеансом связи. Пользователь может установить соединение с Интернет, но не устанавливать ни с кем логического соединения, то есть не принимать и не передавать данные.

Представительный уровень (Presentation Layer)

Представительный уровень изменяет форму передаваемой информации, но не изменяет ее содержания. Например, средствами этого уровня может быть выполнено преобразование информации из одной кодировки в другую. Также на этом уровне выполняется шифрование и дешифрование данных.

Пример протокола представительного уровня: SSL (Secure Socket Layer). Данный протокол обеспечивает секретный обмен данными.

Прикладной уровень (Application Layer)

Данный уровень представляет собой набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к совместно используемым ресурсам. Единица данных называется сообщением.

Примеры протоколов: HTTP, FTP, TFTP, SMTP, POP, SMB, NFS.

Интернет и модель OS/

При взаимодействии открытой системы и Интернет модель OSI упрощается, так как некоторые протоколы Интернет включают в себя функции нескольких уровней. Если к сети Интернет подключается один пользователь, а не вся сеть, то автоматически исчезают канальный и физический уровни, потому что нет сетевых адаптеров, а значит, нет и физических адресов. В данном случае конечным протоколом будет протокол типа «точка-точка», например, PPP. В этот протокол будут вложены все остальные.

1.7.3. Протокол TCP/IP

В этом разделе давайте рассмотрим, как передается информация в TCP/IP-сети. Любая информация передается небольшими порциями, которые называются пакетами. Если нужный объем информации нельзя передать одним пакетом, он разбивается на части. В заголовке каждого пакета указывается IP-адрес отправителя и IP-адрес получателя, а также номер порта.

Любому компьютеру в IP-сети (TCP/IP-сети) назначен уникальный адрес, который называется IP-адресом. **IP-адрес** — это 32-разрядное число, которое принято записывать в десятеричном или шестнадцатеричном формате в виде четырех чисел, разделенных точками, например:

1. 111.111.213.232
2. 127.0.0.1
3. 192.168.9.2

При условии, что ваша сеть подключена к Интернет, протокол TCP/IP обеспечивает работу вашей сетевой программы с любым компьютером в мире, как будто тот находится в локальной сети. Уникальность IP-адреса достигается достаточно просто — IP-адреса назначаются централизованно **Сетевым Информационным Центром** (NIC, Network Information Center).

Для понимания остальной информации нужно отметить, что существуют локальные (LAN, Local Area Networks) и региональные (Wide Area Networks) сети. Сеть Интернет сначала была региональной (Arpanet), а потом стала глобальной, объединив все региональные сети мира. Если ваша локальная (или даже региональная) сеть не соединена с Интернет, то внутри сети вы можете использовать любые IP-адреса без согласования с NIC. Обычно в локальных сетях используются особые IP-адреса, о которых мы поговорим немного позже.

Любую сеть, независимо от типа — LAN или WAN, можно разделить на подсети. Причины разбиения сети на подсети кроются в ранних версиях протокола IP. Тогда существовало несколько сетей класса A, содержащих несколько миллионов узлов (о классах читайте далее). Помимо всего прочего, в таких сетях очень велика вероятность коллизий, то есть одновременного доступа двух или более узлов к среде передачи данных. Управлять такой сетью крайне неудобно, да и сеть будет перегружена собственным трафиком. Поэтому основной принцип разделения — «разделяй и властвуй».

К другим причинам разделения относят создание маленьких подсетей с использованием разных технологий — Ethernet, Token Ring, FDDI, ATM. Вы не можете смешивать эти технологии в одной сети, однако они могут быть взаимосвязаны с помощью разделения на подсети.

Разделение на подсети может быть также произведено из соображений безопасности. Более подробно об этой и других причинах разделения сети на подсети вы можете прочитать в руководстве IP Sub-networking-HOWTO, которое вы найдете на прилагаемом компакт-диске.

Как я уже писал, каждый компьютер в сети имеет свой уникальный адрес. Но оказывается, что и сеть (подсеть) также имеет свой уникальный адрес. Под сетью можно понимать «пачку» IP-адресов, идущих подряд, то есть 192.168.1.0...192.168.1.255. Самый младший и самый старший адреса резервируются. Младший (192.168.1.0) является адресом сети, а старший является широковещательным (broadcast) адресом сети. Адрес сети может потребоваться, когда нужно указать всю сеть (подсеть), например, при задании маршрутизации для этой сети.

Представьте, что у вас есть две отдельных сети и вам нужно объединить их в одну. Тогда эта одна «большая» сеть станет называться сетью, а две

«маленькие» — подсетями. Устройство, которое будет обеспечивать связь этих сетей (маршрутизацию), называется, как уже было отмечено выше, маршрутизатором. Маршрутизатор может быть как аппаратным (отдельное устройство), так и программным.

В роли программного маршрутизатора может выступать любой компьютер с двумя (или более) сетевыми интерфейсами, например, двумя сетевыми платами. В качестве операционной системы может быть установлена любая сетевая операционная система, поддерживающая перенаправление пакетов IPv4-Forwarding. Такой операционной системой может быть Linux, FreeBSD, любая UNIX-система, Windows NT/2000. Маршутизатор можно настроить и на базе Windows 98, но делать это я не рекомендую, поскольку вряд ли он будет работать надежно. Традиционно в роли маршрутизатора используются UNIX-системы, к которым относится и Linux.

Широковещательный адрес используется для передачи сообщений «всем — всем — всем» в рамках сети, то есть когда нужно передать сообщение (пакет) сразу всем компьютерам сети. Широковещательные запросы очень часто используются, например, для построения ARP-таблиц.

Для каждой подсети определена ее маска. Фактически, маска — это размер сети, • то есть число адресов в сети. Маску принято записывать в десятично-побайтном виде:

255.255.255.0.....маска на 256 адресов (0...255);
255.255.255.192.....маска на 64 адреса (192...255);
255.255.0.0.....маска на 65536 адресов (256*256).

В общем случае IP-сети делятся на пять классов: A, B, C, D **и** E.

Сети класса A — это огромные сети. Маска сети класса A: 255.0.0.0. В каждой сети такого класса может находиться 16777216 адресов. Адреса таких сетей лежат в промежутке 1.0.0.0...126.0.0.0, а адреса хостов (компьютеров) имеют вид 125.*.*.*

Сети класса B — это средние сети. Маска такой сети — 255.255.0.0. Эта сеть содержит 65536 адресов. Диапазон адресов таких сетей 128.0.0.0...191.255.0.0. Адреса хостов имеют вид 136.12.*.*

Сеть класса C — маленькие сети. Содержат 256 адресов (на самом деле всего 254 хоста, так как номера 0 и 255 зарезервированы). Маска сети класса C – 255.255.255.0. Интервал адресов: 192.0.1.0...223.255.255.0. Адреса хостов имеют вид: 195.136.12.*

Класс сети определить очень легко. Для этого нужно перевести десятичное представление адреса сети в двоичное. Например, адрес сети 128.11.1.0 в двоичном представлении будет выглядеть так:

10000000 00001011 00000001 00000000

а сети 192.168.1.0:

11000000 10101000 00000001 00000000

Если адрес начинается с последовательности битов 10, то данная сеть относится к классу B, а если с последовательности ПО, то — к классу C.

Если адрес начинается с последовательности 1110, то сеть является сетью класса D, а сам адрес является особым — групповым (multicast). Если в пакете указан адрес сети класса D, то этот пакет должны получить все хосты, которым присвоен данный адрес.

Адреса класса E зарезервированы для будущего применения. В табл. 1.2 приведены сравнительные характеристики сетей классов A, B, C, D и E.

Характеристики сетей различных классов *Таблица 1.2*

Класс	Первые биты	Диапазон адресов	Количество узлов
A	0	1.0.0.0...126.0.0.0	16777216 (224)
B	10	128.0.0.0...191.255.0.0	65536 (216)
C	110	192.0.1.0...223.255.255.0	256 (28)
D	1110	224.0.0.0...239.255.255.255	Multicast
E	11110	240.0.0.0...247.255.255.255	Зарезервирован

Теперь самое время немного сказать о специальных адресах, о которых я упомянул немного выше. Если весь IP-адрес состоит из нулей (0.0.0.0), то значит, что он обозначает адрес того узла, который сгенерировал этот пакет.

Адрес 255.255.255.255 — это широковещательный адрес. Пакет с таким адресом будет рассылаться всем узлам, которые находятся в той же сети, что и источник пакета. Это явление называется ограниченным широковещанием. Существует также другая рассылка, которая называется широковещательным сообщением. В этом случае вместо номера узла стоят все единицы в двоичном представлении (255). Например, 192.168.2.255. Это означает, что данный пакет будет рассылаться всем узлам сети 192.168.2.0.

Особое значение имеет IP-адрес 127.0.0.1 — это адрес локального компьютера. Он используется для тестирования сетевых программ и взаимодействия сетевых процессов. При попытке отправить пакет по этому адресу данные не передаются по сети, а возвращаются протоколам верхних уровней, как только что принятые. При этом образуется как бы «петля». Этот адрес называется loopback. В IP-сети запрещается использовать IP-адреса, которые начинаются со 127. Любой адрес подсети 127.0.0.0 относится к локальному компьютеру, например: 127.0.0.1, 127.0.0.5, 127.77.0.6.

Существует также специальные адреса, которые зарезервированы для несвязанных локальных сетей — это сети, которые используют протокол IP, но не подключены к Интернет. Вот эти адреса:

10.0.0.0 (сеть класса A, маска сети 255.0.0.0).

172.16.0.0...172.31.0.0 (16 сетей класса B, маска каждой сети 255.255.0.0).

192.168.0.0...192.168.255.0 (256 сетей класса C, маска каждой сети 255.255.255.0).

В этой книге я старался использовать именно такие адреса, чтобы не вызвать пересечение с реальными IP-адресами.

1.7.4. Система доменных имен — DNS

Для того чтобы подключиться к какому-нибудь другому компьютеру, например, Web-серверу, нужно знать его IP-адрес. Это не очень удобно, потому что человеку намного проще запомнить символьное название сервера,

чем последовательность чисел. Представьте, что вместо http://www.romb.net в окне браузера вам нужно было бы вводить http://62.244.59.193. Оба способа будут работать, но первый запоминается намного проще. Фактически, нужно запомнить только слово из четырех букв — romb, а www и net — это «само собой». Компьютеру же, наоборот, проще обрабатывать числа, а не символьную информацию.

Для преобразования IP-адреса в символьное имя и обратно используется служба доменных имен — DNS (Domain Name System). Обычно на любом сервере устанавливается своя служба DNS, даже если этот сервер не поддерживает домена. В отличие от одноранговой сети, в IP-сети компьютеры объединяются в домены, а не в рабочие группы. На самом деле, понятие домен гораздо шире, чем рабочая группа, но пока остановимся на таком определении.

Предположим, адрес Web-сервера вашего подразделения выглядит так: http://www.department.firma.isp.ru. Рассмотрим, что происходит, когда пользователь вводит в окне браузера этот адрес. Сначала отправляется запрос на разрешение (преобразование) имени в IP-адрес серверу DNS, который принадлежит провайдеру пользователя. Если такое имя есть в кэше DNS-сервера провайдера (для определенности назовем его user-dns), он возвращает IP-адрес и браузер устанавливает соединение с этим компьютером. Если же такого адреса в кэше сервера DNS не оказалось, DNS-сервер провайдера обращается к серверу, который содержит домен наивысшего уровня, то есть к корню дерева (см. рис. 1.9). Тот обращается к домену ru (этот сервер пусть называется ru-dns). Сервер ru-dns в свою очередь обращается к серверу, который делегирует домен isp (это ваш провайдер). Сервер isp обращается к серверу, который делегирует (администрирует) домен firma, а он уже к серверу, отвечающему за домен department, который и возвращает IP-адрес компьютера www.department.firma.isp.ru. Таким образом, получается своеобразная цепочка. Ясно, что если эта цепочка оборвется на каком-нибудь звене, то пользователю, точнее, серверу DNS user-dns, будет сообщено о невозможности разрешения имени компьютера в IP-адрес.

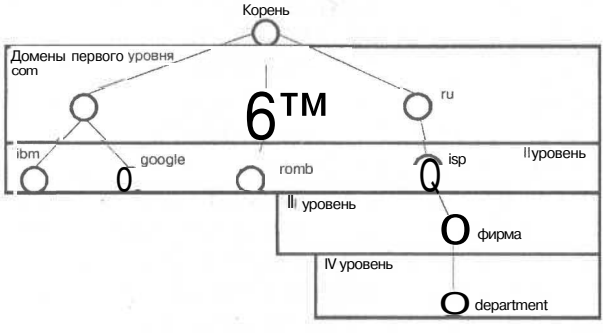


Рис. 1.9. *Иерархическая структура системы доменных имен*

Вся структура службы DNS является иерархической. Существуют домены первого, второго, третьего, n-го уровней. В рассмотренном примере доменом первого уровня является ru, isp — второго, firma — третьего, а department — четвертого уровня (см. рис. 1.9).

Корневой домен управляется центром InterNIC. Домены верхнего (первого) уровня назначаются для каждой страны (см. табл. 1.3).

Обозначения стран по стандарту ISO 3166 *Таблица 1.3*

Домен	Страна	Домен	Страна
ru	Россия	ua	Украина
by	Белоруссия	lt	Литва
lv	Латвия	ee	Эстония
md	Молдова	kz	Казахстан
tr	Турция	ro	Румыния
iq	Ирак	ir	Иран
il	Израиль	tm	Туркменистан
pl	Польша	it	Италия
es	Испания	gb	Великобритания
fr	Франция	de	Германия
id	Индонезия	vn	Вьетнам
gr	Греция	va	Ватикан
at	Австрия	co	Колумбия
hu	Венгрия	mx	Мексика

Для США и Канады единый домен отсутствует, но иногда используется обозначение us. Обозначения стран соответствуют международному стандарту ISO 3166. Данные сведения могут быть получены по адресу **ftp:/ftp.ripe.net/iso3166-countrycodes**.

Для различных типов организаций могут использоваться такие обозначения:

com.....коммерческие организации (например, yahoo.com).
edu.....образовательные учреждения (например, mit.edu).
gov.....правительственные организации (например, nasa.gov).
org.....некоммерческие организации (например, linux.org).
net.....обычно провайдеры (например, ukr.net).

По данным ISC (Internet Software Consortium) по состоянию на январь 2002 года зарегистрировано около 150 миллионов узлов сети Интернет (см. рис. 1.10). Данные сведения публикуются с разрешения ISC.

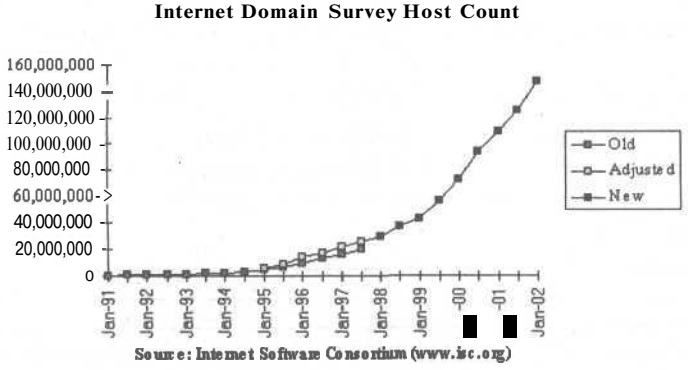


Рис. *1.10. Динамика роста узлов в сети Интернет*

1.7.5. Многоуровневая архитектура стека TCP/IP

Этот пункт книги является необязательным: если вы считаете, что у вас уже достаточно знаний о протоколе TCP/IP, то можете перейти к следующим разделам, а к этому вернуться позже. Здесь будет описана многоуровневая архитектура протокола TCP/IP — для большего понимания происходящего.

Вначале давайте рассмотрим историю создания протокола TCP/IP. Протокол TCP/IP был создан в конце 60-х — начале 70-х годов агентством DARPA Министерства Обороны США (U.S. Department of Defense Advanced Research Projects Agency). Основные этапы развития этого протокола отмечены в табл. 1.4.

Этапы развития протокола TCP/IP *Таблица 1.4*

Год	Событие
1970	Введен в использование протокол NCP (Network Control Protocol) для узлов сети Arpanet
1972	Вышла первая спецификация Telnet (см. RFC 318)
1973	Введен протокол FTP (RFC 454)
1974	Программа TCP (Transmission Control Program)
1981	Опубликован стандарт протокола IP (RFC 791)
1982	Объединение протоколов TCP и IP в одно целое — TCP/IP
1983	Сеть Arpanet переведена на протокол TCP (ранее использовался протокол NCP)
1984	Введена доменная система имен DNS

Как вы видите, все стандарты Интернет-протоколов опубликованы в документах RFC. **Документы RFC** (Request for Comments) -- это запрос комментариев. В этих документах описывается устройство сети Интернет.

Документы RFC создаются сообществом Интернет (Internet Society, ISOC). Любой член ISOC может опубликовать свой стандарт в документе RFC. Документы RFC делятся на пять типов:

Требуется (Required)..... данный стандарт должен быть реализован на всех основных узлах TCP/IP.

Рекомендуется (Recommended)..... обычно такие спецификации RFC также реализуются.

Выборочно (Elective)..... реализация не обязательна.

Ограниченное использование (Limited use)..... не рекомендуется для всеобщего применения.

Не рекомендуется (Not recommended)..... не рекомендуется.

Все необходимые документы RFC вы найдете на прилагаемом компакт-диске.

Протоколы семейства TCP/IP можно представить в виде модели, состоящей из четырех уровней: прикладного, основного, межсетевого и сетевого (см. рис. 1.11).

Уровень 1	Прикладной уровень (уровень приложения, Application Layer)
Уровень 2	Основной (транспортный) уровень (Transport Layer)
Уровень 3	Межсетевой уровень (уровень Internet, Internet Layer)
Уровень 4	Уровень сетевых интерфейсов (Network Interface Layer)

Рис. 1.11. *Уровни стека протоколов TCP/IP*

Каждый из этих уровней выполняет определенную задачу для организации надежной и производительной работы сети.

Уровень сетевого интерфейса

Данный уровень лежит в основании всей модели протоколов семейства TCP/IP. Уровень сетевого интерфейса отвечает за отправку в сеть и прием из сети кадров, которые содержат информацию. Кадры передаются по сети как одно целое. *Кадр* (frame) — это единица данных, которыми обмениваются компьютеры в сети Ethernet. Для обозначения блоков данных определенных уровней используют термины кадр (frame), пакет (packet), дейтаграмма (datagram), сегмент (segment). Все эти термины обозначают транспортируемые отдельно блоки данных и их можно считать синонимами. Название блока пересылаемых данных изменяется в зависимости от уровня (см. рис. 1.12).

*Рис. **1.12.** Пересылка блока данных в стеке протоколов TCP/IP*

Межсетевой уровень

Протоколы Интернет инкапсулируют блоки данных в пакеты (дейтаграммы) и обеспечивают необходимую маршрутизацию. К основным Интернет-протоколам относятся:

IP (Internet Protocol) .... предназначен для отправки и маршрутизации пакетов.

ARP (Address Resolution Protocol) ... используется для получения MAC-адресов (аппаратных адресов) сетевых адаптеров.

ICMP (Internet Control Message Protocol) ... предназначен для отправки извещений и сообщений об ошибках при передаче пакетов.

IGMP (Internet Group Management Protocol)... используется узлами для сообщения маршрутизаторам, которые поддерживают групповую передачу, о своем участии в группах.

RIP (Route Internet Protocol) и OSPF (Open Shortest Path First) ... протоколы маршрутизации.

На этом уровне реализуется передача пакетов без установки соединения — дейтаграммным способом. Межсетевой уровень обеспечивает перемещение пакетов по сети с использованием наиболее рационального маршрута (протокол OSPF). Основная функция межсетевого уровня — передача пакетов через составную сеть, поэтому этот уровень также называется уровнем Интернет.

Транспортный (основной) уровень

Данный уровень обеспечивает сеансы связи между компьютерами. Существует два транспортных протокола: TCP (Transmission Control Protocol) и UDP (User Datagram Protocol). Протокол TCP ориентирован на установление соединения, то есть перед передачей данных компьютеры «договариваются» между собой. Обычно по этому протоколу передаются большие объемы данных или данные, для которых требуется подтверждение их приема. Этот протокол используется большинством сетевых приложений, так как обеспечивает достаточную надежность при передаче данных.

Протокол UDP не ориентирован на соединение и не гарантирует доставку пакетов (дейтаграмм). Однако протокол UDP является более быстродействующим по сравнению с TCP. Обычно по этому протоколу передаются небольшие объемы данных. Ответственность за доставку данных несет сетевая программа.

Уровень приложений

Данный уровень является вершиной модели TCP/IP. На этом уровне работают практически все распространенные утилиты и службы: DNS, Telnet, WWW, Gopher, WAIS, SNMP, FTP, TFTP, SMTP, POP, IMAP.

В качестве завершения данного пункта рассмотрим соответствие уровней стека протокола TCP/IP семиуровневой модели OSI (см. табл. 1.5).

Соответствие уровней стека TCP/IP модели OSI *Таблица 1.5*

Уровень модели OSI	Протокол	Уровень стека TCP/IP
7, 6	WWW (HTTP), FTP, TFTP, SMTP, POP, telnet, WAIS, SNMP	1
5, 4	TCP, UDP	2
3	IP, ICMP, RIP, OSPF, ARP	3
2, 1	Ethernet, PPP, SLIP	4

В следующем пункте рассмотрено такое важное понятие протокола TCP/IP как порт. В том же пункте будут рассмотрены структуры пакетов IP и TCP, поскольку рассмотрение этого материала без введения определения порта не имеет смысла.

1.7.6. Порты и демоны

Дальнейшее изложение материала построено, исходя из того, что вы уже знаете, что такое сервер и какие службы вам придется настраивать. В пункте *Как устроена книга* (п. 1.5) было подробно описано, в каких главах описана настройка той или иной службы. Здесь же давайте рассмотрим некоторые базовые понятия, каковыми являются понятие «демон» и «порт».

Как уже было отмечено, в заголовке каждого пакета указывается IP-адрес отправителя и IP-адрес получателя, а также номер порта. С IP-адресом отправителя и получателя все понятно, осталось сказать, что же такое порт. Дело в том, что сразу несколько приложений на одном компьютере могут осуществлять обмен данными через сеть. При этом, если в качестве адресата указывать только IP-адрес получателя, то приложения, выполняемые на нем, не смогут разобраться кому из них предназначены присланные данные. Чтобы

решить эту проблему используется механизм портов. Номер порта — это просто номер программы, которая будет обрабатывать переданные данные. Каждой сетевой программе, которая работает по протоколу TCP/IP, сопоставлен свой номер порта. Например, 80 — это порт WWW-сервера (обычно это Apache), а 53 — это порт системы доменных имен.

Термин **демон** происходит от английского слова demon (или daemon) и означает программу, которая выполняется в фоновом режиме и дополняет операционную систему каким-нибудь сервисом. Как правило, пользователь не замечает работу демона: он даже и не подозревает, что данная программа запущена. Программа-демон чем-то напоминает резидентные программы в операционной системе DOS. Как видите, нет ничего общего с ужасными существами потустороннего мира. Обычно демон ожидает определенного события, после которого он активизируется и выполняет свою работу. Сетевые демоны ожидают получения пакета с определенным номером порта и, получив его, обрабатывают содержащиеся в нем данные. В книге мы еще неоднократно поговорим об этих «существах», поэтому сейчас не будем подробно останавливаться на них.

1.7.7. Структура пакетов IP и TCP

Вот теперь можно смело перейти к рассмотрению структуры пакетов IP и TCP. Протокол IP не ориентирован на соединение, поэтому не обеспечивает надежную доставку данных. Поля, описание которых приведено в табл. 1.6, представляют собой IP-заголовки и добавляются к пакету при его получении с транспортного уровня.

Структура заголовка IP-пакета *Таблица 1.6*

Поле	Описание
Source IP-address (IP-адрес отправителя)	Отправитель пакета
Destination IP-address (IP-адрес получателя)	Получатель пакета
Protocol (Протокол)	TCP или UDP
Checksum (Контрольная сумма)	Значение для проверки целостности пакета
TTL (Time to Live, время жизни пакета)	Определяет, сколько секунд дейтаграмма может находиться в сети. Предотвращает бесконечное блуждание пакетов в сети. Значение TTL автоматически уменьшается на одну или более секунд при проходе через каждый маршрутизатор сети
Version	Версия протокола IP — 4 или 6. Шестая версия протокола IP рассматривается в гл. 8 (4 бита)
Header Length (Длина заголовка)	Минимальный размер заголовка — 20 байт (4 бита)
Type of Service (Тип обслуживания)	Обозначение требуемого для этого пакета качества обслуживания при доставке через маршрутизаторы IP-сети. Здесь определяются приоритет, задержки, пропускная способность. (8 бит)
Total Length (Общая длина)	Длина дейтаграммы IP-протокола (16 бит)
Identification (Идентификация)	Идентификатор пакета. Если пакет фрагментирован (разбит на части), то все фрагменты имеют одинаковый идентификатор (16 бит)
Fragmentation Flags (Фрагментационные флаги)	3 бита для флагов фрагментации и 2 бита для текущего использования
Fragmentation Offset (Смещение фрагмента)	Указывает на положение фрагментов относительно начала поля данных IP-пакета. Если фрагментации нет, смещение равно 0x0 (13 бит)
Options and Padding (Опции и заполнение)	Опции

Протокол TCP, в отличие от протокола IP, ориентирован на установление соединения и обеспечивает надежную доставку данных. Структура TCP-пакета описана в табл. 1.7.

Структура TCP-пакета *Таблица 1.7*

Поле	Описание
Source port (Порт отправителя)	Порт TCP узла-отправителя
Destination Port (Порт получателя)	Порт TCP узла-получателя
Sequence Number (Порядковый номер)	Номер последовательности пакетов
Acknowledgement Number (Номер подтверждения)	Порядковый номер байта, который локальный узел рассчитывает получить следующим
Data Length (Длина данных)	Длина TCP-пакета
Reserved (Зарезервировано)	Зарезервировано для будущего использования
Flags (Флаги)	Описание содержимого сегмента
Window (Окно)	Показывает доступное место в окне протокола TCP
Checksum (Контрольная сумма)	Значение для проверки целостности пакета
Urgent Pointer (Указатель срочности)	При отправке срочных данных (поле Flags) в этом поле задается граница области срочных данных

1.8. Общие рекомендации

Поздравляю с успешным окончанием курса молодого бойца (администратора)! Осталось только сказать несколько слов об общей настройке сервера. Настройку сервера «с нуля» следует проводить именно в такой последовательности, которая описана в книге. После установки системы следует разобраться с правами пользователей, определить, кто и к чему будет иметь доступ. Именно с этого и начинается локальная безопасность сервера. Нет смысла настраивать все службы с соблюдением всех требований безопасности, если пользователь *pupkin* имеет доступ ко всему и его пароль *123!* (или же установлены права доступа к корневой файловой системе, позволяющие всем модифицировать ее).

Я рекомендую использовать RAID-массивы, если это, конечно, позволяют ваши финансы. Использование массивов RAID значительно повысит надежность вашего сервера. К тому же, при использовании аппаратных массивов, настройка которых не рассматривается в этой книге, можно существенно повысить производительность сервера, особенно при использовании SCSI-дисков. Хотя в последнее время производительность некоторых моделей ATA-дисков (ATA133) практически не уступает производительности дисков с интерфейсом SCSI, а по цене ATA-диски дешевле, чем SCSI.

После того, как вы наведете порядок на локальном уровне, можно приступать к настройке сети. Сначала следует сконфигурировать сервер как обыкновенную рабочую станцию и проверить корректность работы компьютера в сети. Всему этому посвящена гл. 7.

В гл. 8 описана настройка сервера. Вне зависимости от того, какой сервер вы настраиваете -- почтовик или сервер баз данных, вам нужно

сначала настроить суперсервер -- **xinetd** или же **inetd.** В восьмой главе объясняется, что такое суперсервер и как его нужно настраивать. Настройка фундаментальных серверных служб описана в той же главе. Однако это не означает, что вам нужно настраивать все, например, вы можете не использовать сетевую файловую систему или протокол **DHCP.** Существует простое правило: если вам не нужна какая-нибудь служба, просто отключите ее, но ни в коем случае не оставляйте ее ненастроенной — это потенциальная «дыра» в вашем сервере.

Все дальнейшие главы книги посвящены конфигурированию популярных сетевых служб, которые сейчас повсеместно используются. После настройки всех необходимых вам служб рекомендуется перекомпилировать ядро, удалив из него ненужный код. Это повысит производительность сервера и сделает ядро более компактным.

1.9. Обзор дистрибутивов Linux

Сейчас на отечественном рынке можно купить практически любой дистрибутив Linux, а самый новый или какой-нибудь малораспространенный можно заказать через Интернет-магазин, например, www.linuxcenter.ru. Самыми распространенными на сегодняшний день являются дистрибутивы:

1. Red Hat Linux.
2. Linux Mandrake.
3. ALT Junior Linux.
4. ASP Linux.
5. Black Cat Linux.
6. Slackware.
7. Astaro Security Linux.
8. SuSE Linux.

В книге я подробно рассмотрел дистрибутивы Red Hat Linux и Linux Mandrake, указал отличия между ними. Однако я не буду агитировать вас использовать Mandrake или Red Hat, наоборот, вы можете использовать эту книгу для настройки любого дистрибутива. Если что-то не будет работать, например, в SuSE или Slackware, прочитайте поставляемую с дистрибутивом документацию, там наверняка найдутся ответы на все ваши вопросы. Несмотря на то, что большинство дистрибутивов, приведенных выше, совместимо с Red Hat, у каждого дистрибутива есть определенные отличия, поэтому не верьте надписям на компакт-дисках наподобие «100%-ная совместимость с Red Hat». Каждый дистрибутив имеет свое предназначение, например, ALT Junior больше подходит для домашнего компьютера, чем для сервера.

Итак, начнем наш обзор с дистрибутива **Red Hat Linux.** Бесспорно, это один из самых популярных и распространенных дистрибутивов. «Красная шапочка» (именно так переводится название дистрибутива) является своеобразным образцом в мире Linux. Не зря же, когда сравнивают дистрибутивы, говорят об их совместимости с Red Hat.

Прежде всего, следует отметить простоту установки этого дистрибутива. Программа установки Red Hat отличается интуитивно-понятным интерфейсом и богатыми функциональными возможностями. При этом, как и для многих современных дистрибутивов, установка может выполняться как в графическом, так и в текстовом режимах. Кстати, первую графическую программу установки Linux, если я не ошибаюсь, предложила именно компания Red Hat. Должен заметить, что установка системы в текстовом режиме выполняется немного быстрее, чем в графическом.

После установки вы получаете практически функционирующую систему: все в ней настроено и работает, остается настроить систему «под себя». Конечно, есть небольшие недоработки, но разработчики не могли же предусмотреть всех вариантов?

На сегодняшний день последними версиями являются версии 7.2(7.3) и 8.0.. По сравнению с предыдущими версиями программа установки стала еще более простой, гибкой и функциональной. Например, появилась возможность сохранения параметров установки, что позволяет установить точно такую же конфигурацию системы на другие компьютеры сети. При установке осуществляется конвертирование файловой системы *ext2* в *ext3*. Переход на новую файловую систему обеспечивает более надежную ее работу.

Среди нововведений можно отметить оконную среду Gnome 1.4 и файловый менеджер Nautilus, который существенно облегчает операции по копированию, перемещению и удалению файлов.

Все параметры системы полностью настраиваются. Можно даже выставить уровень сложности интерфейса пользователя: от новичка до эксперта. Также нужно отметить, что исчезла грань между локальными и сетевыми ресурсами.

Настройка устройств осуществляется теперь намного проще благодаря новой утилите конфигурирования системы. Теперь все конфигураторы системы собраны в одной оболочке, которая называется **Контрольной панелью.** Расширена поддержка устройств, в том числе добавлена поддержка устройств USB и Firewire.

Особое внимание уделено безопасности системы: firewall теперь настраивается в процессе установки системы, а графическая утилита конфигурирования значительно упрощает процесс создания цепочек.

Следующий на очереди — дистрибутив **Linux Mandrake.** Linux Mandrake — это мощная операционная система для платформ Intel Pentium, AMD Athlon и PowerPC. Эта операционная система прекрасно сочетает в себе мощь Linux и простой интерфейс пользователя. ОС Mandrake как нельзя лучше подходит для начинающего пользователя, при этом ее можно устанавливать на компьютеры практически любого типа — от домашнего ПК до сервера сети, чего нельзя сказать о других дистрибутивах, которые имеют более четкое применение.

Программа установки по своей простоте, наверное, обогнала Red Hat. После установки система нормально работает, так как автоматически устанавливаются драйверы для большинства устройств (кроме программных win-устройств). Нет даже небольших недоработок со стороны программы установки.

Как я уже говорил, операционная система Linux Mandrake подойдет и для сервера, и для рабочей станции. Разработчики позаботились о системных администраторах, снабдив систему большим числом конфигураторов, с помощью которых можно настроить все сервисы сервера. Не забыли они и о пользователях: в состав дистрибутива входят распространенные оконные среды, офисные пакеты, графические редакторы, браузеры, MP3-плейеры.

В версии 9.0 произошли такие изменения:

1. Добавлена возможность минимальной установки. Для этого потребуется всего 64 Мб (!) на жестком диске.
2. Улучшены процедуры обнаружения оборудования.
3. Поддерживаются большие объемы ОЗУ (более 1 Гб) и многопроцессорность. Если у вас установлен всего один процессор и объем ОЗУ меньше 1 Гб, в целях повышения производительности системы я рекомендую перекомпилировать ядро системы, отключив поддержку этих функций.
4. Поддерживаются журналируемые файловые системы EXT3, ReiserFS, XFS и JFS.
5. Добавлена поддержка таких устройств: Firewire, USB, USB2, i830 DRM, ATA133, GeForce3.
6. Переработан Центр Управления (Control Center).

Система основана на ядре версии 2.4.19. В состав дистрибутива входят такие приложения:

1. Среда KDE 3.0.3 с интегрированным офисным пакетом K Office.
2. Среда GNOME 2.0.1 плюс Evolution 1.0.8 , WindowMaker 0.8, IceWM 1.2, Enlightenment 0.16.5, BlackBox 0.62
3. Офисные пакеты StarOffice 6.0 и KOffice 1.2. Оба пакета поддерживают формат MS Office.
4. Браузеры Mozilla 1.1, Konqueror 3.02 и Galeon 1.2.5.
5. Графический пакет GIMP 1.2.3.
6. Компиляторы GCC 3.2, библиотеки Glibc 2.2.5.
7. Сервер Apache 1.3.26.
8. Интерпретатор PHP 4.2.3.
9. Серверы баз данных MySQL 3.23.52 и PostgreSQL 7.2.2
10. Агенты MTA Sendmail и Postfix.

ALT Junior Linux — однодисковый дистрибутив для домашних и начинающих пользователей. Этот дистрибутив можно использовать как дистрибутив для рабочей станции, но не как серверный дистрибутив. В его состав входит все необходимое программное обеспечение для домашнего компьютера, в том числе есть несколько довольно неплохих игрушек.

Дистрибутив ALT Junior 1.0 основывается на ядре 2.4.5, поддерживает большинство современных чипсетов, графические 3D-ускорители Matrox G-серии, ATI Rage & ATI Radeon, Intel 810/815, 3DFX Voodoo 3/4/5. Добавлена поддержка всех видеоплат nVidia.

ASPLinux 7.2 — это еще один универсальный дистрибутив, который подойдет как для сервера, так и для рабочей станции. Отдельно хочу отметить уни-

кальную программу установки. Помимо своей простоты, она обладает некоторыми функциями, которых мне не хватало в других дистрибутивах:

1. Возможность установки основных пакетов (ядро и библиотеки) под конкретный процессор, что повышает производительность системы и не требует перекомпиляции ядра после установки системы для оптимизации ее работы.
2. Появилась возможность создавать корневую файловую систему на устройстве RAID.
3. Добавлена поддержка новых контроллеров RAID.

ОС ASPLinux 7.2, в отличие от других дистрибутивов, поддерживает все процессоры семейства i386: от i80386DX до Pentium IV.

Единственным неудобством является отсутствие драйвера для 3D-ускорителей nVidia (у меня Riva TNT2), но его можно свободно загрузить с сайта www.nvidia.com

Особого внимания заслуживает поддержка русского и украинского языков.

Для вас, как системного администратора, повышенный интерес должен представлять пакет **pptpd**, который позволяет организовывать сети VPN для клиентов Windows, а также пакет **portslave**, который в сочетании с улучшенной версией **pppd** позволяет организовать сервер удаленного доступа с авторизацией через сервер RADIUS и использованием функции «обратный звонок» **(callback).**

Заслуживает также внимания полностью переработанная документация: она находится на специальном компакт-диске с документацией (Documentation CD).

В завершении этого небольшого обзора отмечу довольно нестандартный дистрибутив **Astaro Security Linux.** С помощью этого дистрибутива вы можете превратить обыкновенный офисный компьютер в настоящий бастион. Все, что для этого нужно -- просто загрузиться с инсталляционного диска, ввести необходимые параметры сети и больше не подходить к этому компьютеру: можно сразу отключить монитор, потому что все администрирование сервера производится по протоколу https через бразуер. К сожалению в этом дистрибутиве не поддерживаются SCSI (впрочем, SCSI-диски на шлюзе — это излишество) и шины ISA, что не позволяет использовать старые сетевые платы.

1.10. Глоссарий

В этом небольшом пункте приведены описания основных терминов, которые использовались в первой главе, а также некоторых новых.

Аутентификация — проверка подлинности.

Интерфейс — средства и правила взаимодействия компонент системы между собой.

Коллизия — попытка одновременного доступа двух или более машин к среде передачи информации. Самый простой (и немного некорректный) пример коллизий — это параллельный телефон. В вашей жизни наверняка случалось, что кому-то нужно одновременно поговорить по телефону, и вы пытались набрать номер одновременно.

Концентратор (хаб) — устройство, которое просто передает полученные пакеты во все свои порты независимо от адресата. Все устройства, подключенные к концентратору Ethernet (включая другие концентраторы), «видят» весь сетевой трафик, но получить пакет должен только тот узел, которому он адресован. Все остальные узлы должны игнорировать этот пакет.

Маршрутизатор (router) — устройство для пересылки пакетов. Маршрутизатор собирает информацию о топологии межсетевых соединений и на основании этой информации пересылает пакеты сетевого уровня в сеть назначения. Маршрутизатор может быть программным и аппаратным.

Маршрутизация (routing) -- процесс передачи пакетов данных между двумя подсетями.

Мост (bridge) — устройство для соединения двух или более физических сетей. Мосты передают пакеты порту, к которому подключен адресат. Однако в отличие от большинства коммутаторов Ethernet, мосты не передают фрагменты пакетов при возникновении коллизий и пакеты с ошибками, поскольку все пакеты буферизуются перед их пересылкой в порт адресата. Мосты не зависят от протокола.

Порт — физическое или логическое устройство, через которое осуществляется процесс приема-передачи данных.

Протокол — совокупность правил, определяющая взаимодействие абонентов вычислительной системы и описывающая способ выполнения определенного класса функций.

Сервер — специальный компьютер, который предоставляет определенные услуги другим компьютерам.

Сокет (soket) — во многом аналогичен дескриптору файла (file handle). Сокет обеспечивает конечную точку соединения. Приложение, создавая сокет, указывает три параметра: IP-адрес узла, протокол (TCP или UDP) и порт, используемый приложением.

Узел — устройство, подключенное к сети.

Шлюз (Gateway) — межсетевой преобразователь. Выполняет функции, аналогичные мосту, но используется для связи сетей разных типов, например, LAN и WAN. Обычно для устройства, которое связывает две локальные сети, используется термин «маршрутизатор», а для устройства, которое соединяет вашу локальную сеть с Интернет, — шлюз, хотя оба эти устройства выполняют одну и ту же функцию — маршрутизацию пакетов.

2 Установка системы

2.1. Установка Red Hat Linux

Установку данного дистрибутива я буду рассматривать на примере, который применим к версиям, начиная с 6 (и, по крайней мере, до 8.x). Вообще не следует гнаться за новизной в версиях при создании сервера, так как, допустим, версия 6.0 (Hedwig) обладает достаточно низкими (по сегодняшним меркам) системными требованиями, что позволяет использовать устаревшую технику с максимальной отдачей. Например, старенький Intel 486DX4, на котором жутко «тормозят» Windows вместе с MS Office, можно с большим успехом использовать в качестве шлюза для выхода в Интернет (см. табл. 2.1). Это позволит вам сэкономить значительные денежные средства. Тем не менее, весь описанный далее механизм настройки сервера применим и к новым версиям (с некоторыми поправками, которые будут приведены в тексте). Более того, при использовании версий 6.0 и 8.0 вы не заметите почти никакой принципиальной разницы, так как конфигурирование всех основных служб остается одинаковым во всех версиях. Резюмируя, еще раз хочу сказать, что все примеры, приведенные в этой книге, должны работать в дистрибутиве любой версии, начиная с шестой. Однако может возникнуть аппаратная несовместимость, так как версия 6 по умолчанию не поддерживает шины AGP. При этом вам следует либо использовать видеокарту PCI, либо самостоятельно переустановить **XFree86** и сервер для вашей видеокарты (о том, что это такое, читайте далее). Версии 7.x, а тем более 8.x, этой проблемы лишены.

Примечание.

*Описание шестой версии операционной системы Linux Red Hat в книге приведено намеренно. Тому есть две веские причины. Во-первых, в состав шестой версии Red Hat входит суперсервер **inetd**, в то время как в состав более поздних версий (начиная с версии 7) входит суперсервер **xinetd**. А различия в настройке первого и второго довольно большие. Вы сами убедитесь в этом, прочитав гл. 8. Суперсервер **inetd** прекрасно работает на старых серверах, которые были настроены еще до вас. К тому же, **inetd** является предком **xientd**, поэтому для большего понимания функционирования **xientd** следует разобраться с **inetd**.*

Во-вторых, отличия в программе установки: программа установки более поздних версий Linux более дружелюбна. А вдруг вам придется столкнуться со старым дистрибутивом Linux или с установкой одного из варианта Unix: далеко не все варианты Unix имеет понятную программу установки.

Действительно, сравнивая системные требования Linux и продуктов от Microsoft, разницу ощущаешь сразу. Компьютер на базе процессора Pentium 133 и 32 мегабайтами ОЗУ на платформе Linux прекрасно справлялся с обязанностями Internet-шлюза, почтовика и прокси-сервера. Для того чтобы организовать те же функции на платформе Windows 2000 Server, вам необходим, по крайней мере, процессор Celeron 300 МГц и 256 Мб ОЗУ. Минимальные системные требования для установки Red Hat 6 приведены в табл. 2.1, а в табл. 2.2 указаны рекомендуемые системные требования для настройки сервера на ее основе.

Системные требования ОС Red Hat 6 *Таблица 2.1*

Показатель	Значение
Процессор	486DX
ОЗУ, Мб	8
Жесткий диск	Минимум 150 Мб

При использовании системы X Window (графической среды под Linux) вам понадобятся еще 8...16 Мб ОЗУ и 200...300 Мб дополнительного места на жестком диске. При установке сервера, как правило, система X Window не нужна, однако, чтобы удовлетворить всем читательским запросам, описание системы X Window приведено в гл. 20 данной книги.

Рекомендуемые системные требования для установки сервера *Таблица 2.2*

Показатель	Значение
Процессор	Pentium 133 МГц
ОЗУ, Мб	32
Жесткий диск, Мб	600

В зависимости от выполняемых задач вам может потребоваться дополнительное место на диске, например, для кэша прокси-сервера. Прокси-сервера используются для уменьшения времени загрузки Web-страниц (и не только). Конечно, это не единственное применение прокси-серверов, но об этом мы подробнее поговорим в гл. 15, а здесь немного рассмотрим как они работают. Войдем, так сказать, в курс дела.

Если пользователь подключен к Интернет через прокси-сервер и запрашивает какую-нибудь страницу, то запрос идет через прокси-сервер, который сперва ищет ее в своем кэше. Если запрашиваемая страница найдена, он передает ее пользователю, а если нет, то получает ее из Интернет, *кэширует* (сохраняет в кэше) и затем возвращает пользователю. Уже при повторном запросе данной страницы она будет загружаться из кэша прокси-сервера. При этом резко увеличивается скорость загрузки страницы, так как обычно прокси-сервер находится в одной подсети с пользователем, а передача данных в рамках одной сети осуществляется намного быстрее, даже если прокси-сервер и пользователь связаны каналом 33 Кбит/с. Обратите внимание, что кэшированию не подлежат часто обновляемые страницы, содержащие оперативную информацию, например, прогноз погоды.

Использование прокси-сервера оправдывает себя, если к нему подключается более трех-четырех пользователей. В противном случае хватит и локального кэша браузера пользователя. Таким образом, возвращаясь к основной теме данной главы, под кэшем прокси-сервера подразумевается определенная область на жестком диске, предназначенная для сохранения страниц. Обычно для небольшой локальной сети хватает и 300 Мб кэша, однако, если вы являетесь Интернет-провайдером, то вам может потребоваться кэш около 10 Гб. Настройка прокси-сервера описана в гл. 15.

Создание Linux-раздела на жестком диске

Теперь перейдем непосредственно к установке самой операционной системы. Так как Linux использует другой тип файловой системы по сравнению с другими ОС, то вы должны создать разделы для Linux. В самом простом случае вам потребуется два раздела: один — для самой операционной системы, другой — для свопинга (подкачки).

Примечание.

Существуют несколько определений файловой системы. Для себя вы можете запомнить одно из них — все они будут правильными.

** Файловая система — часть операционной системы, обеспечивающая выполнение операций над файлами.*

« Файловая система — способ организации и представления битов на жестком диске. Подробнее работа с файловой системой Linux рассмотрена в гл. 4.

Так как файловая система является частью операционной системы, естественно, у каждой операционной системы будет своя основная файловая система. Кроме основной файловой системы ОС может поддерживать несколько дополнительных. Например, основной файловой системой для Linux является ext2 (ext3), а дополнительными будут VFAT, ISO9660, UFS, XFS и другие.

Под процессом подкачки (swapping — свопинг) подразумевается перемещение страниц или сегментов виртуальной памяти или образов задач между оперативной памятью и внешней памятью, обеспечивающее нахождение используемой в данный момент информации в оперативной памяти. Запутал? Я так и думал. А теперь то же самое, но другими словами: если данные не умещаются в оперативной памяти, то они перемещаются на жесткий диск (внешняя память). При этом, если программе понадобилась какая-нибудь часть информации, находящаяся во внешней памяти, то операционная система подгрузит ее в оперативную память. Рассмотрим это дело на примере. Допустим, у вас в данный момент свободно всего 8 Мб оперативной памяти, а вы пытаетесь открыть документ размером в 16 Мб. В оперативную память при этом будут загружены первые 6...7 Мб, а все остальное будет находиться во внешней памяти. Когда вам потребуется перейти в конец документа операционная система *подкачает* в память нужные данные, а не используемые будут помещены во внешнюю память. Совокупность оперативной памяти и внешней памяти, используемой для подкачки (раздела или файла подкачки), называется **виртуальной памятью.**

Перед созданием разделов Linux обязательно выполните дефрагментацию файловой системы Windows, если такие разделы имеются. Причем,

если у вас несколько логических дисков, нужно дефрагментировать ВСЕ логические диски.

Существует несколько способов создания разделов для Linux:

» Классический способ — это использование **fdisk** для Linux. Использованию **fdisk** посвящен п. 4.7 данной книги.

» Вторым способом является использование встроенных возможностей программы установки.

» Третий — это использование программ посторонних разработчиков, например, Partition Magic.

Сейчас мы пойдем по пути наименьшего сопротивления: будем использовать второй способ создания раздела. Следует сразу оговориться: средство конфигурирования разделов программы установки Red Hat **(Disk Druid)** не умеет изменять размер раздела. При этом будет производиться конфигурирование всего указанного раздела под Linux, и если не будут размещены какие-либо данные, то они будут уничтожены. Поэтому, если вы хотите сохранить всю информацию, которая используется в вашем Windows-разделе, а свободное место отвести под Linux, вам нужно использовать программу **fips.** Ее можно найти в каталоге **dosutils** дистрибутива Red Hat. Перед ее использованием следует сначала дефрагментировать раздел Windows, размер которого вы хотите изменить, а еще лучше дефрагментировать все разделы.

После дефрагментации вам нужно перезагрузить компьютер в режиме эмуляции MS DOS и удалить файл подкачки Windows (`win368.swp`). Затем следует создать загрузочную дискету Windows — **(format a:** /s), скопировать на нее программу **fips** (используйте последнюю версию!), загрузиться с дискеты и ввести команду **fips.**

Интерфейс этой программы предельно прост — вам нужно только выбрать раздел, размер которого вы хотите уменьшить, и указать, сколько места на нем должно остаться после выполнения операции. После этого будет создан еще один раздел FAT или FAT32, который нужно будет удалить с помощью **Disk Druid** и создать вместо него Linux-раздел.

Четыре варианта установки Linux

Существует несколько вариантов установки Linux:

1. Используя загрузочный компакт-диск.
2. Используя загрузочную дискету (boot floppy).
3. Используя жесткий диск.
4. Установка по сети.

Самым удобным является первый способ — нужно просто загрузиться с компакта.

Если ваш BIOS не поддерживает загрузки с CD-ROM, то в этом случае нужно создать загрузочную дискету и загрузиться с нее. При этом, если не вдаваться в подробности, нужно сделать следующее:

* Скопируйте каталоги dosutils и images на жесткий диск (желательно на C:).
« Введите команду **rawrite** в сеансе MS DOS (а еще лучше перезагрузить компьютер в режиме MS DOS, так как программа **rawrite** может работать

некорректно из-под Windows. При работе в Windows можно использовать **rawritewin**):

```
c:\dosutils\rawrite
```

- На запрос программы

Enter disk image source file name:

введите:

```
c:\images\xxxx.img
```

где **xxxx**.....нужный вам образ:

boot.img.....обычная установка;

bootnet.img....установка по сети.

» А затем на предложение программы ввести диск назначения:

Enter destination drive:

введите **a:**

При установке с жесткого диска нужно скопировать каталог Red Hat на жесткий диск. Установку с жесткого диска целесообразно использовать, если скорость CD-ROM слишком мала.

Установка по сети производится по протоколу FTP или через NFS. При этом желательно, чтобы каталог Red Hat находился на сервере в общем (публичном) каталоге (см. рис. 2.1).

Итак, вы загрузились с компакт-диска или дискеты, выбрали язык операционной системы, раскладку клавиатуры. Затем программа установки попросит указать вас метод установки: CD-ROM или жесткий диск. При установке с жесткого диска еще потребуется указать путь к каталогу Red Hat. При установке по сети вам нужно ввести имя сервера, а также каталог на сервере, в котором расположен каталог Red Hat (обычно это /pub/Red Hat). Потом вам следует указать инсталляция это или обновление. Выберите Инсталляция.

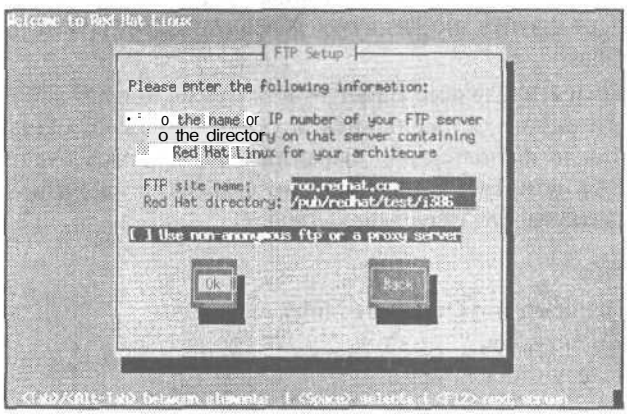


Рис. 2.1. Установка Red Hat по протоколу FTP

После этого вы приступите к самому интересному: выбору класса установки. При этом из предлагаемых вариантов вам следует выбрать класс По **выбору:**. При выборе Сервер (Server) или Рабочая станция (Workstation) содержимое диска уничтожится, а также вы не сможете выбрать пакеты вручную (см. рис. 2.2).

Для заинтересовавшихся привожу сведения о классах установки Server и Workstation.

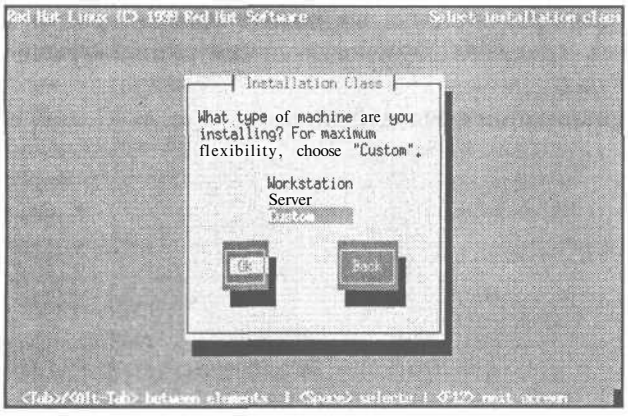


Рис. 2.2. *Выбор класса установки*

При выборе **Рабочей станции** (Workstation) будет удалена информация во всех разделах Linux, создан раздел подкачки размером 64 Мб, 16 Мб для каталога /boot - - в нем хранятся ядра системы, а также будет использовано все нераспределенное место на диске. Для этого класса установки вам необходимо как минимум 600 Мб на жестком диске.

При выборе **Сервера** (Server) будут удалены ВСЕ разделы, будет выделен раздел 64 Мб для раздела подкачки, 256 Мб для корневой файловой системы, минимум 512 Мб для каталога /usr, 512 Мб — для /home и 256 Мб — для /var. Для этого всего вам понадобится хотя бы 1.6 Гб на жестком диске.

Когда вы устанавливаете Linux, вы создаете корневую файловую систему. Под корневой файловой системой будем понимать раздел, на который вы установили Linux.

Приведенное определение корневой файловой системы является не совсем полным. В силу традиций определение корневой файловой системы (как и файловой системы вообще) является перегруженным. С одной стороны — это часть ядра, управляющая файлами и каталогами, а с другой —е набор утилит для загрузки, восстановления, починки и других операций с файловой системой.

Вот еще одно определение: корневая файловая система — это система, которая обладает инструментами для осуществления операций загрузки, восстановления и/или ремонта системы. При этом корневая файловая система должна быть по возможности небольшой, так как благодаря этому она будет меньше подвержена возникновению ошибок, будет легче в обслуживании, меньше затрат и потерь будет при ее восстановлении.

В данном контексте первое определение является более удачным, поскольку мы имеем дело с разделами (файловой структурой), а не с частью ядра. Для справедливости нужно отметить, что ядро операционной системы Linux имеет параметр root, который задает корневую файловую систему. Значениями данного параметра являются как раз разделы. Например, у вас может быть две основные файловые системы: два раздела (/dev/hda1 и /dev/hda2). В первый вы установили одну версию операционной системы, а во второй — другую (например, с X Window и без нее). При загрузке вы можете передать ядру параметр root, то есть указание, какую файловую систему нужно сделать *корневой.* Это делается так:

```
linux root=/dev/hdal
```

Впоследствии (после загрузки системы) вы можете примонтировать к корневой файловой системе (/dev/hda1) файловую систему, которая расположена на разделе /dev/hda2.

Рассмотрим пример организации файловой системы:

```
/ •- корень
|
|__/bin
|
|__/dev
|
|__/etc
|
|__/home
|
|__/mnt
|
|__/var
|
|__/root
|
|__/sbin
|
|__/tmp
```

Названия всех каталогов в данной системе являются стандартными. Их описание приведено в гл. 4 (п. 4.5), целиком посвященной файловой системе Linux.

Как я уже писал выше, в ОС Red Hat для создания разделов используется программа **Disk Druid,** которая является частью программы установки. Программа установки позволяет выбрать **fdisk** вместо **Disk Druid.** Какую программу использовать — это дело вкуса. Однако **Disk Druid** вам все равно придется использовать для указания точек монтирования (об этом читайте далее).

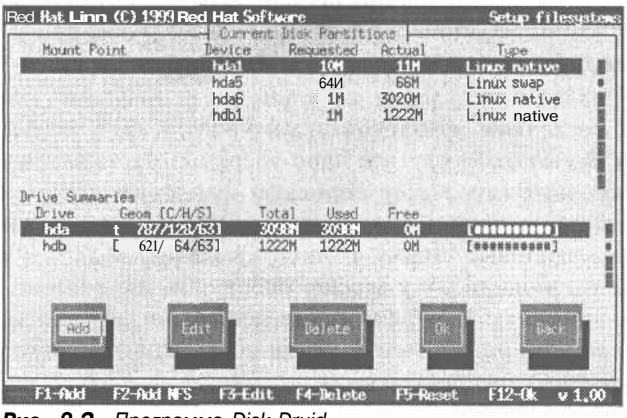


Рис. 2.3. *Программа Disk Druid*

При изменении размера Windows-раздела вы отвели какую-то его часть под раздел для Linux. Например, весь раздел составлял 2048 Мб, а для Linux вы выделили 600 Мб. Так вот, из этих 600 Мб вам потребуется примерно 64 Мб для организации раздела подкачки. Для этого создайте один раздел размером 536 Мб (тип **Linux Native),** второй — 64 Мб (тип **Linux**

Swap). Для первого установите точку монтирования «/», для второго точка монтирования не нужна. Далее вы можете установить точки монтирования для своих Windows-разделов (например, /mnt/disk_c) с помощью кнопки «Edit». Обратите внимание, что очень важно правильно определить размер раздела подкачки (см. табл. 2.3).

Определение размера раздела подкачки *Таблица 2.3*

Объем ОЗУ, Мб	Размер раздела подкачки, Мб
8...16	64
32	32
64...128	Равен объему ОЗУ
Более 128	Половина объема ОЗУ

Возможно, в процессе эксплуатации системы вам потребуется увеличить размер раздела подкачки. Однако на начальном этапе следует опираться именно на табл. 2.3. В дальнейшем вы в любой момент сможете создать файл подкачки.

По завершении всех этих действий система спросит у вас, какой раздел нужно использовать для подкачки (выберите только что созданный раздел **Linux Swap),** а также, какие разделы следует форматировать. В качестве последних выберите все, кроме Windows-разделов.

Для добавления нового раздела нажмите на кнопку «Add» (Добавить) и укажите размер, а также точку монтирования (см. рис. 2.3).

У вас наверняка есть разделы Windows. Чтобы работать с информацией, расположенной на этих разделах, вам нужно их *примонтировать*, то есть подключить к корневой файловой системе. **Точка монтирования** — это всего лишь каталог, через который будет происходить обращение к другим файловым системам. Обычно другие разделы подключаются к подкаталогам каталога /mnt. Например, если Windows-раздел примонтирован к подкаталогу win каталога /mnt, то просмотреть его содержимое можно командой **ls /mnt/win.**

С точки зрения безопасности каталоги /home и /var следует размещать на других разделах вашего жесткого диска, а еще лучше -- на другом жестком диске. В каталоге /home находятся файлы домашних каталогов пользователей. В случае краха корневой файловой системы (а такое иногда случается) файлы пользователей останутся неповрежденными. Для вас, как администратора, безопасность файлов пользователей является более критичным фактором, чем безопасность каких-либо других каталогов системы: ведь в случае чего систему можно довольно быстро восстановить, а вот файлы пользователей уже не вернешь.

Выбор пакетов для установки

Теперь нужно выбрать пакеты программ для установки (см. рис. 2.4). Если вы хотите выбрать пакеты самостоятельно, установите флажок «Индивидуальный выбор пакетов» и нажмите на кнопку «OK». При выборе пакетов будьте внимательны: некоторые пакеты для своей работы требуют наличия других пакетов — это называется зависимостью пакетов. Если это

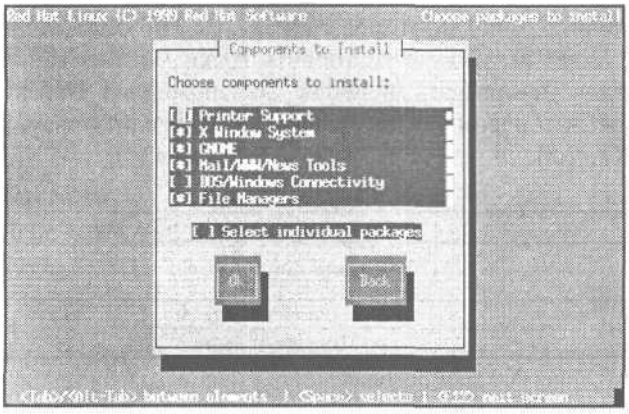


Рис. 2.4. *Выбор пакетов*

первая в вашей жизни установка Linux, не используйте возможность индивидуального выбора пакетов — просто выберите из списка категории программ, которые вас интересуют. После этого останется немного подождать, пока проинсталлируются выбранные вами пакеты. Время ожидания зависит от производительности вашего компьютера.

Настройки мыши

Следующий этап — настройка мыши. Здесь нужно выбрать вашу мышь и указать порт, к которому она подключена. При наличии двухкнопочной мыши включите режим эмуляции трехкнопочной — тогда одновременное нажатие сразу на обе кнопки будет восприниматься системой как нажатие на среднюю кнопку мыши.

Параметры локальной сети

После выбора мыши вам нужно указать параметры локальной сети (а не модемного соединения), если таковая имеется. Указание параметров начните с выбора сетевой платы из предлагаемого списка. Большинство сетевых плат ISA совместимо с платой NE2000, а PCI - с NEY2KPCI (NE2000 PCI). После выбора платы нужно указать параметры сетевого адаптера — IRQ, DMA, порт ввода/вывода. Для PCI плат эти параметры указывать не нужно!

Примечание.

При использовании установки по сети, тип и модель платы уже должны быть выбраны.

Установка времени и выбор демонов

Следующие два этапа — это установка времени и выбор демонов, которые будут автоматически запускаться при старте системы. Как быть с демонами? Оставьте пока все по умолчанию, позже, когда разберетесь, для чего предназначена каждая программа, отключите то, что вам не нужно.

Настройка принтера

Скорее всего, у вас локальный или сетевой принтер, подключенный к Windows-станции. На первом этапе выберите нужный вам тип принтера. После этого программа установки попросит ввести имя очереди и имя каталога для спула — оставьте все как есть. **Спул** — это область на диске, в которую помещаются печатаемые файлы непосредственно перед печатью. И при печати он считываются именно оттуда.

Следующий этап настройки принтера — выбор порта (см. рис. 2.5). Устройство `/dev/lp0` в Linux соответствует порту LPT1 в DOS, `/dev/lp1` — LPT2

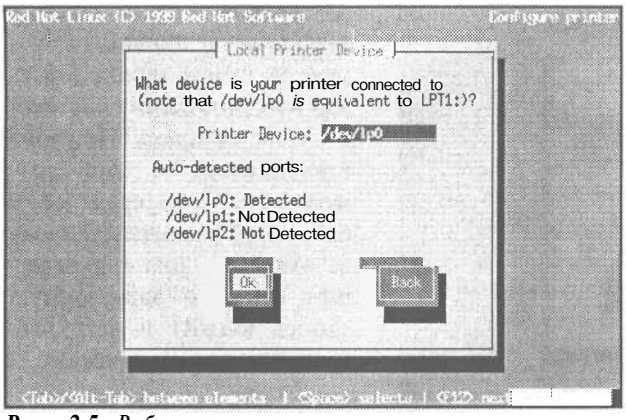


Рис. 2.5. Выбор порта принтера

и т.д. Выберите порт, к которому подключен ваш принтер и нажмите «OK». Если у вас сетевой SMB-принтер, то нужно будет еще ввести имя и IP-адрес сервера (машина под управлением Windows в вашей сети, к которой подключен сетевой принтер), рабочую группу, имя общего ресурса и, если нужно, имя пользователя и пароль.

Следующее, что необходимо сделать — это выбрать модель принтера и указать параметры печати (размер бумаги, разрешение печати). Очень советую включить режим «Исправлять ступенчатую печать» (Fix stair-stepping of text).

Если не включить режим «Исправлять ступенчатую печать», то при печати текстовых файлов большинство принтеров напечатает примерно следующее:

```
Первая строка
    Вторая строка
        Третья строка
```

Дело в том, что символ LF, используемый в Linux для обозначения конца строки, интерпретируется некоторыми принтерами как перевод строки без возврата каретки (символ CR). При использовании этого режима программа-фильтр, через которую проходит информация перед выводом на печать, добавляет после каждого символа LF символ CR.

Настройка безопасности. Задание паролей

Теперь самый ответственный с точки зрения безопасности системы этап — нужно ввести пароль для пользователя root. Пользователь root (суперпользователь) является главным пользователем системы — администратором. Пользователь root в Linux аналогичен пользователю Adminisrtator в Windows NT (2000/XP). Суперпользователь имеет право настраивать аппаратные средства, устанавливать и обновлять системное программное обеспечение и выполнять прочие операции, недоступные другим пользователям.

Задаваемый пароль должен быть не короче 6 символов. При вводе символы не будут отображаться на экране. Категорически не рекомендуется использовать в качестве пароля что-то вроде 123456, qwerty, password и тому подобное. Подумайте о выборе пароля — он должен быть одновременно легким для запоминания и трудным для подбора.

После ввода пароля для суперпользователя вам будет предложено установить параметры аутентификации (см. рис. 2.6). Включите использование теневых паролей и алгоритм MD5 («Use Shadow Password» и «Enable MD5 Password» соответственно). Изменить данные параметры позволяет программа **authconfig**.

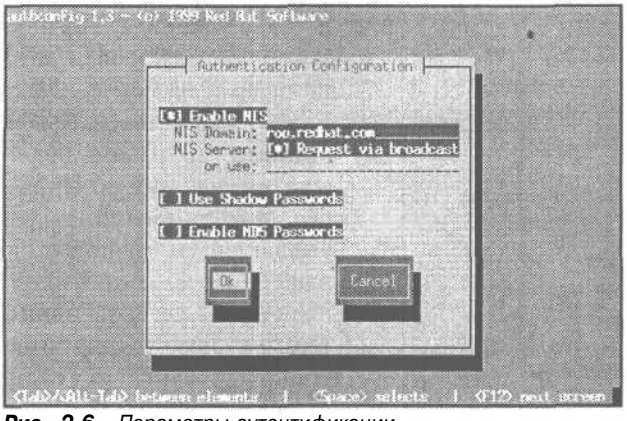


Рис. 2.6. *Параметры аутентификации*

Теперь нужно создать загрузочную дискету, которую вы будете использовать для восстановления системы в случае ее краха. Не стоит отказываться от этой возможности, т.к. если вы в очередной раз переустановите Windows, она перезапишет главную загрузочную запись (MBR) и загрузить Linux вы уже не сможете.

Установка LILO

LInuxLOader (LILO) — это загрузчик Linux и других операционных систем. Вам нужно решить, куда именно вы хотите установить загрузчик: в MBR или на первый сектор загрузочного раздела Linux. Если у вас установлена только Windows 9x, смело выбирайте MBR. В противном случае — выберите первый сектор загрузочного раздела и прочитайте раздел об установке нескольких операционных систем на одном компьютере.

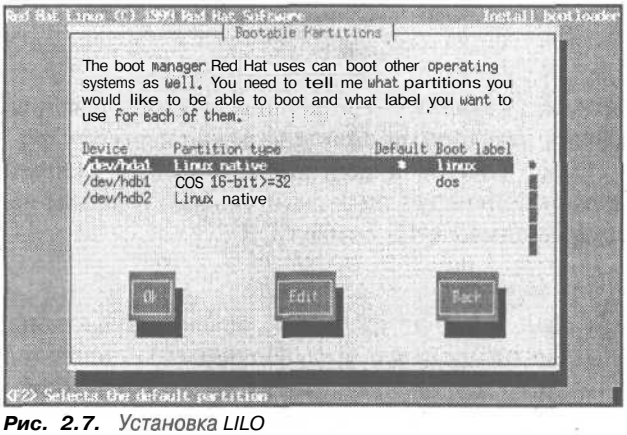


Рис. 2.7. *Установка LILO*

Имя загрузочного раздела для каждой операционной системы задается с помощью метки (label). Чтобы загрузить с помощью **LILO** определенную систему, вам нужно будет ввести соответствующую ей метку. При загрузке **LILO** выведет на экран приглашение:

```
LILO boot:
```

Чтобы загрузить Linux, вам нужно ввести **linux** или просто нажать «Enter», если раздел Linux у вас выбран по умолчанию (default). Чтобы загрузить DOS с раздела `/dev/hdb1` нужно ввести **dos.** Для просмотра всех доступных меток нажмите клавишу «Tab».

Конфигурирование XFree86 (X Window)

Система **X Window** является мощной графической средой для UNIX-станций. Данная система была разработана Массачусетским технологическим институтом (MIT) и стала стандартом для всех UNIX-систем. Практически каждая рабочая станция UNIX работает на одном из вариантов системы **X Window.**

Группа программистов, возглавляемая Дэвидом Вексельблатом (David Wexelblat) создала свободно распространяемую версию **MIT X Window** для процессоров i80386-Pentium IV и совместимых с ними. Эта версия получила название **XFree86,** поскольку могла выполняться в операционных системах, предназначенных для процессоров, использующих систему команд x86 (это Linux, FreeBSD и другие). **XFree86** является торговой маркой XFree86 Project, Inc. Подробнее о X Window читайте в гл. 20 этой книги.

Настройка видеоплаты

Итак, вернемся к процессу настройки аппаратных ресурсов компьютера. На данном этапе система сама попытается определить тип видеоплаты и размер видеопамяти. Модель монитора вы должны выбрать из списка. Если вашего монитора в списке не окажется, выберите *Custom* и введите параметры монитора вручную. Под параметрами я имею в виду разрешение, горизонтальную и вертикальную частоты. Нужные параметры можно найти в документации по монитору. Очень важно указать правильные параметры или хотя бы такие параметры, которые не превышают возможности монитора, иначе можно вывести из строя не только сам монитор, но и видеоплату. Если вы сомневаетесь, а документации на данный момент нет, выберите Standard VGA 640x480 — этот вариант работает всегда.

Нужно отметить, что Linux Red Hat 6 не поддерживает видеоплаты AGP. Если же у вас установлена именно она, то вам нужно использовать Red Hat 7.x (8.x) или самостоятельно установить более новую версию **XFree86** и сервер для вашей видеоплаты. В крайнем случае, замените видеоплату AGP на плату, предназначенную для шины PCI.

На этом все! Я могу вас поздравить — установка завершена.

2.2. Установка Linux Mandrake

В этом разделе книги будет рассмотрена установка дистрибутива Linux Mandrake. В табл. 2.4 указаны требования для рабочей станции. Для настройки сервера потребуется 128 Мб ОЗУ и жесткий диск объемом 10...20 Гб, хотя объем диска очень зависит от поставленных задач. Минимальная установка Linux Mandrake занимает 350 Мб. Если вы настраиваете шлюз для небольшой сети, вам хватит и 1 Гб с учетом кэша для прокси-сервера. При настройке рабочей станции нет смысла использовать жесткий диск объемом менее 2 Гб, так как вам потребуется место не только под систему, но и под программы и файлы пользователей.

Системные требования ОС Linux Mandrake *Таблица 2.4*

Устройство	Минимальные требования	Рекомендуемые требования
Процессор	Pentium	Pentium 166 и выше
ОЗУ, Мб	32	64
Жесткий диск, Мб	650	2048
Графическая плата	Совместимая с VESA 2.0	Совместимая с VESA 2.0

Примечание.

О том, что такое кэш прокси-сервера и сам прокси-сервер было написано в начале предыдущего раздела, посвященном установке Red Hat.

Программа установки Linux Mandrake выглядит немного покрасивее (см. рис. 2.8) по сравнению с Red Hat. Слева отображается ход установки. Красные звездочки (или кружочки в Mandrake 9) — это еще не пройденные этапы. Вы можете в любой момент переключиться на нужный вам этап установки или пропустить текущий, а потом вернуться к нему. Естественно, если такое возможно: например, без создания файловой системы вам никто не разрешит установить пакеты. Внизу экрана будут отображаться подсказки, которые помогут вам на всех этапах установки.

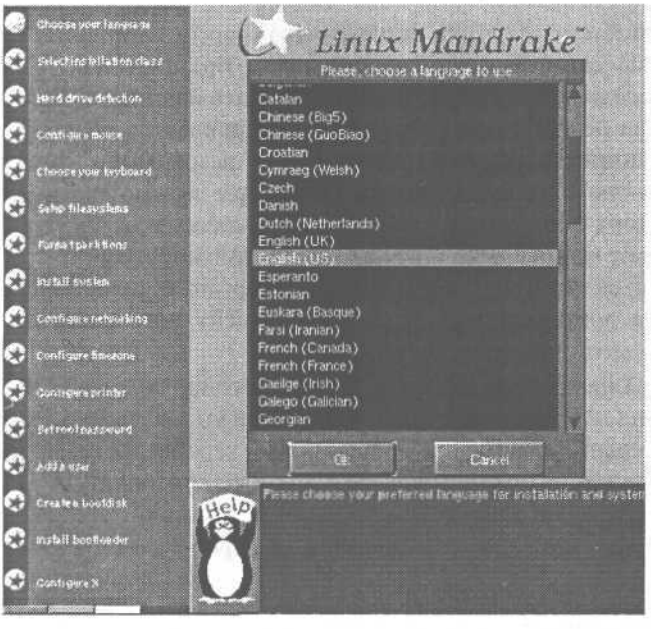


Рис. 2.8. *Программа установки Linux Mandrake*

При выборе типа установки, даже если вы начинающий пользователь, я рекомендую выбрать тип **Настроено** (см. табл. 2.5). Прежде всего, это обусловлено выбором нужных вам пакетов — зачем устанавливать лишнее? В Mandrake 9 список выбора сокращен до двух пунктов: Рекомендуется и Эксперт.

Выбор типа установки *Таблица 2.5*

Тип установки	Описание
Рекомендуется	Подходит для начинающих пользователей — программа установки сама создаст разделы, установит нужные пакеты и сконфигурирует аппаратные средства
Настроено	Вы можете сами создавать, форматировать разделы и выбирать группы пакетов
Эксперт	Настроено + индивидуальный выбор пакетов и возможность более тонкой настройки аппаратных средств. В Mandrake 7.2 вы также можете создать дискету для клонирования Linux

Для полноты описания я буду рассматривать тип **Эксперт.** При выборе этого типа вам будет предложено выбрать роль компьютера (см. табл. 2.6).

Роль компьютера *Таблица 2.6*

Роль компьютера	Описание
Рабочая станция (Workstation)	Самый лучший выбор для решения повседневных задач или для домашнего компьютера
Сервер (Server)	Будут установлены пакеты для решения серверных задач — http, ftp, mail и т.д. Графические рабочие столы и офисные приложения будут установлены в минимальном объеме
Разработчик (Development)	Оптимизация пакетов для разработки и отладки программного обеспечения

Выбираем **Сервер** и переходим к следующему шагу -- определению устройств SCSI. Если у вас есть SCSI-устройства, нажмите «Yes» и система попытается самостоятельно их определить.

Примечание.

SCSI (Small Computer System Interface) — системный интерфейс малых компьютеров — стандарт подключения быстродействующих подсистем и устройств. Данный интерфейс используется в компьютерах, в которых требуется обеспечить большую скорость обработки данных. Интерфейс SCSI более универсален, чем (E)IDE, потому что к нему можно подключить различные устройства, а не только магнитные накопители, например, сканер, использующий интерфейс SCSI.

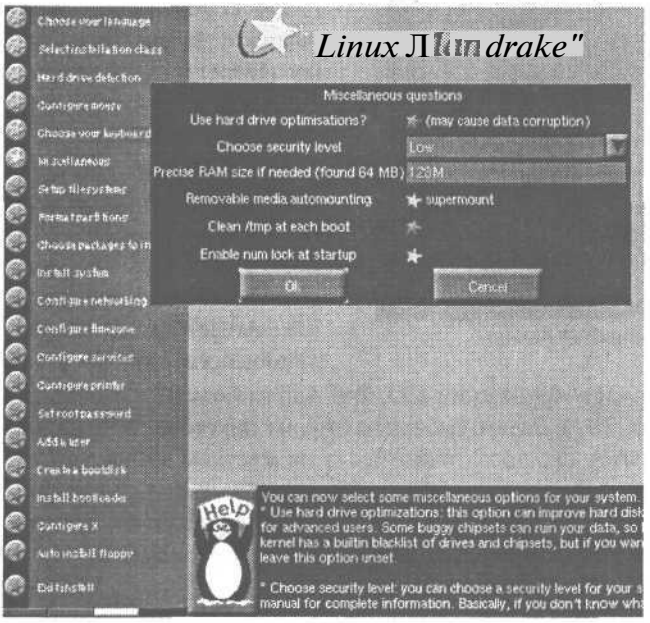


Рис. 2.9. *Раздел Miscellaneous*

Далее конфигурируем мышь и клавиатуру, а затем переходим к разделу Miscellaneous. Здесь вам предложат уточнить объем ОЗУ, выбрать уровень безопасности, а также произвести еще некоторые настройки (см. рис. 2.9). Чтобы слишком долго не объяснять поступим так: оптимизацию диска отключаем, уровень безопасности — низкий (можете установить средний), уточняем объем ОЗУ, используем **supermount** — очень удобная вещь (автоматическое монтирование съемных устройств — CDROM, Floppy), два последних режима включаем на свой вкус. Я бы включил очистку каталога /tmp при загрузке, а также **numlock.**

Для создания разделов Linux при инсталляции Linux Mandrake используется программа **DiskDrake.** Программа **DiskDrake** очень похожа на Partition Magic (см. рис. 2.10) и позволяет, в отличие от используемой в Red Hat программы Disk Druid, изменять размер существующих разделов.

Чтобы выделить место под Linux-раздел выберите любой Windows-раздел (предварительно дефрагментированный) и нажмите на кнопку «Resize». Установите новый размер раздела, например, на 2 Гб (2048 Мб) меньше. Я не рекомендую использовать раздел меньше 2 Мб для Mandrake — рано или поздно вам все равно потребуется место для других приложений, а также для ваших файлов.

Затем щелкните на «пустом месте» и нажмите на кнопку «Create». Выберите тип раздела **Linux native** и устаните его размер равным размеру свободного места минус размер раздела подкачки (см. табл. 2.7).

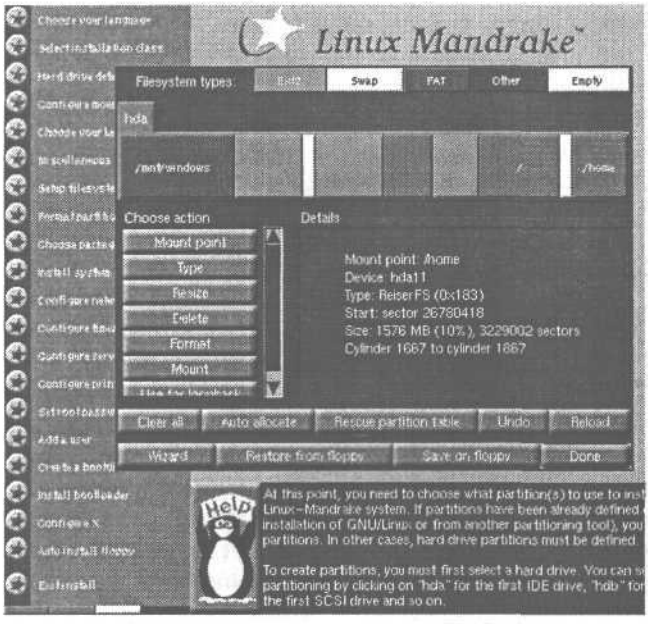


*Рис. **2.10.** Создание разделов с помощью DiskDrake*

Если у вас 256 Мб (или более) ОЗУ, вы можете установить размер раздела подкачки минимальным или вообще его не использовать, тем более, что при желании создать файл подкачки можно всегда. Аналогично создайте раздел для свопа (Linux swap). Далее нажмите на кнопку «Готово» (Done) и согласитесь на форматирование только что созданных разделов.

Как правило, дистрибутив Linux Mandrake (и многие другие) поставляется на нескольких компакт-дисках. Отметьте какие дополнительные (кроме инсталляционного) диски у вас есть, например, Extension CD, Applications CD, 2nd Applications CD, и нажмите на кнопку «OK» (см. рис. 2.11). Затем выберите группы пакетов, которые вам нужны (см. рис 2.12). Если у вас достаточно места на жестком диске, можете установить все пакеты, изучить их, а потом удалить ненужные вам программы. Если вы хотите самостоятельно выбрать пакеты (а не группы пакетов), включите режим «Индивидуальный выбор пакетов» (Individual package selection). Но на

Определение размера объема раздела подкачки *Таблица 2.7*

Объем ОЗУ, Мб	Размер раздела подкачки, Мб
32	128 и более
64	64 или 128
128	64
256	32

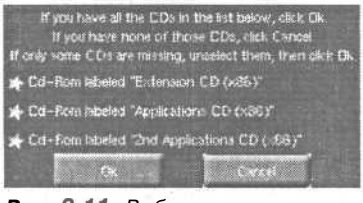


***Рис. 2.11.** Выбор компакт-дисков*

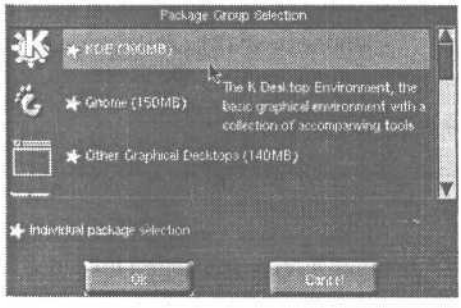


*Рис. **2.12.** Выбор группы пакетов*

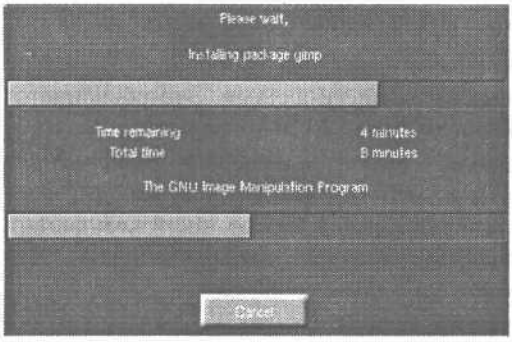


Рис. 2.13. *Установка пакета*

данном этапе я не рекомендую этого делать. После выбора пакетов нужно немного подождать, пока установятся выбранные вами пакеты (см. рис. 2.13).

В состав группы входят пакеты (файлы .rpm), которые необходимы для работы того или иного программного продукта. Существует понятие *Зависимость пакетов,* когда один пакет может требовать наличие другого (других) пакета (пакетов). Если хоть один из требуемых пакетов не установлен, установить этот пакет вы не сможете. Установка такого пакета возможна только при отключении режима проверки зависимостей, однако вряд ли пакет, установленный таким образом, будет работать корректно. Вот поэтому сейчас (пока вы начинающий пользователь, то есть администратор) просто выберите группы пакетов, без выбора составляющих их пакетов.

Следующий шаг — конфигурация сети. При настройке модема выберите порт и укажите параметры соединения. Все необходимые параметры соединения можно уточнить у вашего провайдера. Вы также можете сконфигурировать настройки локальной сети. Для этого вам нужно указать IP-адрес, маску сети, имя машины, адрес DNS-сервера и адрес шлюза. Все необходимые параметры можно узнать у администратора сети. Если вы сами являетесь администратором, то должны знать что здесь к чему...

Настройка принтера аналогична настройке принтера в Red Hat Linux. После ввода пароля суперпользователя (пользователь root) вам будет предложено добавить в систему обычных пользователей. Даже если вы будете использовать машину в гордом одиночестве, создайте одного пользователя для повседневной работы. Работать постоянно под root'oM не рекомендуется из соображений безопасности системы.

После этого вставьте в дисковод чистую отформатированную дискету и создайте на ней загрузочный диск. Что касается настройки LILO, то она аналогична настройке LILO в Red Hat Linux.

Следующий шаг — настройка монитора. Укажите производителя и модель вашего монитора. Если вашего монитора в списке не оказалось, можно попробовать High Frequency SVGA 1024x768 at 70 Hz. А если монитор старый, лучше Extended SVGA 800x600 at 60 Hz. Самый лучший совет в этом случае – читайте документацию по своему монитору! Установленные настройки необходимо протестировать. Во время тестирования экран может мерцать. После тестирования можно будет изменить параметры монитора или видеоплаты.

Вам также будет предложено создать дискету для клонирования Linux. Она вам очень понадобится, если вам нужно установить Linux на два или, большее число компьютеров, используя идентичные настройки и тот же выбор пакетов.

Поздравляю, установка завершена!

2.3. Установка Linux на компьютер с чипсетом Intel 810

Стоимость компьютеров, оснащенных чипсетом Intel 810, значительно ниже, чем компьютеров, оснащенных другими чипсетами. Это объясняется тем, что в материнскую плату с чипсетом Intel 810 интегрируются видео и/или звуковые платы. Такие материнские платы — идеальное решение для офисного компьютера. Чипсет I810 довольно быстро работает под управлением Windows 9x/ME/2000/XP, но при установке Linux на такой компьютер, скорее всего у вас возникнут проблемы с определением видеоплаты. Для решения этой проблемы нам понадобятся 2 файла:

`XFCom_i810-1.2-3.i386.rpm` - **X-сервер**
`I810Gtt-0.2-4.src.rpm` — модуль **agpgart.o**

Их можно скачать с сайта корпорации Intel, используя URL: **http://support.intel.com/support/graphics/intel810/agreeRPM3.htm и http://support.intel.com/support/graphics/intel810/agreeRPM4.htm** соответственно.

Все действия нужно производить от имени пользователя *roof.*

1. Установите **gcc** и **glibc** (если они у вас еще не установлены).
2. Не вдаваясь в технические подробности, отредактируйте ваш `/etc/lilo.conf`, добавив в него строку `append="mem=127M"` после строки `label="linux"`. Mem=127M — это количество доступной памяти, при условии, что всего установлено 128 Мб.
3. Введите команду **lilo**.
4. Установите первый пакет: **rpm -Uvh XFCom-i810-glibc2.1-1.0.0-rh60.i386.rpm**.
5. Откомпилируйте **agpgart.o: rpm —rebuild I810Gtt-0.l-4.src.rpm.**
6. Следите за сообщениями во время компиляции: система сама сообщит вам куда именно будет помещен пакет, приготовленный к установке. Обычно это `/usr/src/Mandrake/RPMS/i586/i810Gtt-0.1-4.i386.rpm` (`/usr/src/redhat/RPMS/i386/i810Gtt-0.1-4.i386.`rpm — для Red Hat).
7. Установите этот пакет: **rpm -Uvh 810Gtt-0.l-4.i386.rpm.**
8. Установите символическую ссылку: **ln** -sf **/usr/X11R6/bin/XFCom_i810** /etc/**X11/X**.

Теперь осталось отредактировать файл `/etc/X11/XF86Config` (см. листинг 2.1).

Листинг 2.1. Фрагмент файла /etc/X11/XF86Config

```
Section "Device"
 Identifier  "i810"
EndSection

Section "Screen"
 Driver "svga"
 Device "i810"
 Monitor ""

 Subsection "Display"
    Depth  16
```

```
    Modes "640x480" "800x600" "1024x768" "1152x864"
          "1280x1024" "1600x1200"
    Viewport 0 0
EndSubsection
Subsection "Display"
    Depth 24
    Modes "640x480" "800x600" "1024x768" "1152x864" "1280x1024"
    Viewport 0 0
EndSubsection
EndSection
```

Теперь нужно перезапустить **сервер X.** Для этого нажмите Ctrl+Alt+Backspace. Если **сервер X** у вас не запускается автоматически, введите команду **startx** для его запуска.

Примечание.

Систему X Window довольно часто называют еще другими именами: сервер X, клиент X (в зависимости от использования), а иногда и просто одной буквой X.

2.4. Установка нескольких операционных систем

Прежде чем устанавливать несколько операционных систем на одном компьютере, сделайте резервное копирование важной информации, т.к. процесс установки операционной системы включает в себя переразбиение жесткого диска на разделы и форматирование созданных разделов. Переразбиение жесткого диска необходимо, потому что Linux и Windows используют разные файловые системы. Хотя существуют средства установки Linux в раздел FAT/FAT32 — **Lin4Win,** но я не рекомендую их использовать, т.к. в этом случае Linux работает крайне нестабильно и медленно.

Для переразбиения диска я бы посоветовал программу Partition Magic v5 (или 6, или 7). Если вы устанавливаете Linux Mandrake 7.0 или выше, Partition Magic вам не потребуется — в программу инсталляции встроен отличный конфигуратор разделов на диске. В книге также будет описано использование программы **fdisk** для Linux, так как она присутствует во всех дистрибутивах Linux.

Рассмотрим два наиболее распространенных варианта установки нескольких операционных систем:

1. Вы устанавливаете Windows 9x и Linux.
2. Вы устанавливаете Windows NT, Windows 9x и Linux.

2.4.1. Установка Windows 9x и Linux

Главное правило при такой установке заключается в том, что сначала следует устанавливать Windows 9x, и только после этого Linux. Дело в том, что при установке Windows 9x перезаписывает главную загрузочную запись MBR (Master Boot Record) и, если Linux уже был установлен, загрузить его вы уже не сможете. При этом очень напрашивается следующий совет: СОЗДАЙТЕ ЗАГРУЗОЧНУЮ ДИСКЕТУ ДЛЯ LINUX! Если вы в очередной раз переустановите Windows, без этой дискеты загрузить Linux вы уже не сможете.

Что делать, если вы переустановили Windows и Linux больше не загружается:

1. У вас есть загрузочная дискета для Linux. Загрузитесь, используя эту дискету. Затем зарегистрируйтесь в системе как пользователь *root* и введите команду **lilo**. Затем перезагрузите машину (команда **reboot).**
2. У вас нет загрузочной дискеты: используйте программу **loadlin** — ее описание будет приведено ниже.
3. Нет загрузочного диска и нет программы **loadlin:** переустановите Linux, после установки Windows, естественно. При установке Linux не форматируйте разделы и тогда ваши данные останутся неповрежденными.

Самым оптимальным загрузчиком при данной схеме установки (Windows 9x + Linux) является **LILO** (Linux Loader). Я бы не рекомендовал использование каких-либо посторонних загрузчиков, как бы стабильно они ни работали. Во время инсталляции Linux программа установки спросит, куда устанавливать LILO — выберите MBR.

В случае деинсталляции Linux (после того, как вы уже удалили разделы Linux) восстановить MBR, то есть удалить LILO, поможет команда **fdisk /mbr**. При этом нужно использовать программу **fdisk** из комплекта загрузочного диска Windows.

2.4.2. Установка Windows 9x, Windows NT/2000 и Linux

Первый способ

В этом способе используется загрузчик NT **Loader.** Выполните установку Windows 2000 в раздел NTFS, а Windows 9x — в раздел FAT/FAT32. Не забудьте приготовить четыре системных дискеты для восстановления! Напомню, что если вы хотите установить Windows 95, то ее нужно устанавлиг вать первой, а потом Windows 2000. При установке Windows 98 и Windows 2000 порядок установки не имеет значения.

Устанавливать Linux нужно после установки Windows 2000. При этом необходимо учесть, что раздел Linux должен находиться до 1024 цилиндра! Это связано с «ошибкой цилиндра 1024» - - Linux может использовать разделы, расположенные после цилиндра 1024, но не может загружаться с таких разделов. В последних версиях Linux данная проблема устранена, но рассматриваемый способ установки требует, чтобы вы создали раздел Linux до цилиндра 1024 — иначе вам придется загружать Linux с дискеты.

Еще раз напоминаю: обязательно создайте загрузочную дискету для Linux. При установке LILO выберите MBR — Linux перезапишет главную загрузочную запись. Далее загрузите с четырех загрузочных дискет Windows 2000 и выберите пункт «Recover» в меню загрузчика и режим «Command mode». Затем зарегистрируйтесь в системе как *Administrator.* Выполните команды **fixboot и fixmbr** — теперь Windows 2000 будет нормально загружаться.

Примечание.

*Команды **fixboot** и **fixmbr** используются в Windows 2000 для восстановления главной загрузочной записи (MBR). Команда **fixmbr** практически аналогична команде **fdisk /mbr** в Windows 9x.*

Затем загрузитесь с системной дискеты Linux и войдите в систему под **root.** Откройте в любом текстовом редакторе файл `/etc/lilo.conf`, например, `joe/etc/lilo.conf`. В начале файла есть ссылка на загрузочный раздел по умолчанию, например, /dev/hda. Вам нужно изменить это значение на диск и раздел, в который была установлена ОС Linux, например, `/dev/hdb1`.

Введите команду **lilo** и увидите, что загрузочный раздел не является первым на диске — именно это вам и нужно. В этом случае загрузочная запись Windows 2000 не будет повреждена. Запишите загрузчик Linux в файл `/linux.ldr`:

```
dd if=/dev/hdcl bs=512 count=l of=/bootsect.lnx
```

Теперь этот файл нужно скопировать на дискету:

```
mcopy /bootsect.lnx a:
```

Перезагрузите Linux командой **reboot** и загрузите Windows 2000. Скопируйте файл linux.ldr в корневой каталог диска C: и присвойте ему атрибут *read-only*. Добавьте строку в файл boot.ini

```
C:\linux.ldr="Linux"
```

В результате, при перезагрузке компьютера вы сможете загрузить Linux с помощью NT **Loader.**

Второй способ

Существует более простой способ установки Linux и любой операционной системы семейства Microsoft Windows — Windows 9x, NT, 2000. Сначала нужно установить все ОС Windows, а потом установить Linux. При этом вместо начального загрузчика будет использоваться не NT **Loader, a LILO**. В этом случае вы получите двойное меню: сначала нужно выбрать между загрузкой Linux и Windows, а потом выбрать нужную вам ОС Windows — Windows 9x или NT/2000 - - в зависимости от того, какую систему вы установили, кроме Windows 9x. Второе меню — это как раз меню загрузчика NT **Loader.** В этом пункте описывался более сложный способ установки нескольких ОС — с помощью загрузчика NT **Loader.**

2.4.3. Использование loadlin

В этой книге я просто не мог не упомянуть о компактном загрузчике, который позволяет загрузить Linux из-под DOS или Windows 95. Если вы используете Windows 98, **loadlin** работать у вас не будет — нужно перезагрузиться в режиме MS DOS. Кроме этого следует учесть, что при использовании loadlin могут возникнуть проблемы с разделами FAT32. В этом случае создайте загрузочную дискету DOS (format a: `/s`) и в autoexec.bat этой дискеты пропишите loadlin. Загрузить Linux из-под операционных систем Windows 98, Me или Windows 2000/NT вам не удастся в любом случае.

Использовать **loadlin** я рекомендую лишь в том случае, если при очередной переустановке Windows оказался «затертым» MBR (а вместе с ним и LILO), а системную дискету Linux вы не создали. Вызов программы **loadlin** имеет следующий формат (описание параметров приведено в табл. 2.8):

```
loadlin ядроroot=корневая_ФС    опции
   или
loadlin @файл_с_параметрами
```

Параметры программы loadlin *Таблица 2.8*

Параметр	Описание
ядро	Ядро, которое вы используете. Если у вас на данный момент нет именно того ядра, которое было установлено, можно взять практически любое (естественно не самое древнее). Обычно ядро можно взять на компакт-диске с дистрибутивом Linux. Оно вам потребуется всего один раз — нужно только загрузиться и перезаписать lilo (команда lilo)
root= корневая_ФС	Корневая файловая система, например root=/dev/hda3
опции	Опции, которые будут переданы ядру во время загрузки. Обычно используется ro vga=normal
@файл_с_параметрами	Если параметры, которые вы передаете loadlin (а не ядру!) окажутся слишком длинными, то их можно записать в отдельный файл, а затем указать его имя в качестве параметра

Примеры:

```
c:\loadlin\loadlin.exe c:\loadlin\kernel\vmlinuz root=/dev/hda3
ro vga=normal
   или
c:\loadlin\loadlin.exe @param.lst
Param.lst:
c:\loadlin\kernel\vmlinuz root=/dev/hda3 ro vga=normal
```

Совет.

Можно включить загрузку Linux в стартовое меню DOS. Для этого отредактируйте свой config.sys следующим образом (см. листинг 2.2).

Листинг 2.2. Фрагмент файла config.sys

```
[MENU]
MENUITEM=DOS, Load DOS
MENUITEM=LINUX, Load Linux
MENUDEFAULT=DOS,5
[LINUX]
install=c:\loadlin\loadlin.exe c:\loadlin\kernel\vmlinuz root=/
dev/hda3 ro vga=normal
[DOS]
; Все остальные команды, которые вы используете в DOS,
; т.е. если у вас не было до этого стартового меню, то здесь
должен находиться
; весь ваш предыдущий config.sys
```

2.5. Постинсталляционная настройка

После установки системы вам, скорее всего, потребуется ее настроить. Практически всю настройку вы сможете выполнить при помощи программы **linuxconf**. На всякий случай в табл. 2.9 и табл. 2.10 перечислены другие программы-конфигураторы дистрибутивов Linux Mandrake и Linux Red Hat.

Основные программы-конфигураторы Linux Mandrake *Таблица 2,9*

Программа	Запуск в консоли (1)	Описание
Drakxconf	Да	Основной конфигуратор
Drakboot	Да	Конфигуратор загрузчика LILO
Drakgw	Да	Совместное использование Интернет-соединения
Draknet	Да	Настройка сети
Drakfloppy	Нет	Создание загрузочного диска
Draksec	Да	Определение уровня безопасности
Drakxservices	Да	Автозапуск сервисов
Diskdrake	Нет	Программа для работы с разделами диска
Adduserdrake	Да	Управление учетными записями
Keyboarddrake	Да	Настройка клавиатуры
Mousedrake	Да	Настройка мыши
Printerdrake	Да	Настройка принтера
Netconf	Да	Настройка сети
Modemconf	Да	Конфигурирование модема
XFdrake	Да	Настройка сервера X
Xdrakres	Да	Установка разрешения монитора
Xconfigurator	Да	Настройка X Window

Основные программы-конфигураторы Linux Red Hat *Таблица 2.10*

Программа	Описание
Setup	Основной конфигуратор
control-panel	Вспомогательный конфигуратор
Modemtool	Конфигурирование модема
Printertool	Настройка принтера
Netconf	Настройка сети
Xconfigurator	Настройка X Window
Authconfig	Параметры аутентификации

Существуют много других программ-конфигураторов, но основную настройку можно произвести, запустив **linuxconf** или основной конфигуратор — **setup** или **drakexconf.**

2.6. Установка программного обеспечения

Многие начинающие пользователи операционной системы Linux сталкиваются с проблемой установки нового программного обеспечения, а также удаления и обновления уже установленного. При описании установки той или иной программы в большинстве случаев написано примерно так: для установки

программы введите такую-то команду. При этом не описывается сам процесс установки и для чего предназначены те или иные опции программы установки.

В операционной системе Windows установка программ выполнялась проще: достаточно запустить setup.exe, ввести серийный номер (если нужно), каталог для установки и нажать на кнопку «Далее». После этого вы можете поступить так, как рекомендует Microsoft: «откиньтесь на спинку табуретки и подождите, пока программа установки все сделает за вас».

В операционной системе Linux существуют три способа установки программного обеспечения: традиционный, из пакетов RPM, из пакетов, содержащих исходный код. Рассмотрим по порядку все три способа.

2.6.1. Традиционный способ установки: установка из исходных текстов

Этот способ заключается в том, что программа распространяется не в собранном виде, а в виде исходных текстов. Данный способ называется традиционным, потому что он был первым способом установки программ до появления менеджера RPM или аналогичных ему **(apt-get).**

Как правило, исходный текст распространяется в архиве. Обычно файл, содержащий исходный текст, имеет двойное расширение: например, **tar.gz** или **tar.bz2**. **Это** означает, что данный файл сжат двумя архиваторами: сначала tar, а потом **gzip.**

Распаковывать архив нужно по принципу стека: сначала внешним архиватором, а потом внутренним. Предположим, что **prg-2.00.tar.gz** — это имя файла нашего архива. Для его распаковки нужно ввести команды:

```
gunzip prg-2.00.tar.gz
tar xvf prg-2.00.tar
```

Первая команда распакует файл **prg-2.00.tar,** который мы укажем в качестве одного из аргументов во второй команде. Параметр x программы **tar** означает, что нужно выполнить извлечение файлов из архива (параметр c — создание). Параметр v можете указывать по собственному усмотрению, он обеспечивает большую информативность при работе программы. Последний параметр f является обязательным при работе с файлами. Первоначально программа **tar** была предназначена для работы с пленками стримеров, поэтому нужно использовать параметр f, чтобы сказать программе, что нам нужно работать с файлами.

Если внешнее расширение не gz, а bz или **bz2,** то вместо первой команды вам нужно ввести команды (соответственно):

```
bunzip prg-2.00.tar.bz
bunzip2 prg-2.00.tar.bz2
```

Затем, как и в первом случае, нужно выполнить команду **tar** (с такими же параметрами).

Иногда файлы исходных текстов имеют всего одно расширение — tgz. В этом случае вам нужно ввести всего одну команду:

```
tar xzf prg-2.00.tgz
```

Параметр z означает извлечение файлов с использованием распаковщика gzunzip. Обычно такое расширение имеют файлы архивов, созданные с помощью программы tar и пропущенные через фильтр архиватора gzip.

Следующий этап — это непосредственная установка программы. После успешного завершения первого этапа (распаковки) перейдите в каталог, содержащий исходные тексты. Обычно это каталог <имя_программы-версия>:

```
cd prg-2.00
```

После этого вам нужно внимательно прочитать файл README и ввести три команды:

```
./configure
make
make install
```

Первая команда конфигурирует устанавливаемую программу для работы с вашей системой. Эта программа также проверяет, может ли устанавливаемая программа работать в вашей системе. Если работа программы невозможна, вы увидите соответствующее сообщение, и процесс установки будет прерван. Обычно такое случается, когда в вашей системе не установлена одна из необходимых новой программе библиотек. Для продолжения установки необходимо установить требуемую библиотеку и попытаться заново ввести команду **./configure**. После успешного завершения работы программы **./configure** будет создан файл **Makefile,** в котором будут указаны необходимые параметры (пути к библиотекам, путь для установки программы) для программы **make.**

Вторая команда **(make)** «собирает» программу. На этом этапе программа компилируется, то есть создаются бинарные исполнимые файлы из исходных текстов.

Третья команда — **make install** -- устанавливает программу и файлы справочной системы в соответствующие каталоги. Обычно программы устанавливаются в каталог **/usr/bin**, но это зависит от содержимого конфигурационного файла **Makefile.**

После успешной установки программы вы можете ее запустить, предварительно прочитав документацию по этой программе.

2.6.2. Программа RPM

Установка программного обеспечения в дистрибутивах Red Hat и Mandrake производится с помощью программы **rpm**. RPM (Red Hat Package Manager) — это менеджер пакетов Red Hat. Несмотря на то, что в названии присутствует «Red Hat», он полностью предназначен работать как открытая пакетная система, доступная для использования кем угодно. Она позволяет пользователям брать исходный код для нового программного обеспечения и упаковывать его в форме исходного и двоичного кода, так что двоичные файлы могут быть легко установлены и отслежены, а исходный код легко построен. Эта система также сопровождает базу данных всех пакетов и их файлов, что может быть использовано для проверки пакетов и запроса информации о файлах и/или пакетах.

В отличие от привычных мастеров InstallShield, которые используются для установки программ для Windows, пакеты **RPM** (файлы с расширением .rpm) не являются выполняемыми файлами, то есть программами. В пакетах содержатся файлы (как в архиве), которые нужно установить, а также различная информация об этом пакете: какой пакет необходим для работы этого пакета, с каким пакетом конфликтует, информация о разработчике, а также информация, указывающая, какие действия нужно выполнять при установке этого пакета, например, какие каталоги нужно создать. Менеджер пакетов RPM используется во многих дистрибутивах Linux (Red Hat, Mandrake, ASP, Black Cat.) и является довольно легкой и гибкой в использовании системой, что обуславливает его популярность.

Обычно в имени файла пакета указывается его название, версия, выпуск, платформа. Последние четыре символа — **«.rpm»** — признак того, что данный файл является пакетом. В Linux отсутствует такое понятие как расширение или тип файла.

Например, для пакета `software-1.0-1.i386.rpm` имеет место:

`software` ..... название,
`1.0` ..... версия программы,
`1` ..... выпуск пакета,
`i386` ..... платформа Intel 386.

Обратите внимание на разницу между версией программного обеспечения и выпуском пакета. Версия, указываемая в имени пакета, является версией программного обеспечения, находящегося в нем. Номер версии устанавливается автором программы, который, обычно, не является изготовителем пакета. Номер версии характеризует и относится к программному обеспечению. Что касается номера выпуска, то он характеризует сам пакет — указывает номер существующего варианта пакета. В некоторых случаях, даже если не изменилось программное обеспечение, бывает необходимо его переупаковать.

С названием и версией программы, я думаю, все ясно. А вот с архитектурой немного сложнее. Самыми «универсальными» пакетами являются пакеты, рассчитанные на архитектуру Intel 386. Данная программа должна работать на любом процессоре Intel, начиная с 80386DX (или совместимого с ним). А вот если у вас процессор 80486, пакет, рассчитанный для работы с архитектурой 80586 (Pentium), скорее всего, не установится в вашей системе. Обычно для процессоров архитектуры CISC (с набором команд x86) используются следующие обозначения:

i386 ..... Intel 80368DX;
i586 ..... Intel Pentium (MMX), AMD K5 (K6);
i686 ..... Intel PPro, Celeron, PII, PIII, PIV.

В самом простейшем случае команда установки пакета выглядит так:

```
rpm -i <пакет>.rpm
```

Перед установкой программы менеджер **RPM** проверит *зависимости пакета,* то есть установлены ли в вашей системе другие пакеты, которые необходимы новой программе или конфликтуют с ней. Если установлены

все нужные программе пакеты (или для работы программы вообще не нужны никакие дополнительные пакеты), а также, если новая программа не конфликтует ни с одним уже установленным пакетом, менеджер **RPM** установит программу. В противном случае вы получите сообщение, что для работы программы нужен какой-то дополнительный пакет или программа конфликтует с уже установленным пакетом. Если нужен дополнительный пакет, просто установите его. А вот, если программа конфликтует с уже установленным пакетом, то вам нужно будет выбрать, какой пакет больше нужен: уже установленный или новый.

При установке программы я рекомендую указывать два дополнительных параметра: h и v. Первый указывает программе вывести полоску состояния процесса установки, а второй выводит дополнительные сообщения. Полоска состояния будет отображена в виде символов #. Учитывая эти два параметра, команда установки немного усложнится:

```
rpm -ihv software-1.0-1.i386.rpm
```

Установку можно производить не только с локального диска, но и по протоколу FTP:

```
rpm -i ftp://somehost.domain/pub/package.rpm
```

Для удаления пакета используется команда:

```
rpm -e <пакет>
```

Еще раз следует напомнить, что при установке или удалении пакетов нужно иметь в виду, что одни пакеты могут требовать наличия в системе других пакетов — это называется *зависимостью пакетов.* Поэтому иногда вы не сможете установить определенный пакет до тех пор, пока не установите все пакеты, которые нужны для его работы. При удалении программы менеджер пакетов также проверяет зависимости между пакетами. Если удаляемый пакет нужен каким-нибудь другим пакетам, удалить его вы не сможете.

Для пропуска проверки зависимостей нужно использовать параметр — **nodeps.** Это бывает иногда полезно. Например, у вас установлена программа **postfix,** а вам нужно установить программу **sendmail**. Обе программы используются для отправки почты. Однако для работы многих почтовых программ необходим агент **MTA** (Mail Transfer Agent) - - программа для отправки почты **(postfix или sendmail).** Поэтому с помощью параметра -e удалить программу **postfix** вы не сможете. Установить программу **sendmail** без удаления программы **postfix** вы также не можете, потому что пакеты конфликтуют друг с другом. В этом случае вам поможет команда:

```
rpm -e –nodeps postfix
```

После такого удаления нормальная работа других программ, которым необходим MTA, невозможна, поэтому вам сразу же нужно установить программу **sendmail** (или другой MTA). Устанавливать программу в таком случае нужно как обычно: с помощью параметра -i.

Для обновления программ используется параметр -U. Я рекомендую использовать его и при установке программ, потому что, если устанавливаемый пакет уже был установлен, то будет произведено его обновление, а если нет, то будет просто установлен новый пакет. Для того чтобы видеть

полоску состояния при установке пакетов, используйте опцию h. Команда для обновления пакета:

```
rpm -Uhv <пакет>
```

например,

```
rpm -Uhv software-1.1-4.i386.rpm
```

Полоска состояния будет отображена в виде символов #. Просмотреть все установленные пакеты можно с помощью команды:

```
rpm -qa | less
```

Если вам требуется узнать установлен ли определенный пакет, выполните команду:

```
rpm -qa I grep название_пакета
```

Просмотреть общую информацию о пакете можно с помощью команды:

```
rpm -qi пакет
```

а информацию о файлах, которые входят в состав пакета:

```
rpm -ql пакет
```

2.6.3. Программы gnorpm, kpackage, apt

Менеджер пакетов RPM является мощным средством для произведения операций над пакетами — создание, установка, обновление, удаление. Однако интерфейс командной строки может понравиться далеко не всем, а особенно начинающему администратору. Существуют и графические (под X Window) реализации менеджера пакетов -- например, **kpackage** из KDE, **gnorpm** и другие. Я рекомендую использовать программу **gnorpm,** которая обладает интуитивным графическим интерфейсом. RPM больше подходит для создания новых пакетов, а также для обновления большого числа пакетов. Для установки одного-двух пакетов лучше и удобнее использовать **gnorpm** (см. рис. 2.14).

Функции программы **gnorpm:**

1. Установка пакетов.
2. Удаление пакетов.
3. Получение сведений о пакете.
4. Проверка пакета.
5. Поиск пакета в базе RPM.

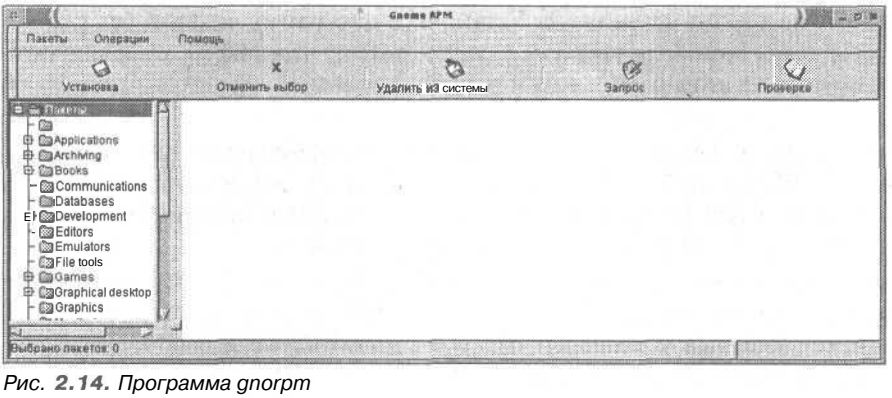


Рис. 2.14. Программа gnorpm

Для установки какого-либо пакета нажмите на кнопку «Установить». Если в приводе CD-ROM находится инсталляционный CD, то в появившемся окне вы увидите список еще не установленных в системе пакетов (см. рис. 2.15).

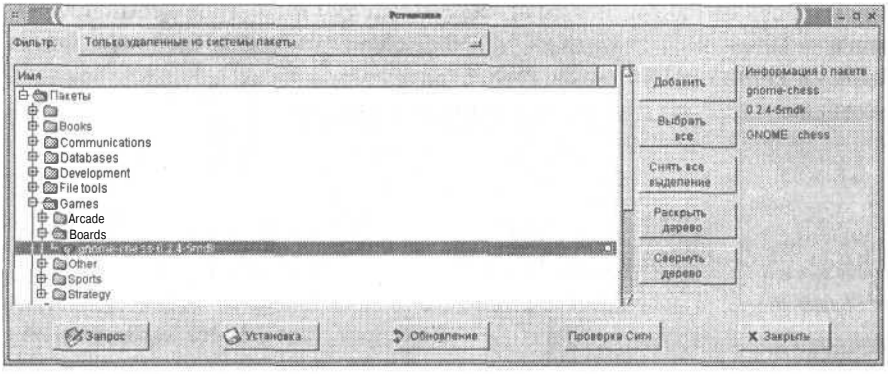


Рис. *2.15. Добавление пакета*

Если пакета нет в списке или вы хотите установить пакет, который не входит в состав дистрибутива, нажмите на кнопку «Добавить» и добавьте в список пакеты, которые вы хотите установить. Нажмите на кнопку «Запрос» для получения сведений о пакете (см. рис. 2.16).

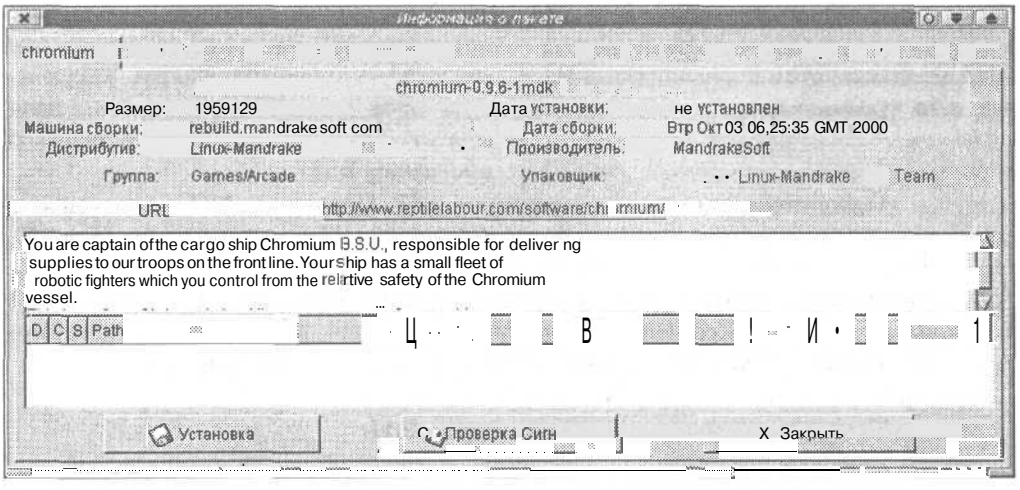


Рис. 2.16. Свойства пакета

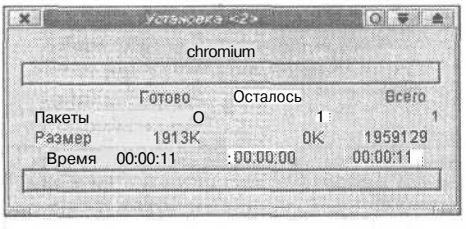


Рис. 2.17. Установка пакета

Если пакет еще не установлен и у вас достаточно места на диске для его установки, нажмите на кнопку «Установка». После этого будет выполнена проверка пакета на предмет удовлетворения зависимостей: не требует ли этот пакет наличия какого-нибудь не установленного пакета и не конфликтует ли он с уже установлен-

ными пакетами. Если все в порядке, вы увидите окно состояния установки пакета (см. рис. 2.17).

Найти пакет вы можете с помощью операции **Поиск.** Для этого нажмите на кнопку «Поиск» на панели инструментов gnorpm или выполните команду меню **Операции → Поиск.** В открывшемся окне вы можете установить критерии поиска и нажать на кнопку «Поиск» (см. рис. 2.18).

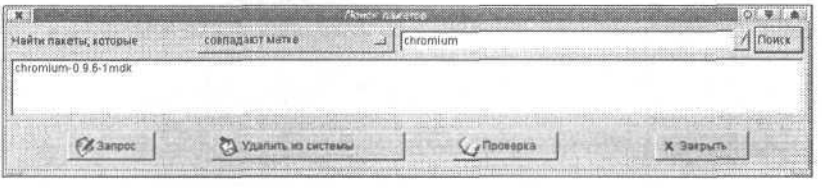


Рис. *2.18. Поиск пакета*

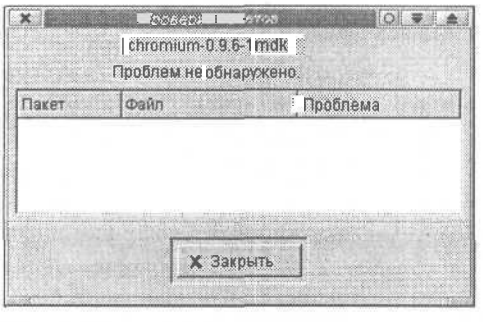


Рис. *2.19. Проверка пакета*

Для проверки пакета выберите интересующий вас пакет и нажмите на кнопку «Проверка» (см. рис. 2.19).

Процесс создания собственных пакетов RPM подробно рассмотрен в гл. 19.

В состав KDE входит программа с графическим интерфейсом пользователя, управляющая пакетами, — **kpackage** (см. рис. 2.20). По своим функциям она аналогична программе **gnorpm.** Какую из этих программы использовать — дело вкуса и привычки (я вообще использую программу rpm).

Также стоит упомянуть о программе **APT.** Программа APT — это система управления пакетами программного обеспечения. Первоначально система **APT** была разработана для **Debian** Linux. Сейчас в состав некоторых Red Hat-совместимых дистрибутивов (например, apt-get входит в состав Alt Linux, но ее вы не найдете в Red Hat Linux) входит эта система. Для управления пакетами используется программа **apt-get.** Формат вызова программы **apt-get** такой:

```
apt-get [опции] [команды] [пакет ...]
```

Команды программы описаны в табл. 2.11.

Команды программы apt *Таблица 2.11*

Команда	Описание
update	Используется для синхронизации файлов описаний пакетов с их источником, который указан в файле /etc/apt/sources.list. В качестве источника может использоваться какой-нибудь каталог файловой системы или FTP-архив. Примеры источников: /mnt/cdrom/RedHat/RPMS/ ftp://ftp.redhat.com/pub/
upgrade	Используется для обновления пакета. Может также использоваться для обновления ВСЕХ установленных в системе пакетов из источников, указанных в файле /etc/apt/sources.list. При выполнении этой команды ни в коем случае не будет установлено ни одного нового пакета, то есть такого, который еще не был установлен в системе, а будет произведено только обновление существующих пакетов. Перед этой командой обязательно должна быть выполнена команда update

Продолжение табл. 2.11

Команда	Описание
dist-upgrade	Более «интеллектуальная» версия команды upgrade. Кроме установки новых версий пакетов, она также проверит зависимости между новыми версиями пакетов. Рекомендуется использовать именно эту команду
install	Установка одного (или более) пакета. В качестве аргумента данной команде нужно передать имя пакета: это НЕ полное имя файла. Например, пусть полное имя файла пакета, который вы хотите установить /mnt/cdrom/RedHat/RPMS/package-1.33.i386.rpm. Тогда для установки этого пакета вам нужно ввести команду apt-get install package. Естественно, источник /mnt/cdrom/RedHat/RPMS должен быть указан в файле /etc/apt/sources.list
remove	Удаление пакетов
check	Используется для диагностики нарушенных зависимостей между пакетами
clean	Очищает локальное хранилище полученных файлов пакетов. Перед установкой пакеты копируются из источника в локальное хранилище, а оттуда потом устанавливаются. Для освобождения места на диске время от времени вводите команду apt-get clean. Данная команда не удаляет пакеты из каталогов /var/cache/apt/archives и /var/cache/apt/archives/partial

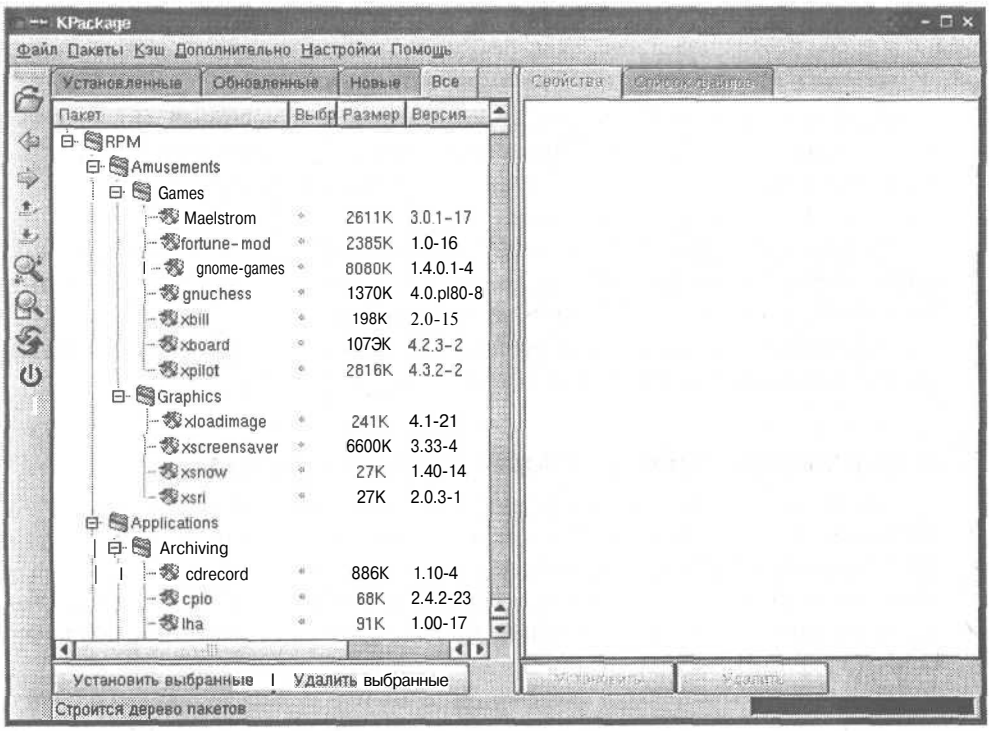


Рис. 2.20. *Программа kpackage*

В табл. 2.10 указаны практически все необходимые вам команды. Более подробное описание команд вы найдете в справочной системе.

Команда **apt-get install** похожа на команду **rpm** -i, однако есть одно важное отличие: при установке нового пакета **apt-get** проверяет зависимости и устанавливает также все необходимые пакеты. Менеджер **rpm** выводит только сообщение о невозможности установки пакета.

При установке группы пакетов с помощью **apt-get** будьте внимательны. Обычно для установки группы пакетов используются символы «?», «*». Если нет пакетов, имена которых совпадают с указанным шаблоном, то этот шаблон будет рассматриваться как выражение **POSIX.** В этом случае, если вы указали шаблон a*, то будут установлены ВСЕ пакеты, имена которых содержат букву а, а не только те, которые начинаются на эту букву. Это же касается и команды **remove.**

Из опций **apt-get** полезными для вас будут -f и -d. При указании первой опции **apt-get** попытается исправить нарушенные зависимости, а при указании второй -- пакеты не будет устанавливать, а только будут выкачаны из источника. Еще есть одна полезная опция: **—force-yes**. При указании этой опции **apt-get** добровольно-принудительно выполнит указанную операцию несмотря ни на что. Данная опция очень опасна, так как может полностью разрушить систему, однако бывают случаи, когда она действительно необходима.

2.6.4. Установка из пакетов, содержащих исходный код

Иногда в пакетах RPM находятся не откомпилированные версии программ, а их исходный код. Признаком этого является слово **src** вместо названия архитектуры. Для установки такого пакета введите:

```
rpm —rebuild software-2 . 00-1 . src. rpm
```

Разумеется, вместо software-2.00-1.src.rpm нужно указать реальное имя файла. Перед установкой программы пакет, ее исходный текст будет откомпилирован, и потом программа будет установлена.

Надеюсь, что всей этой информации достаточно, чтобы вы могли легко устанавливать программное обеспечение для Linux.

2.7. Завершение работы операционной системы

Очень важно правильно завершить работу операционной системы. Не забывайте, что нельзя просто выключить питание или нажать «Reset». Неправильное завершение работы операционной системы может вызвать потерю данных на диске или, в худшем случае, разрушить всю файловую систему. Это относится не только к Linux, но и ко всем многозадачным операционным системам.

При завершении работы системы (останов или перезагрузка) выполняется размонтирование файловых систем, в том числе и корневой файловой системы. При размонтировании файловой системы происходит синхронизация буферов дискового ввода-вывода с самим жестким диском, поэтому, если нажать на «Reset», то велика вероятность того, что программы не успеют записать данные на диск. Раньше (еще во времена UNIX — до появления Linux) системные администраторы перед завершением работы использовали команду **sync,** которая и выполняла эту синхронизацию.

Специально для корректного завершения работы ОС во всех дистрибутивах используется команда **shutdown.** Эту программу можно вызывать со

следующими параметрами: тип остановки, время остановки и сообщение. Тип означает или полную остановку, или перезагрузку системы, а время — когда программа **shutdown** должна остановить систему. Указанное сообщение будет отображено на всех терминалах, чтобы пользователи могли красиво завершить работу. Например, выключить систему в 19:00 можно командой:

```
shutdown -h 19:00 The end of a working day
```

При этом за несколько минут до завершения работы на всех терминалах будет отображено предупреждающее сообщение: «The end of a working day». Для немедленной остановки системы используйте команду:

```
shutdown -h now
```

Параметр -h указывает на то, что должна быть произведена полная остановка системы, a now — это время, в которое ее необходимо произвести. Для перезагрузки вместо параметра -h используется параметр **-r**. Время задается либо в формате ЧЧ:ММ, либо в формате +N, где N — количество минут, которое будет отсчитываться, начиная с текущего момента. Значение now есть ни что иное, как псевдоним +0 минут.

Для завершения работы используются также команды **halt и reboot** — для останова и перезагрузки системы соответственно. Команда **halt** является не чем иным, как символической ссылкой на команду *«shutdown -h now»,* а **reboot** — на *«shutdown -r now»*.

Во время разгрузки системы завершаются все процессы, выполняется синхронизация дисков (sync) и демонтируются файловые системы. Не выключайте питание, пока не увидите сообщение:

```
The system is halted
```

При нажатии Ctrl+Alt+Del обычно выполняется команда **shutdown -r now.** Хотя, в общем случае, реакция системы на нажатие «комбинации из трех пальцев» может быть установлена в файле `/etc/inittab`.

3

Учетные записи пользователей

3.1. Вход в систему

Linux — это многозадачная и многопользовательская система. А это значит, что в системе могут одновременно работать несколько пользователей, которые будут использовать многозадачный интерфейс (запускать несколько программ одновременно). И это в отличие от Windows, где хоть и можно создать различные учетные записи пользователей, но в одно и то же время с системой может работать только один пользователь, который, правда, может использовать многозадачные возможности операционной системы.

Идентификация в Linux состоит из двух шагов: ввода имени пользователя (логина) и пароля, который никто кроме вас не знает (во всяком случае, не должен знать...). При входе в систему вы увидите примерно такую подсказку:

```
localhost login:
Password:
```

На что вы должны ввести свой логин и пароль. При некорректном вводе вы увидите сообщение *Login incorrect* и вам придется ввести пароль еще раз. При начальной регистрации администратор сам назначает вам пароль, который вы потом можете изменить. Так как читатель сам является администратором, то написанная выше строчка его не касается.

В Linux существуют виртуальные консоли. **Консоль** — это дисплей и клавиатура, связанные воедино. Виртуальные консоли позволяют войти в систему под одним и тем же именем несколько раз одновременно. Для демонстрации этого момента нажмите Alt + F2 — вы перейдете на вторую виртуальную консоль. Если до этого вы работали в X Window, нажмите Ctrl + Alt + F2 (см. табл. 3.1).

Некоторые комбинации клавиш при работе в консоли и X Window — *Таблица 3.1*

Комбинация клавиш	Описание
All + Fn	Переключение между ВК
Ctrl + Alt + Fn	Переключение из X Windows ВК с номером п
Alt + F7	Возврат в X Window
Ctrl + Alt + Backspace	Аварийный выход из X Window

Для выхода пользователя из системы, то есть для завершения текущего сеанса работы, используется команда **exit.** В некоторых старых дистрибутивах для этого использовалась команда **logout.**

3.1.1. Вход в систему под другим именем

Команда **login** используется для входа в Linux-систему. Для входа под другим именем нужно ввести **login [имя] [параметры].** Если имя не указано, программа запросит его. Команда login позволяет использовать параметры, указанные в табл. 3.2. Для регистрации в качестве суперпользователя (root) нужно использовать команду **su.**

Параметры команды login *Таблица 3.2*

Параметр	Описание
-f	Пропускает вторичную аутентификацию. Этот параметр сохранился со времен UNIX и в Linux работает не так как надо
-P	Сохраняет переменные окружения, используемые getty. Программа getty — это программа, устанавливающая связь между операционной системой и указанным терминалом
-h имя_хоста	Передает команде login имя удаленного хоста. Обычно используется на серверах

3.2. Изменение пароля

Чтобы изменить свой пароль воспользуйтесь командой **passwd.** Ее нужно ввести без параметров. При этом команда запросит новый пароль. Если вы не *root,* система не разрешит вам ввести пароль, являющимся словом, или короткий пароль. В другом случае вы можете ввести все, что угодно — даже *123* тоже будет допустимо — система не сможет вам помешать. Обычно, если пользователь изменяет свой пароль, система проверяет пароль на возможность подбора, и не является ли этот пароль словом из системного словаря. Также проверяется длина пароля. Минимальная допустимая длина пароля – шесть символов, в некоторых дистрибутивах — восемь.

Чтобы изменить пароль какого-нибудь пользователя, вы должны (зарегистрировавшись как *root)* в качестве параметра команды **passwd** указать имя этого пользователя.

3.3. Идентификаторы пользователя и группы

Система Linux чем-то похожа на монархическое государство: в нем существует один суперпользователь — root, которому все подчиняется, и определенное число обыкновенных пользователей. Это значит, что если вы попробуете удалить один из жизненно важных файлов Linux под учетной записью пользователя, то система не позволит вам этого сделать. Под учетной записью root вы имеете право делать все, что вам заблагорассудится. Поэтому в целях безопасности, даже если вы используете систему в гордом одиночестве, рекомендуется создавать в системе хотя бы одного пользователя, под которым вы будете работать. Если вы работаете с системой на правах пользователя, а вам вдруг потребовались права суперпользователя, то

в этой ситуации вы можете воспользоваться командой **su**, которая в большинстве случаев вам поможет. Эта команда разрешит вам использовать привилегии суперпользователя без завершения текущего сеанса работы.

Для создания учетной записи используется команда **adduser <имя_пользователя>**. Естественно, добавлять новых пользователей может только root. После создания новой учетной записи не забудьте изменить пароль пользователя командой **passwd < имя_пользователя>**.

При добавлении новых пользователей используются параметры по умолчанию. Естественно, они вас не будут устраивать (например, только потому, что для каждого нового пользователя создается новая группа с именем пользователя). Изменить эти параметры можно с помощью опций программы **passwd.** Более подробно об этих опциях вы можете прочитать в справочной системе **(man passwd).** Для автоматизации процесса создания учетных записей я предлагаю использовать небольшой сценарий, который вы найдете в конце этой главы.

Примечание.

Обычно пользователи объединяются в группы для решения какой-нибудь задачи, например, для работы над одним проектом или же вы просто хотите создать различные группы пользователей для разных отделов предприятия, чтобы потом определенным образом установить права доступа к какому-нибудь объекту. Если каждый пользователь будет находиться в своей группе, то такое управление доступом будет невозможным.

В общем случае система хранит следующую информацию о пользователе:

Имя пользователя (**username**) -- регистрационное имя пользователя, то есть логин. Желательно, хотя и не обязательно, создавать пароли, каким-то образом ассоциирующиеся с определенными пользователями (с их реальными именами). Это упрощает работу администратора, позволяя ему быстро по паролю распознавать пользователя.

Идентификатор пользователя (User ID) — индивидуальный числовой идентификатор пользователя (UID). Система обычно работает с UID, а не с именами пользователей. Идентификатор задается из диапазона 0...65534 и должен быть уникальным. Число 0 соответствует пользователю root. Желательно идентификаторы назначать не произвольным образом, а системно. Например, выделить определенный интервал (1000...1100) под одну группу пользователей, а еще один (2000...2100) — под другую группу. В каждом диапазоне назначать идентификаторы последовательно. Это опять же упростит администрирование и позволит, бегло взглянув на список процессов, сразу-же определить кто чем занимается.

Идентификатор группы **(Group ID)** — числовой идентификатор первичной группы пользователя (GID). Помимо первичной группы пользователь может входить или не входить в состав разных групп, но в первичную группу (native group) он входит всегда. В различных дистрибутивах это выглядит поразному. Идентификатор группы 0 соответствует группе root.

Пароль **(password)** — пароль.

Реальное имя пользователя (**full name**) — обычно представляет собой реальное (фактическое) имя пользователя, например, Ivan Ivanov. Может содержать и

другие данные: номер телефона и т.п. Эти сведения используются в информационных целях.

Домашний каталог пользователя **(home dir) — в** качестве домашнего каталога обычно используется каталог /home/<имя_пользователя> (например, /home/den). Без особых причин не рекомендуется изменять такую организацию домашних каталогов.

Оболочка пользователя **(login shell)** — командный интерпретатор пользователя, который используется им по умолчанию. Программа-оболочка (командный интерпретатор) запускается при входе пользователя в систему. Примеры командных интерпретаторов: ash, bash, csh, fcsh, ksh.

Вся эта информация хранится в файле /etc/passwd **в следующем виде:**

```
username:password:UID:GID:full_name:home_dir:login_shell
```

Пример фрагмента файла /etc/passwd:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
den:x:500:500:den:/home/den:/bin/bash
evg:x:501:501::/home/evg:/bin/bash
```

На практике обычно используются теневые пароли, и вместо пароля в файле /etc/passwd стоит *, а сам пароль хранится в файле /etc/shadow, естественно в зашифрованном виде. Применение теневых паролей оправдывает себя с точки зрения безопасности. Обычно к файлу /etc/passwd разрешен доступ в режиме «только чтение» всем пользователям. К файлу /etc/shadow обычный пользователь не имеет даже такого доступа.

В качестве основного алгоритма шифрования используется MD5. Этот алгоритм является самым надежным. Раньше использовались алгоритмы DES и 3DES, но здесь я не буду подробно останавливаться ни на одном из них. При установке системы обычно спрашивается, хотите ли вы использовать теневые пароли **(Shadow Passwords)** и **MD5.** Я очень рекомендую вам использовать обе эти возможности.

Домашние каталоги пользователей обычно располагаются в каталоге /home основной файловой системы, но вы можете назначить и любой каталог в качестве домашнего. Суперпользователь использует домашний каталог /root. Для перехода в домашний каталог используется команда cd ~.

Примечание.

В Linux текущий каталог обозначается точкой, родительский каталог — двумя точками, а домашний каталог пользователя — символом ~ (тильда).

3.4. Создание группы

Каждый пользователь принадлежит к одной или более группам. Группы используются для того, чтобы пользователи, принадлежащие одной группе, могли работать с общими файлами. Например, в группу user входят пользователи ivanov и petrov, а пользователь sidorov входит в группу sgroup. Пользователь ivanov создает файл report и разрешает всем членам группы users

работать с этим файлом. При попытке обратиться к файлу report пользователь sidorov получит сообщение **Permission denied.**

Как правило, все члены группы имеют доступ к домашним каталогам друг друга. По умолчанию разрешается только чтение для членов группы. Более подробно о правах доступа к файлам мы поговорим в гл. 4. Информация о группах пользователей хранится в файле /etc/group. Формат этого файла следующий:

```
имя_группы:пароль:GID:члены_группы
```

Пароль используется крайне редко. Пример файла /etc/group представлен в листинге 3.1.

Листинг 3.1. Пример файла /etc/group

```
root:*:0:
local:*:100:den,operator,ivan
guest:*:200:
dialup:*:250:victor,evg
```

В этом примере группа root зарезервирована для пользователя *root*. Группа с идентификатором 100 используется для локальных пользователей. В ее состав входят пользователи den, operator, ivan. Группа quest предназначена для гостевого входа и пользователя guest. В состав группы dialup входят пользователи victor и evg.

Существует несколько групп, определяемых самой системой, вроде **bin, mail** и sys. Пользователи не могут принадлежать к какой-либо из этих групп. Эти группы используются для системных файлов. Добавить группу вы можете с помощью команд **groupadd**. **Я,** как правило, просто добавляю запись в файл /etc/group, а если мне нужно удалить группу, то удаляю соответствующую строку.

3.5. Удаление и модификация учетных записей

Для удаления пользователя можно воспользоваться командой **userdel.** Удалять учетные записи умеет и **linuxconf**. При удалении пользователя программа **linuxconf** спросит у вас, что делать с домашним каталогом удаляемого пользователя: удалить, архивировать или оставить без изменения. Что делать с ним — это уже вам решать.

Модифицирование учетной записи реализуется программой **usermod**. Здесь хочу дать один совет: для модифицирования (да и создания) учетных записей гораздо удобнее пользоваться программой **linuxconf.** Например, для создания учетной записи, введите команду **linuxconf** (или **userconf** — для Linux Mandrake). Затем запустите конфигуратор учетных записей, нажав на кнопку «User Accounts» (если вы запустили **userconf,** этого делать не нужно). В окне **User Account Configurator** перейдите на вкладку **Normal** (см. рис. 3.1) и нажмите на кнопку «User Accounts» (см. рис. 3.2), а затем на кнопку «Add».

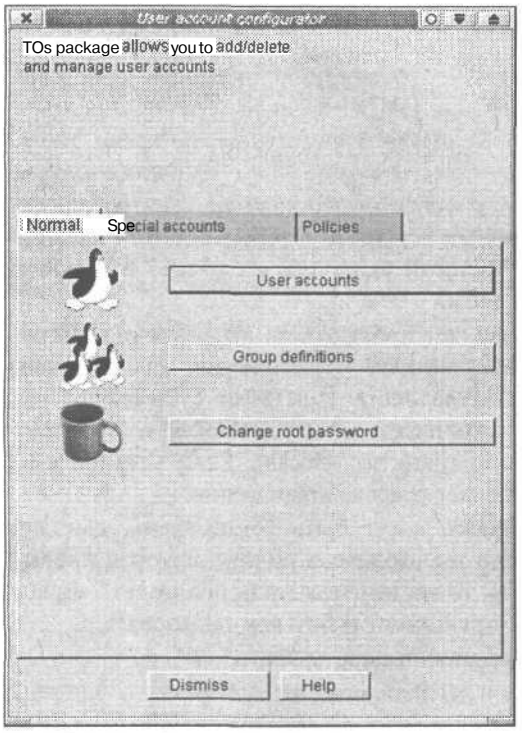


Рис. 3.1. *User Account Configurator*

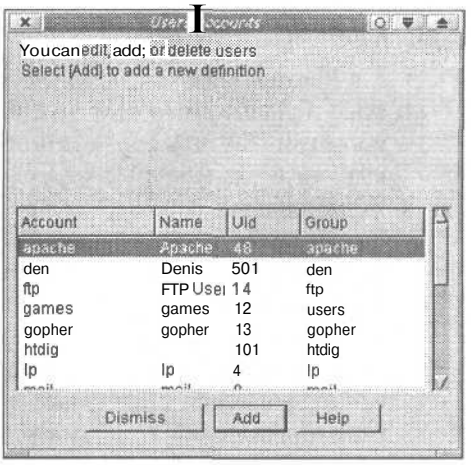
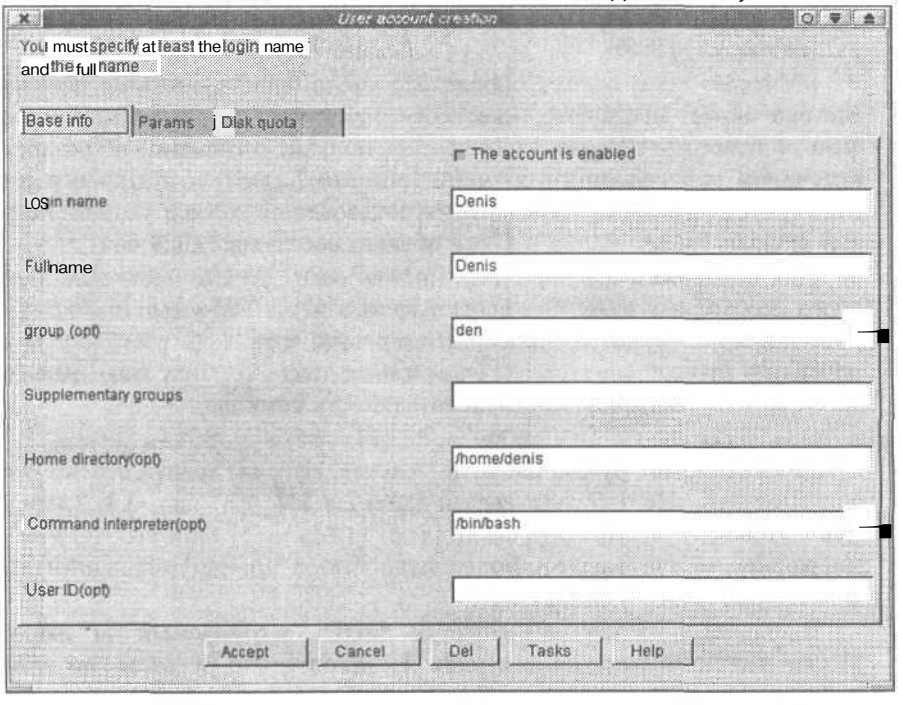


Рис. 3.2. *Учетные записи*

Рис. 3.3. *Добавление учетной записи*



3.6. Квотирование

Квотирование — мощный механизм ограничения дискового пространства, используемый еще в ранних версиях UNIX. Данный материал настолько объемный, что ему можно было бы смело посвятить целую главу. Надеюсь, читатель мне простит, если я немного отклонюсь «от основного курса». Вспоминается народная мудрость: все новое — это хорошо забытое старое. Квоты поддерживались самыми ранними версиями UNIX, тогда как в состав ОС семейства Windows компания Microsoft включила их только в Windows 2000, естественно, гордо заявив об этом.

Команда **quota** (см. дальше) позволяет просмотреть ограничения дискового пространства для данного пользователя. Ограничения, как правило, устанавливаются системным администратором. При этом существуют два типа ограничений — ограничение на количество файлов **(inodes)** и ограничение на использование дискового пространства **(blocks).** Если установлены оба ограничения, то они будут использоваться одновременно.

Ограничения на **inodes и на blocks** могут быть установлены как для пользователя, так и для группы. Если вы входите в группу, которая превысила наложенное на нее ограничение, то вы не сможете использовать дисковое пространство, даже если вы не превысили квоту как пользователь.

Для каждого ограничения характерны четыре числа:

1. Ограничение, которое используется в данный момент.
2. «Мягкое» ограничение (softlimit).
3. «Жесткое» ограничение (hardlimit).
4. Время, по истечении которого «мягкое» ограничение будет интерпретироваться как «жесткое».

«Мягкое» ограничение определяет число блоков, которые пользователь все еще может превысить, «жесткое» ограничение превысить невозможно. При попытке сделать это, пользователь получит сообщение об ошибке. При истечении определенного времени (обычно 7 дней) «мягкое» ограничение переходит в «жесткое». За это время пользователь должен удалить ненужные ему файлы. Размер блока в Linux обычно составляет 1024 байт.

Для поддержки ограничений, прежде всего, нужно настроить ядро. Для этого необходимо включить **quota support** в меню **Filesystem** при конфигурировании ядра, а потом перекомпилировать ядро. С этой целью нужно перейти в каталог `/usr/src/linux` и выполнить команду **make menuconfig**, а затем ввести следующую последовательность команд:

```
make dep; make bzImage; make modules; make modules_install
```

Последние две нужно вводить в случае, если вы хотите перекомпилировать модули. Не забудьте ввести команду **lilo** (см. рис. 3.4) (перед этим нужно немного подправить файл `/etc/lilo.conf`, если вы этого не сделали раньше -- см. гл. 18). Более подробно о конфигурировании ядра вы можете прочитать в гл. 18.

Звездочкой на рис. 3.4 отмечен раздел, загружаемый по умолчанию. Перед перекомпилированием ядра я настоятельно рекомендую прочитать

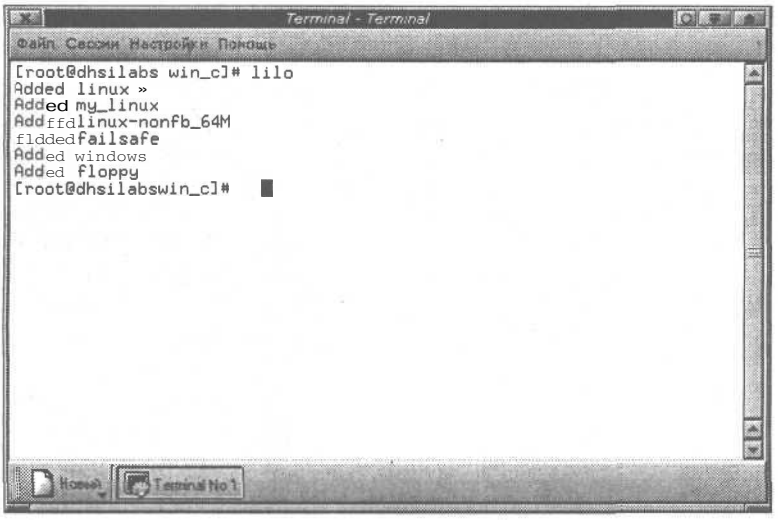


Рис. 3.4. *Запись загрузчика LILO*

п. 4.11 «Использование LILO» и гл. 18. Если при загрузке ядра на экране монитора вы увидите сообщение *Turning on user and group quotas for local filesystems,* это значит, что. ваше ядро уже поддерживает квоты.

Теперь нужно определить, использование какой файловой системы вы хотите ограничить. Обычно это /home (домашние каталоги пользователей), /usr (пользователи имеют право записывать информацию в этот каталог) и, возможно, /var. Для этого отредактируйте файл /etc/fstab следующим образом:

```
/dev/hda1       /        ext2     defaults
/dev/hda4       /home    ext2     defaults,usrquota
/dev/hda5       /usr     ext2     defaults,usrquota,grpquota
none            /proc    proc     defaults
```

Параметр **usrquota** означает ограничение пространства для пользователей на данном устройстве, а **grpquota** — для групп. Если вы пишите **usrquota (grpquota)** без знака «=», то подразумевается, что файлы ограничений находятся в корневом каталоге каждой файловой системы, для которой используются ограничения на дисковое пространство. Обычно эти файлы называются quota.user и quota.group для квот пользователей и групп соответственно. Однако вы сами можете указать какие файлы следует использовать для определения квот, например, **usrquota=/quotas/user.quota**. Старайтесь не указывать слишком длинный путь.

Периодически необходимо проверять содержащиеся в файле ограничения на целостность действительного числа блоков и файлов, выделенных для пользователя. Для этого используется команда **quotacheck.** Ее можно выполнять даже на смонтированных файловых системах, а также на файловых системах, на которых не используются квоты. Для проверки файловой системы на число блоков, которые используются пользователем, выполните команду:

```
# quotacheck -avug
```

В основном данную проверку нужно выполнять при некорректной перезагрузке. При этом вы можете включить эту команду в один из **rc-сценариев** и выполнять ее так же, как и **fsck.**

Для того, чтобы включить систему ограничений при загрузке операционной системы, добавьте команду **quotaon -avug** в сценарии загрузки системы.

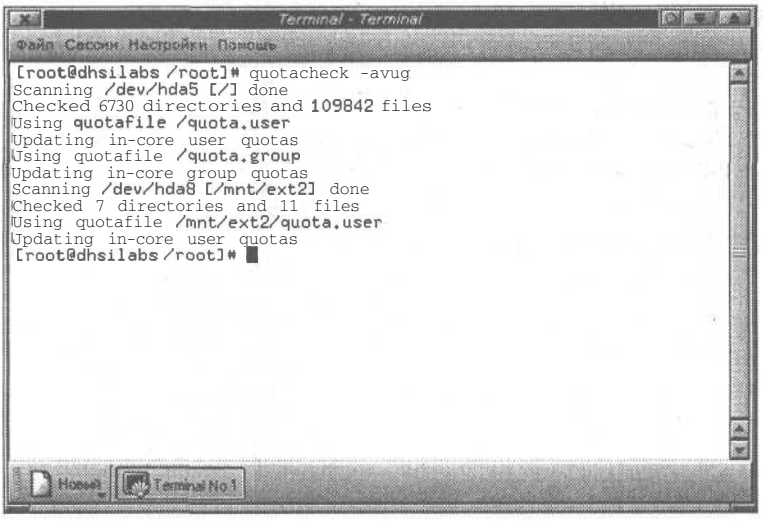


Рис. 3.5. *Результат выполнения команды quotacheck -avug*

Только root имеет право использовать команды, связанные с квотированием. Команда **quota,** как уже было выше отмечено, используется для проверки используемых ограничений любого пользователя, а команда **repquota** – для проверки используемого пространства и ограничений для всех пользователей на данной файловой системе. При выполнении команды **quotacheck -avug** будут автоматически изменены файлы квот, а также будет изменена таблица ядра. Файлы квот имеют размер 2 Мб, даже если ни один из пользователей не использует квот.

Для того, чтобы получить информацию об ограничениях, наложенных на пользователя (группу), используется команда **repquota -ua** (см. рис. 3.6).

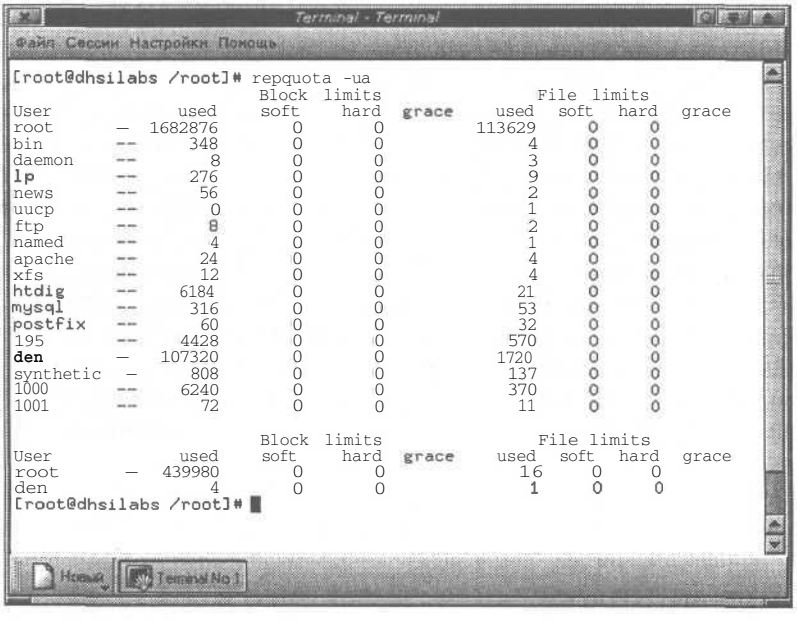


Рис. 3.6. *Результат выполнения команды repquota -ua*

3.6.1. Определение ограничений

Для определения (задания) ограничений используется команда **edquota.** Ограничение дискового пространства пользователя производится командой **edquota** с параметром -и (см. рис. 3.7), а определение квот для группы — с параметром **-g**. После выполнения этой команды будет запущен редактор, который указан в переменной окружения $EDITOR, для редактирования квот. Редактировать надо только число, которое следует за словом **hard или soft.**

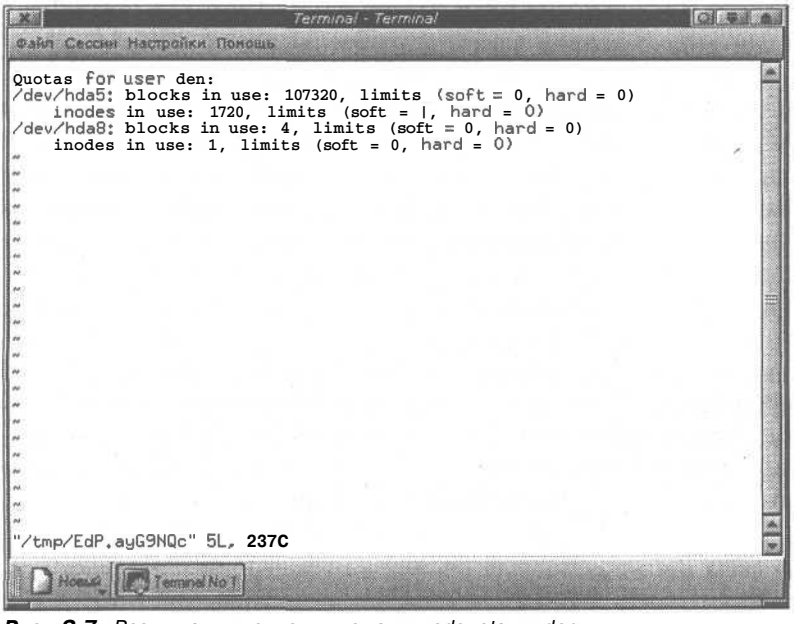


Рис. 3.7. *Результат выполнения команды edquota -u den*

```
# edquota -u den
```

Перед выполнением этой команды выполните две следующие команды:

```
quotacheck -u <filesystem>
quotaon -u <filesystem>
```

Например:

```
quotacheck -u /mnt/ext2
quotaon -u /mnt/ext2
```

Данные команды необходимы для создания файлов `quota.user` и `quota.group`. В противном случае при редактировании ограничений пользователей (групп) вы получите сообщение о том, что данные файлы не существуют.

Для каждой файловой системы, на которую наложены квоты, вы увидите две строки. Слово **soft** означает, что на данную файловую систему наложено «мягкое» ограничение, а слово **hard** — «жесткое». При этом пользователь имеет некоторый интервал времени, по истечении которого «мягкое» ограничение перейдет в «жесткое». Данный интервал можно изменить с помощью команды **edquota -t**:

```
# edquota -t
```

```
Time units may be: days, hours, minutes, or seconds
Grace period before enforcing soft limits for users:
/dev/hda4: block grace period: 50 minutes, file grace period: 50 minutes
```

«Жесткое» ограничение является максимальным значением, которое может иметь пользователь или группа на данной файловой системе.

Строка /dev/hda4: blocks in use: 1024, limits (soft = 1, hard = 0) определяет количество блоков, которое может быть выделено для пользователей или группы. Напомню, что обычно размер блока в Linux составляет 1024 байт. В данном случае ограничение равно 1 Мб.

Строка inodes in use: 94, limits (soft = 0, hard = 0) сообщает какое число **inode** (файлов, устройств, поименованных каналов (pipes)) может быть выделено для данного пользователя или группы.

В большинстве случаев у вас есть группа пользователей, которая должна иметь одинаковые ограничения. Самым быстрым способом редактирования ограничений в этом случае является использование прототипа. С помощью команды:

```
# edquota -u <пользователь/группа_который(ая)_станет_прототипом>
```

можно определить ограничения прототипа, а затем с помощью команды:

```
# edqouta -p <прототип> пользователь
```

создать квоты для всех оставшихся пользователей, применив к ним ограничения прототипа. При этом вам не нужно редактировать ограничения отдельно для каждого пользователя/группы. Например, вам нужно добавить пользователя user, который будет использовать такие же ограничения, что и пользователь den. Делается это следующим образом:

```
# edquota -p den user
```

Команда **quota** используется для проверки ограничений дискового пространства пользователей и групп и применяется со следующими параметрами:

```
quota [-guqv]
quota [-qv] -u <имя_пользователя>
quota [-qv] -g <имя_группы>
```

Параметр -v используется для вывода информации о файловых системах, которые не имеют активных ограничений, а также о файловых системах, на которых квоты уже активны, но не занят еще ни один блок.

Параметр -q используется для получения сведений о файловых системах, на которых превышено значение «мягкого» ограничения.

Параметр -g предоставляет информацию об ограничениях, наложенных на указанную группу.

Параметр -и предоставляет информацию об ограничениях, наложенных на указанного пользователя. Этот параметр используется по умолчанию и аналогичен запуску программы **quota** без параметров.

Например, просмотр ограничения для пользователя user выглядит следующим образом:

```
Disk quotas for user user (uid 1002):
Filesystem  blocks  quota  limit  grace  files  quota  limit  grace
/dev/hdb3   1024*   1      0      none   94     0      0
```

Пользователь user, ограничен так же, как и его прототип — пользователь den.

Если ограничения для данного пользователя не заданы, вы увидите примерно такое сообщение:

```
Disk quotas for user root (uid 0): none
```

Только суперпользователь может просматривать квоты других пользователей. Обычный пользователь может просматривать только свои квоты и квоты группы, к которой он принадлежит.

3.6.2. Запрет квоты для пользователя или группы

Иногда не нужно ограничивать какого-то отдельного пользователя — и в самом деле, не будете же вы ограничивать самого себя? Тогда для этого вам нужно использовать программу **edquota** и установить значения **soft и hard** равными 0. После этого данный пользователь или группа сможет использовать дисковое пространство без ограничений.

3.6.3. Использование программы linuxconf для определения квот

Очень удобно редактировать ограничения с помощью **linuxconf.** Для этого запустите **linuxconf** и выберите в меню **Filesystems -> Set quotas default** (см. рис. 3.8).

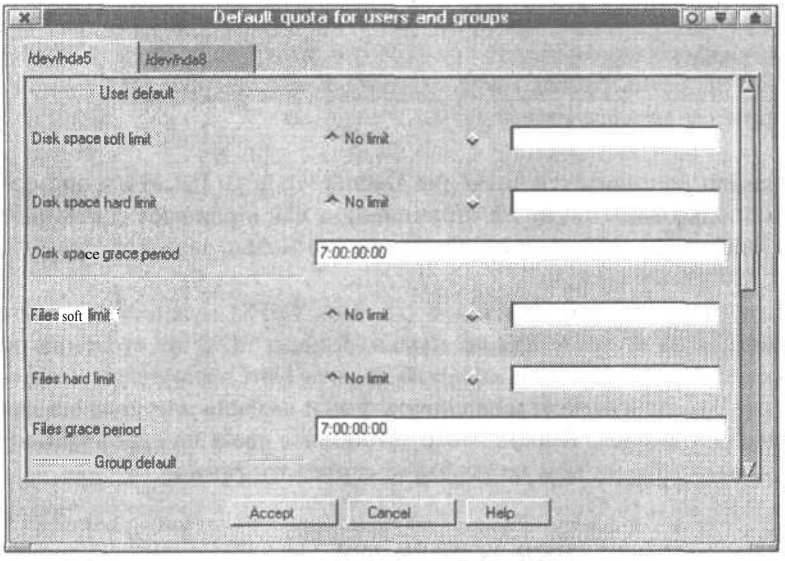


Рис. 3.8. *Редактирование квот программой linuxconf*

Здесь можно указать ограничения, которые будут использованы по умолчанию для пользователей или групп. При этом можно также указать интервал времени (grace period), по истечении которого «мягкое» ограничение перейдет в «жесткое» . Ограничения можно задать отдельно и для конкретного пользователя. Для этого выберите в меню программы **User accounts → User accounts** и укажите пользователя, для которого вы хотите задать ограничения (см. рис. 3.9).

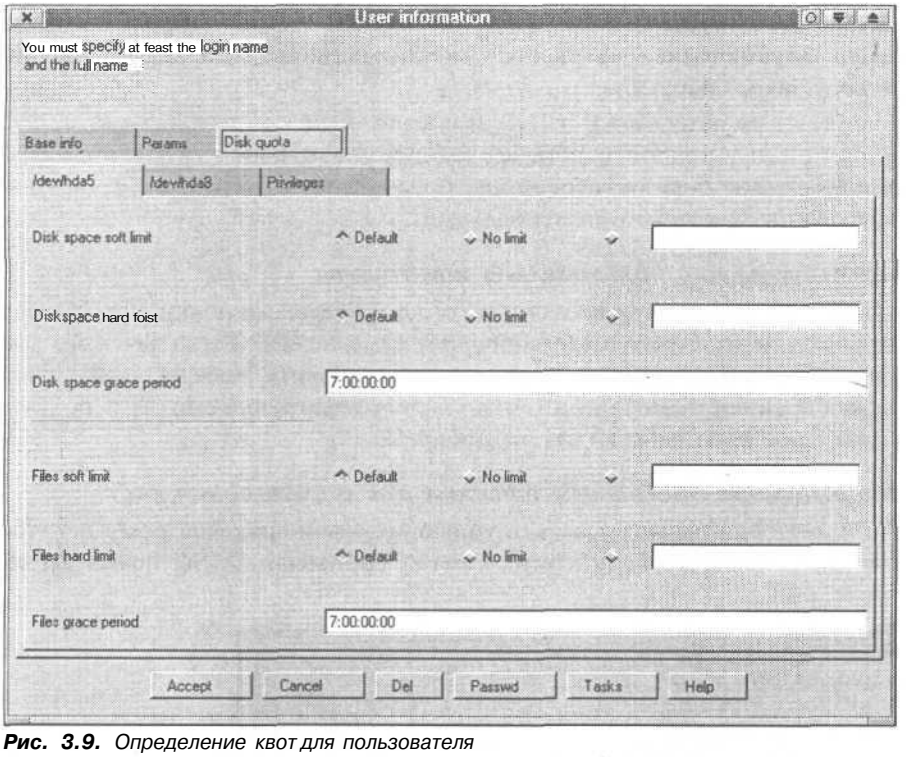


Рис. 3.9. *Определение квот для пользователя*

По умолчанию используется значение **Default,** то есть для этого пользователя будут использованы такие же ограничения, как и для всех остальных. Значение No **limit** — без ограничения. Ограничения можно задавать отдельно и для разных файловых систем.

Для квотирования сетевой файловой системы (NFS) нужно установить **quota** на сервере, а на клиенте она не нужна. Клиент получит сведения об ограничениях с помощью **rquotad,** который должен быть установлен и запущен на сервере. Вам не нужно устанавливать флаги **usrquota** или **grpquota** для монтирования NFS-дисков. Вместо этого установите **quota** на своем файловом сервере и запустите на нем из rc-файла сервер **rpc.rquotad.**

3.7. Сценарий создания пользователей

В качестве завершения этой главы приведу обещанный мною сценарий добавления новых пользователей (см. листинг 3.2). Данный сценарий нужно поместить в каталог `/sbin` и сделать этот файл исполнимым: **chmod 711 /sbin/nu.**

Листинг 3.2. Сценарий nu

```
# !/bin/bash
# nu (New User) — Сценарий добавления пользователей.
I Группа по умолчанию
```

```
GROUP=100
# Оболочка по умолчанию
SHELL=/bin/bash
# Префикс для домашнего каталога
HOME=/home
# Время окончания действия пароля (дни)
EXPIRE=30
# Минимальное количество дней до смены пароля
DAYS=0
# За 5 дней предупреждаем пользователя
WARN=5
WHOAMI=`/usr/bin/whoami`
if [ $WHOAMI!="root" ]; then
    echo "Access violation."
    exit 1
fi
echo -n "Enter new name: "
read USERNAME
echo -n "Enter full name: "
read FULLNAME
adduser -c "$FULLNAME" -d $HOME/$USERNAME -e $EXPIRE  \
-g $GROUP -S $SHELL $USERNAME
passwd -n $DAYS -w $WARN $USERNAME
passwd $USERNAME
```

4

Файловая система Linux

4.1. Файлы и каталоги. Дерево каталогов

В свое время, при использовании DOS вводилось определение файла как поименованной области данных на диске — на то DOS и дисковая операционная система. В Linux понятие файла значительно расширено. Практически все, с чем вы имеете дело в Linux, является файлом. Команды, которые вы вводите с клавиатуры, — это файлы, которые содержат программы. Устройства вашего компьютера — это тоже файлы. Грубо говоря, файл -- это последовательность битов, а жесткий диск — просто смесь нулей и единиц. Linux представляет биты так, как вам понятно, и в этом заключается одна из ее основных функций — управление файловой системой. Файловая система – способ организации и представления битов на жестком диске.

Большинство файловых систем Unix-подобных операционных систем сходны между собой. Файловая система Linux — **ext2 (ext3)** — очень похожа на файловую систему ufs. К основным понятиям файловых систем в мире Unix относятся:

1. Блок загрузки (boot block).
2. Суперблок (superblock).
3. Индексный (информационный) узел (inode).
4. Блок данных (data block).
5. Блок каталога (directory block).
6. Косвенный блок (indirection block).

Блок загрузки содержит программу для первоначального запуска Unix.

В суперблоке содержится общая информация о файловой системе. В суперблоке также содержится информация о количестве свободных блоков и информационных узлов. Для повышения устойчивости создается несколько копий суперблока, но при монтировании используется только одна. Если первая копия суперблока повреждена, то используется его резервная копия.

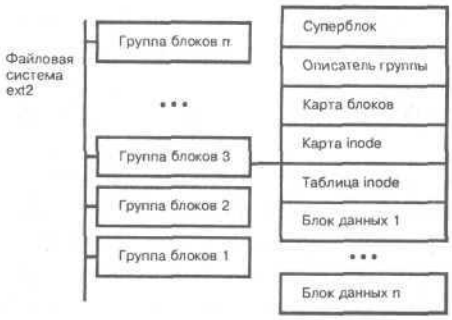


Рис. *4.1. Структура файловой системы*

В индексном (информационном) узле хранится вся информация о файле, кроме его имени. Имя файла и его дескриптор (номер информационного узла — inode) хранятся в блоке каталога. В информационном узле есть место только для хранения нескольких блоков данных. Если же нужно обеспечить большее количество, то в этом случае динамически выделяется необходимое пространство для указателей на новые блоки данных. Такие блоки называются **косвенными.** Для того, чтобы найти блок данных, нужно найти его номер в косвенном блоке.

Файловая система ext2 имеет следующую структуру (см. рис. 4.1):

1. Суперблок.
2. Описатель группы.
3. Карта блоков.
4. Карта информационных узлов.
5. Таблица информационных узлов.
6. Блоки данных.

Суперблок, как уже отмечалось выше, содержит информацию обо всей файловой системе. Он имеется в каждой группе блоков, но это всего лишь копия суперблока первой групы блоков: так достигается избыточность файловой системы.

Описатель (дескриптор) группы содержит информацию о группе блоков. Каждая группа имеет свой дескриптор группы. Карты блоков и информационных узлов — это массивы битов, которые указывают на блоки или информационные узлы соответственно. Таблица информационных узлов содержит информацию о выделенных для данной группы блоков в информационных узлах.

Блоки данных — это блоки, содержащие реальные данные.

Что касается файлов, то в операционной системе Linux существует четыре типа файлов:

1. Файлы устройств.
2. Каталоги.
3. Обычные файлы.
4. Ссылки.

Файлы устройств представляют собой устройства вашего компьютера. Файлы устройств находятся в каталоге /dev. Например, /dev/ttyS0 — первый последовательный порт (COM1). Обычные файлы, в свою очередь, делятся на *нормальные* (текстовые) и *двоичные.*

Каталоги — это специальные файлы, содержащие информацию о других файлах (файлах устройств (/dev), обычных файлов и ссылок). Конечно, это довольно грубое определение, скорее интуитивное, чем точное.

Ссылки позволяют хранить один и тот же файл, но под разными именами. Немного позже мы поговорим более подробно о ссылках, а сейчас рассмотрим команды для работы с файлами и каталогами. Максимальная длина имени файла составляет 254 символа. Имя может содержать практически любые символы, кроме: / \ ? > < | " *

В своей работе я не рекомендую использовать слишком длинные, а также русскоязычные имена файлов. Linux чувствительна к регистру символов, поэтому file.txt, FILE.TXT и File.txt — совершенно разные имена файлов, и данные файлы могут находиться в одном каталоге. Понятие «расширение файла» в Linux отсутствует. Напомню, что в DOS имена файлов строились по схеме 8+3: 8 символов — для имени и 3 — для расширения. Расширением (или типом в терминологии Windows) называется последовательность символов после точки.

Свойства файловой системы ext2:

Максимальный размер файловой системы.....4 Тбайт.
Максимальный размер файла.....2 Гбайт.
Максимальная длина имени файла.....255 символов.
Минимальный размер блока.....1024 байт.
Количество выделяемых индексных дескрипторов....1 на 4096 байт раздела.

4.2. Команды для работы с файлами и каталогами

4.2.1. Команды для работы с файлами

Прежде чем приступить к описанию команд для работы с файлами, необходимо отметить, что для выполнения операций над файлами вы должны иметь права доступа к этим файлам. О правах доступа будет написано ниже, а здесь предполагается, что пользователь имеет права доступа к используемым файлам.

Создание и просмотр файла

Для просмотра файла обычно используется команда **cat.** Например:

```
$ cat file.txt
```

При этом на стандартный вывод, то есть на ваш терминал, будет выведен файл `file.txt`. Однако более удобными командами для просмотра файлов являются команды **more** или **less:**

```
$ less file.txt
```

Создать файл можно так:

```
$ cat > file.txt
```

Здесь используется перенаправление ввода/вывода, о котором подробно сказано в гл. 5. Данные со стандартного ввода (клавиатуры) перенаправляются в файл `file.txt`. Проще говоря все, что вы после этой команды введете с клавиатуры, будет записано в файл `file.txt`. Чтобы закончить ввод нажмите Ctrl+D. Помните, что вы не сможете создать файл в каталоге, к которому у

вас нет доступа. Вы даже не сможете просмотреть файл, если пользователь, которому этот файл принадлежит, запретил чтение этого файла.

Копирование файла

Для копирования файлов в ОС Linux используется команда **cp,** которая имеет следующий формат вызова:

```
$ cp [параметры] источник назначение
```

Рассмотрим несколько примеров:

```
$ cp file.txt file2.txt
$ cp file.txt /home/user/text/
```

В первом случае выполняется копирование файла `file.txt` в файл `file2.txt`. Оба файла находятся в текущем каталоге. Во втором случае — копирование файла `file.txt` в каталог `/home/user/text/`.

Напомню, что вы можете использовать точку (.) в качестве ссылки на текущий каталог, символ тильды (~) — на домашний каталог. Родительский каталог обозначается двумя точками (..). Корневой каталог обозначается символом косой черты (/). Параметры команды **cp** указаны в табл. 4.1.

Параметры команды cp *Таблица 4.1*

Параметр	Описание
-a	При копировании сохраняются атрибуты файлов
-b	Создание копии вместо перезаписи существующего файла
-d	Поддержка символических ссылок. При этом копироваться будут сами символические ссылки без файлов, на которые они указывают
-i	Перед перезаписью существующего файла от пользователя потребуется подтверждение этого
-l	Создание жестких ссылок вместо копирования (при копировании в каталог)
-r	Копирование каталога вместе с подкаталогами
-s	Создание символических ссылок вместо копирования (при копировании в каталог)
-u	Не перезаписывать, если перезаписываемый файл имеет более позднюю дату модификации
-v	Вывод сведений обо всех выполняемых действиях (verbose). Выводит имена всех копируемых файлов
-x	Игнорировать каталоги, расположенные в других файловых системах, по отношению к системе, откуда выполняется копирование

Переименование и перемещение файлов

Команда **mv** перемещает или переименовывает файлы. Например:

```
$ mv file.txt file2.txt
```

Данная команда переименовывает файл `file.txt` в файл `file2.txt`. Можно также перемещать файлы в другой каталог:

```
$ mv ~/*.txt /tmp
```

Эта команда перемещает все текстовые файлы из домашнего каталога пользователя в каталог /tmp.

Будьте очень осторожны при использовании команды **mv:** при перемещении она не предупреждает о существовании файла-назначения и если таковой существует, то он будет перезаписан.

Удаление файла и каталога

Для удаления указанного файла используется команда **rm**. Например: $ **rm** ffle2.txt. При этом для удаления файла пользователь должен иметь право на запись в каталог. Права на чтение или запись файла необязательны. Если нет права на запись в файл, то выдается (в восьмеричном виде) режим доступа к файлу и запрашивается подтверждение на удаление. Если стандартный вывод назначен не на терминал, то команда *rm* будет вести себя так же, как при наличии опции **-f**. А при указании опции **-f** не выдается предупреждений, если удаляемый файл не существует, а также не запрашивается подтверждение при удалении файла, на запись в который нет прав. Если нет права и на запись в каталог, то файлы не удаляются. Сообщение об ошибке выдается лишь при попытке удалить каталог, на запись в который нет прав.

Опция -r предназначена для рекурсивного удаления всех файлов и каталогов, указанных в командной строке. При удалении непустых каталогов команда **rm** с параметром **-r** предпочтительнее, чем команда **rmdir**, поскольку последняя не может удалить непустой каталог.

Опция -i аналогична одноименной опции команды cp и требует подтверждения от пользователя перед удалением каждого файла.

Быстрый поиск файла

Команда **locate** производит поиск заданного файла в файловой системе. Вместо имени файла можно указать образец имени, например, в тех случаях, когда вы забыли точное название файла:

```
$ locate passwd
```

Поиск программы

Если вы не знаете, в каком каталоге находится нужная вам команда (программа), введите команду which, указав в качестве параметра нужную вам команду (программу).

```
$ which awk
```

Данная команда бывает очень полезна в тех случаях, когда вы хотите узнать, установлена ли вообще та или иная программа. Быстрый поиск имени программы можно выполнить прямо из командной строки Linux: для этого введите первые буквы нужной вам команды и нажмите «Tab». Такая функция называется *автозаполнением командной строки*. Для вывода всех доступных команд нажмите «Tab» дважды. Естественно, что полный список команд на одном экране не поместится. Чтобы «листать» консоль, используйте клавиши «PageUp» и «PageDown».

4.2.2. Команды для работы с каталогами

Просмотр содержимого каталога

Для просмотра содержимого каталога используется команда **ls**. Закоренелые пользователи DOS могут использовать привычную им команду **dir**, но команда Is намного удобнее. Программа is имеет около сорока параметров, о назначении которых вы можете узнать в справочной системе, введя команду man **ls**.

Вывод имени текущего каталога

Команда **pwd** сообщит вам имя текущего каталога. Эту команду очень удобно использовать при написании сценариев.

Создание и удаление каталога

Как и в DOS, для создания каталога используется команда **mkdir,** а для удаления — **rmdir**. При удалении каталога нужно учитывать то, что удаляемый каталог должен быть пуст. В противном случае команда заявит о своем бессилии.

Смена каталога

Команда **cd** сменяет текущий каталог на указанный.

Файловый менеджер Midnight Commander

Для вызова Midnight Commander введите команду **тс.** Естественно, пакет тс должен быть предварительно установлен. Midnight Commander очень похож на всем известный Norton Commander, так что я не буду здесь подробно останавливаться на описании данной программы (см. рис. 4.2).

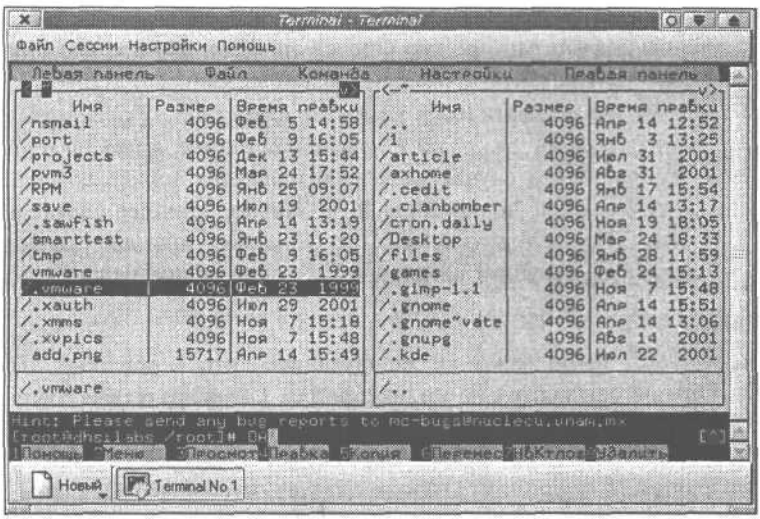


Рис. 4.2. *Midnight Commander*

4.3. Ссылки

Иногда очень полезно, чтобы в каталоге находился один и тот же файл, но под разными именами. Можно просто скопировать этот файл в другой, но при этом неэффективно используется дисковое пространство. Для этих целей в ОС Linux существует специальный тип файлов — ссылки. Ссылки позволяют хранить один и тот же файл, но под разными именами. Linux поддерживает два типа ссылок: жесткие (прямые) и символические.

Жесткие ссылки

Каждый файл в файловой системе Linux имеет свой индекс. Индекс – это уникальный номер файла. Получить информацию обо всех индексах в текущем каталоге можно с помощью команды **ls** -i. Исходя из принятых положений можно дать более точное определение каталога: каталог — это просто список индексов файлов. Допустим, у нас есть файл text. Просмотрим его индекс:

```
ls -i text
25617 text
```

Теперь создадим жесткую ссылку на файл text командой **ln**:

```
ln text words
```

Обратите внимание, что ссылка words на файл text имеет тот же индекс, что и файл text:

```
ls -i words
25617 words
```

Отсюда следует, что жесткие ссылки привязываются к индексу файла. В рамках одной файловой системы вы можете организовывать только жесткие ссылки.

Командой **ln** можно создать множество ссылок на один файл и все они будут иметь один и тот же индекс.

Изменяя файл words, вы автоматически измените файл text. Удаляя файл words, вы можете удалить и файл text, но только в том случае, когда на него нет больше ссылок. В противном случае удалению подлежит только ссылка. Количество ссылок отображается по команде **ln** -l. Число, стоящее слева от имени владельца, и есть количеством ссылок. При этом доступны две ссылки на каталоги: "." - ссылка на текущий каталог, а ".." - на родительский.

Символические ссылки

Символические ссылки выполняют ту же функцию, что и жесткие, но несколько иначе. Они не ссылаются на индекс файла. Символическая ссылка представляет собой специальный файл, при обращении к которому система понимает, что на самом деле нужно обратиться к другому файлу и обеспечивает прозрачность операции. Отсюда следует, что операции с символическими ссылками выполняются медленнее, чем с жесткими. Создать символическую ссылку можно командой **ls** -s, например:

```
ls -s text words
```

Теперь, введя команду **ls** -i, вы увидите, что файлы text и words имеют разные индексы. Посмотрим, какую информацию выведет команда **ls -l** text words:

```
ls -1 text words
lrwxrwxrwx  1  den  group  3 Dec  5 12:11  words -> text
-rw-r-r-    1  den  group  12 Dec  5 12:50  words
```

Обратите внимание на первый символ строки Irwxrwxrwx — символ «l». Это означает, что данный файл является символической ссылкой на файл text, о чем свидетельствует информация в последней колонке words->text.

Символическая ссылка не имеет прав доступа — для нее всегда используется набор rwxrwxrwx. Подробнее о наборах прав доступа будет сказано ниже. А на данном этапе хочется отметить следующее: символические ссылки очень полезны, так как они позволяют идентифицировать файл, на который они ссылаются, тогда как для жестких ссылок нет простого способа определить, какие файлы привязаны к одному и тому же индексу. Однако, если вы удалите файл, на который ссылаются символические ссылки, то получите паутину бесполезных ссылок, которые ни на что не ссылаются. При использовании жестких ссылок вы не сможете удалить файл до тех пор, пока на него ссылается хоть одна жесткая ссылка.

Переменная окружения **$cwd** содержит имя символической ссылки каталога, если такая существует. Просмотреть ее значение можно с помощью команды:

```
echo $cwd
```

4.4. Стандартные имена устройств в Linux

Как уже отмечалось раньше, все устройства в Linux являются файлами. Файлы устройств находятся в специальном каталоге /dev. Для просмотра данного каталога удобнее всего использовать команду **тс.** Запустите тс и перейдите в каталог /dev. Если возле файла слева вы видите +, то данное устройство подключено и функционирует.

В этом пункте я вкратце постараюсь объяснить, какие файлы ассоциируются с какими устройствами. Договоримся, что символ N обозначает номер устройства, например, ttyN обозначает устройства /dev/tty1 .. /dev/ttyN, а x — символ. Наиболее используемые стандартные имена устройств (в соответствии с принятыми обозначениями) приведены в табл. 4.2.

На устройствах hdxN и sdxN необходимо остановиться подробнее. Известно, что к (E)IDE (ATA) контроллеру можно подключить четыре IDE-устройства: Primary Master, Primary Slave, Secondary Master, Secondary Slave.

Наиболее используемые стандартные имена устройств *Таблица 4.2*

Файл	Устройство
TtyN	Консоль
mouse	Мышь
audio	Звуковая карта
modem	Модем. Обычно файл /dev/modem является ссылкой на один из файлов /dev/ttyS0
ttySN	Последовательный порт. Файл /dev/ttySO аналогичен файлу COM4 в DOS
lpN	Параллельный порт
cuaN	Могут обозначать последовательные порты. Используются немного в другом контексте, чем ttySN
hdxN	IDE жесткий диск
sdxN	SCSI жесткий диск
fd0	Первый дисковод для гибких дисков, то есть A:, для B: используется имя /dev/fd1
stN	Стример с интерфейсом SCSI
nrtfN	Стример с интерфейсом FDC
mdN	Массив RAID
ethN	Сетевая плата
null	Пустое устройство

Этим устройствам соответствуют символы: a, b, c, d. Например, /dev/hda — Primary Master, a /dev/hdd — Secondary Slave. Номер N в обозначении устройства обозначает номер раздела на жестком диске. Первичный раздел DOS на первом жестком диске обозначается так: /dev/hda1.

4.5. Стандартные каталоги

В ОС Linux есть каталоги, которые называются стандартными. Иногда их еще называют системными. Эти каталоги присутствуют практически в каждой ОС Linux. В них находятся файлы, необходимые для управления и сопровождения системы, а также стандартные программы. Описание стандартных каталогов сведено в табл. 4.3.

Стандартные каталоги *Таблица 4.3*

Каталог	Назначение
/	Корневой каталог
/bin	Содержит стандартные программы
/home	Содержит домашние каталоги пользователей
/usr	Содержит все программы, используемые системой
/sbin	Команды для системного администрирования
/var	Содержит файлы, которые постоянно изменяются, например, спулы для принтеров, а также файлы почтовых ящиков
/etc	Содержит файл настройки системы
/dev	Здесь находятся файлы устройств
/tmp	Каталог для временных файлов
/mnt	Обычно здесь создаются точки монтирования. Тем не менее, подмонтировать файловую систему можно к любому другому каталогу, а использование каталога /mnt не является обязательным

4.6. Создание файловой системы. Типы файловых систем

Каждая операционная система имеет основной тип файловой системы, а также дополнительные типы, поддержка которых осуществляется модулями (драйверами), подключаемыми к ядру. В случае с Linux поддержку той или иной файловой системы можно встроить непосредственно в ядро. Основной файловой системой Linux на момент написания этих строк является **ext2fs,** однако на ее смену сейчас приходит **ext3fs** и последние версии дистрибутивов Linux используют именно ее. Переход на новую файловую систему обеспечивает более надежную ее работу.

Кроме основной файловой системы, Linux поддерживает файловые системы, указанные в табл. 4.4.

В табл. 4.4 рассмотрены базовые типы файловых систем. ОС Linux поддерживает и другие файловые системы, не указанные в таблице. Поддержку нужной вам файловой системы можно включить при перекомпилировании ядра. Подробно этот процесс рассмотрен в гл. 18. Для нормальной работы вам потребуются только файловые системы, отмеченные звездочкой.

Типы файловых систем *Таблица 4.4*

Файловая система	Комментарий
Minix Filesystem (minix)	Устаревшая и практически неиспользуемая в наше время файловая система
Xia Filesystem (xia)	Редко используется
UMSDOS Filesystem (umsdos)	Использовалась для установки Linux в раздел MSDOS
MSDOS Filesystem (msdos)	Файловая система msdos
(*) VFAT Filesystem (vfat)	Файловая система Windows 9x
NT Filesystem (ntfs)	Файловая система Windows NT
HPFS Filesystem (hpfs)	High Performance FS. Файловая система OS/2
(*) ISO 9660	Файловая система, используемая большинством CDROM
(*) /proc	Предоставляет информацию о процессах
Extended Filesystem (ext)	Устаревшая версия основной файловой системы Linux
(*) Second Extended Filesystem (ext2) или Third Extended Filesystem (ext3)	Основная файловая система
Network Filesystem (nfs)	Сетевая файловая система

Внимание!

Старайтесь никогда не использовать файловую систему ***msdos.*** *Она поддерживает только имена в формате 8+3 (так называемые короткие имена файлов). Вместо нее нужно использовать файловую систему* ***vfat.*** *Она поддерживает «длинные» имена файлов и нормально работает со старыми разделами (дискетами), отформатированными под файловую систему* ***msdos.***

Прежде чем перейти к созданию файловой системы, хочу рассмотреть несколько новых файловых систем, которые, скорее всего, становятся стандартом в настоящее время. Ядро 2.4.8 уже поддерживает файловые системы: Ext3, ReiserFS, XFS.

Список файловых систем, которые поддерживаются ядром системы, содержится в файле `/proc/filesystems`. Просмотреть этот список поможет команда cat /proc/filesystems.

Файловая система Ext3 (Third Extended Filesystem) представляет собой журналируемую надстройку над ext2, поэтому возможно чтение одной файловой системы как драйвером **Ext3**, так и драйвером **Ext2**. Возможно отключение журналирования. Файловую систему **ext2** можно конвертировать в ext3, запустив программу создания журнала. После конвертирования новую файловую систему можно использовать и без журнала — для этого достаточно примонтировать ее драйвером для ext2.

RaiserFS — журналируемая файловая система. Основной ее особенностью является способность хранить несколько мелких файлов в одном блоке.

XFS — также журналируемая файловая система, первоначально разрабатывалась компанией Silicon Graphics (SGI) для ОС Irix. Особенностью этой файловой системы является устройство журнала: в журнал пишется часть метаданных самой файловой системы таким образом, что весь процесс восстановления после сбоя сводится к копированию этих данных из журнала в файловую систему. Размер журнала задается при создании системы, он должен быть не меньше 32 мегабайт.

JFS первоначально разрабатывалась компанией IBM для AIX OS, позднее была перенесена на OS/2, а не так давно и под Linux. Размер журнала

составляет примерно 40% от размера файловой системы. Максимальный размер равен 32 мегабайтам. Эта файловая система может содержать несколько сегментов, содержащих журнал и данные. Эти сегменты называются агрегатами и могут монтироваться отдельно.

Все эти файловые системы предназначены для создания высокопроизводительного файлового сервера или рабочей станции, ориентированной на работу с файлами больших размеров. Какая из них лучше — трудно сказать. Нужно исходить из потребностей.

Производительность JFS ниже, чем у остальных трех файловых систем, но она более предсказуема по своему поведению, то есть можно с большой вероятностью предсказать, когда начнется падение производительности. XFS обладает значительно большими показателями производительности. Особенно хорошо она себя проявляет при работе с файлами больших размеров. Производительность этой файловой системы можно значительно повысить, если создать журнал на отдельном контроллере.

Файловая система ReiserFS показала еще большую производительность, но трудна в прогнозировании падения производительности. Файловая система ext3 практически по всем параметрам производительности мало чем отличается от ReiserFS.

Вот тут я слишком часто употребляю слово «журналируемая». Давайте же разберемся, что же собой представляет журналируемая файловая система, и в чем состоят ее преимущества.

Прежде всего нужно отметить, что журналируемые файловые системы не предназначены для восстановления ваших данных любой ценой после сбоя. Они предназначены для других целей. Например, вы открываете файл, и он успешно открывается — файловая система отмечает операцию открытия в своем журнале записью транзакции. Затем вы начинаете писать в файл. При этом файловая система не запоминает копии этих данных. Затем происходит сбой. Когда происходит восстановление после сбоя, происходит откат до последней успешной транзакции — открытия нового пустого файла. Поэтому, когда вы пишите в файл и происходит сбой, вы получите файл нулевой длины.

Давайте рассмотрим создание основной файловой системы типа ext2. А создать файловую систему такого типа можно с помощью команды:

```
mke2fs -c <устройство>
```

Опция -c указывает программе, что нужно сделать проверку устройства на наличие ошибок. В качестве устройства обычно выступает раздел жесткого диска. Некоторые опции команды **mke2fs** перечисленны в табл. 4.5. Полный список опций с описанием вы можете получить, введя команду man **mke2fs**.

Естественно, прежде чем создавать файловую систему на жестком диске, необходимо создать на нем разделы с помощью программы fdisk. Linux в качестве устройства может использовать обыкновенный файл. Его можно создать командой dd. Затем файловую систему нужно примонтировать командой mount. Обо всем этом будет сказано немного позднее. Сейчас же рассмотрим, как перейти от обыкновенной файловой системы **ext2** к журналируемой системе ext3. Как я уже отмечал, после/конвертирования с файло-

Параметры mke2fs *Таблица 4.5*

Опция	Описание
-b размер_блока	Устанавливает размер блока. Допустимыми являются 1024, 2048 и 4096 байт/блок. Если размер блока не указан, то mke2fs устанавливает его исходя из размера и типа файловой системы (см. опцию -T)
-f размер_фрагмента	Устанавливает размер фрагмента в байтах
-l имя_файла	Читает список «плохих» блоков из указанного файла
-T тип_ФС	Устанавливает оптимальные параметры для файловой системы указанного типа
-т количество_блоков	Резервирует указанное количество блоков под суперпользователя. Задается в процентах от общего количества. По умолчанию резервируется 5%
-c	Перед созданием файловой системы производит проверку устройства на предмет присутствия «плохих» блоков
-v	В процессе работы выдает подробную информацию

вой системой **ext3** можно будет работать в режиме **ext2,** отключив журналирование. Другими словами, просто нужно будет использовать драйвер **ext2.**

Если вы используете операционную систему Red Hat 7.2 или Mandrake 8.1 и выше, то, скорее всего, у вас уже будет установлена файловая система ext3. Если же вы во время установки не включили поддержку **ext3,** то сейчас самое время это сделать, хотя переходить на новую файловую систему или нет, решать только вам.

Прежде чем перейти к практике, прочитаем немного теории о новой файловой системе. Файловая система **ext3** имеет два основных преимущества перед **ext2.** Первое состоит в том, что **ext3** записывает изменение данных и метаданных, что позволяет сохранять содержимое файлов. Второе преимущество состоит в том, что разделы **ext3** ничем не отличаются от разделов **ext2,** поэтому всегда можно перейти к старой файловой системе и наоборот. Главным здесь является то, что вы можете спокойно делать резервную копию файловой системы **ext3,** а потом развернуть ее на **ext2-**разделе. Позже можно будет включить журналирование.

Немного определений:

Метаданные (metadata) -- это данные, которые являются описанием других данных (например, схема базы данных по отношению к содержимому базы данных).

Журналирование — это запись изменения метаданных во время совершения транзакции. В журнал записывается состояние трех типов данных: метаданных, блоков описания и блоков-заголовков. Уровень журналирования (то, что будет писаться в журнал) можно указать при монтировании файловой системы с помощью программы mount. Журналируемый блок всегда записывается полностью, даже если произошло маленькое изменение. Это делается очень быстро, так как операции журналируемого ввода/вывода объединены в большие кластеры.

Блоки описания описывают другие метаданные. Запись блоков описания происходит перед записью метаданных. **Блоки-заголовки** описывают заголовок и окончание журнала. Каждому блоку-заголовку присваивается порядковый номер, чтобы гарантировать упорядоченную запись во время восстановления.

Теперь перейдем непосредственно к практике. Для того, чтобы наилучшим образом понять этот материал, вам следует прочитать пункт этой главы о монтировании дисков, а также гл. 18. Тем не менее, дальнейший материал изложен таким образом, что перекомпилировать ядро вы сможете и не читая гл. 18, просто на данном этапе вы не все поймете. В этом разделе книги я попытаюсь как можно подробнее описать процесс перехода на новую систему.

Переход на файловую систему **ext3** нужно начинать, собственно, с включения поддержки новой файловой системы. При этом необходимо перекомпилировать ядро. С этой целью перейдите в каталог `/usr/src/linux` и запустите программу make **menuconfig.** В разделе *Filesystem* отметьте файловую систему **ext3.** Если эта опция уже включена, то ядро уже поддерживает файловую систему **ext3.**

Примечание.

Для поддержки ext3 необходимо ядро версии 2.4.7 или выше. В дистрибутиве Linux Red Hat 7.2 используется ядро версии 2.4.7.

После этого согласитесь сохранить изменения в конфигурации ядра и выполните следующие команды:

```
make dep
make bzImage
make modules
make modules_install
make install
```

Некоторые из этих команд, возможно, вам и не понадобятся, а какие именно, вы узнаете в гл. 18. Однако вы уже сейчас можете вводить эти команды — они будут у вас работать, но при этом на их выполнение понадобится гораздо больше времени, так как это общий случай — для всех.

Первая команда производит необходимую подготовку к компилированию ядра. В принципе, она необязательна, но относится к категории весьма желательных. Вторая собирает само ядро. Две следующих — собирают и устанавливают модули ядра. Последняя команда устанавливает ядро. После выполнения последней команды желательно ввести команду **lilo** для перезаписи главной загрузочной записи MBR. Кроме этого, желательно перезагрузить компьютер (не подумайте, что по рекомендациям Microsoft!) для того, чтобы убедиться, что собранное ядро работает. Если же ядро отказалось работать, то загрузитесь с системной дискеты Linux (создается при установке) и повторите процесс сборки ядра.

Затем нужно создать журнал командой

```
tune2fs -i 0 -c 0 -j /dev/hda1
```

Эта команда создает и конвертирует корневую файловую систему типа **ext2,** расположенную на устройстве /dev/hda1, в файловую систему ext3. На этом, собственно, весь процесс конвертирования можно считать завершенным. Остается только сказать системе, что ей нужно использовать драйвер **ext3.** Для этого откройте в любом текстовом редакторе файл /etc/fstab, в котором вы должны увидеть примерно такую строку:

```
/dev/hda1 / ext2 defaults,usrquota,grpquota 1 1
```

Обычно она самая первая строка в файле. Эта строка означает, что корневая файловая система (/) расположена на устройстве /dev/hda1 и для нее используется драйвер **ext2**. Просто замените **ext2** на **ext3** и сохраните изменения. После этого можно перезагрузить компьютер. Аналогично вы можете конвертировать другие **ext2**-разделы: выполните команду tune2fs для каждого раздела и измените драйвер в файле /etc/fstab.

4.7. Использование программы fdisk

Программа **fdisk** для Linux используется при создании разделов под Linux. Естественно, ее можно использовать для создания разделов и других типов. Каждая операционная система имеет свою версию **fdisk.** При этом рекомендуется для создания разделов конкретной операционной системы использовать ее «родную» версию **fdisk.** Запуск **fdisk** производится следующим образом:

```
fdisk <диск>
```

Дальнейшее изложение материала будет построено на примере, в котором я буду издеваться над старым жестким диском, имеющем 683 цилиндра. Честно говоря, это устройство как-то страшно даже назвать жестким диском — скорее устройством, выполняющим его функции.

Итак, запускаем **fdisk,** указав параметр /dev/hda. Если этого не сделать, то вам придется созерцать недовольное ворчание программы по этому поводу — это вам не DOS.

```
# fdisk /dev/hda
Comamnd (m for help):
```

Для получения справки можно ввести m. На экране вы увидите примерно следующее:

```
Command action
a toggle a bootable flag
d delete a partition
1 list known partition types
m print this menu
n add a new partition
p print the partition table
q quit without saving changes
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)
```

Нам потребуются только команды p, n, q и w.

Для начала распечатаем таблицу разделов:

```
Command (m for help): p
Disk /dev/hda: 16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes
```

```
Device      Boot  Begin   Start   End    Blocks   Id   System
/dev/hda1   *     1       1       203    61693    6    DOS 16-bit >=32M
```

Здесь видно, что имеется один DOS-раздел приблизительно на 60 Мб (1 блок = 1024 байт). Этот раздел начинается с первого цилиндра и заканчивается на 203-ем. Всего на диске 683 цилиндра, т.е. для создания раздела Linux у нас осталось 480 цилиндров. Создаем новый раздел (команда n):

```
Command (m for help):  n
Command action
e extended
p primary partition (1-4)
P
```

Задаем тип создаваемого раздела *(первичный* или *расширенный).* Разница между первичным и расширенным разделами состоит в следующем:

1. Может существовать только 4 первичных раздела.
2. В первичных разделах могут храниться данные, а расширенный раздел также содержит информацию о разделах, которые находятся в нем.

В нашем примере будет 2 первичных раздела — один раздел для файловой системы Linux *(Linux Native)* и один раздел для подкачки *(Linux Swap):*

```
Partition number (1-4):  2
First cylinder (204-683):  204
Last cylinder or +size or +sizeM or +sizeK (204-683): +80M
```

Номер первичного раздела -- 2. Первый цилиндр — 204. Последний цилиндр вычисляется автоматически. Здесь можно ввести непосредственно номер последнего цилиндра, но это неудобно. Проще ввести размер в байтах +размер, в Кб или в Мб (*+размерК*или *+размерМ*соответственно).

Теперь создадим второй раздел для свопинга.

```
Command (m for help): n
Command action
e extended
p primary partition (1-4)
P
Partition number (1-4):  3
First cylinder (474-683): 474
Last cylinder or +size or +sizeM or +sizeK (474-683):  +10M
```

По умолчанию **fdisk** создаст раздел типа *Linux Native* (81). Нам нужно изменить его на тип *Linux Swap* (82). Это можно сделать с помощью команды t, а с помощью команды l можно просмотреть доступные типы разделов.

Теперь распечатаем таблицу разделов:

```
Command (m for help): p
  Disk /dev/hda: 16 heads, 38 sectors, 683 cylinders
  Units = cylinders of 608 * 512 bytes
  Device      Boot  Begin  Start  End   Blocks   Id   System
  /dev/hdal   *     1      1      203   61693    6    DOS 16-bit >=32M
  /dev/hda2         204    204    473   82080    81   Linux Native
  /dev/hda3         474    474    507   10336    82   Linux Swap
```

Номера цилиндров 508...683 не использованы — здесь можно создать дополнительные разделы.

Далее нужно ввести команду w для того, чтобы записать таблицу разделов на диск:

```
Command (m for help): w
```

До тех пор, пока вы не введете эту команду, ни одно из сделанных изменений не даст эффекта. Вы можете в любой момент выйти из программы без сохранения изменений, введя команду q.

Теперь, согласно традиции, нужно упомянуть о проблеме цилиндра с номером 1024. Старые версии Linux не могут загружаться с разделов на цилиндрах, номера которых превышают 1024. Поэтому раздел для корневой файловой системы нужно стараться разместить до цилиндра № 1023. Однако использовать разделы на цилиндрах, номера которых превышают 1024, Linux может.

4.8. Программа Disk Drake

Программа **Disk Drake** обладает понятным графическим интерфейсом и запускается из-под X Window. Эта программа входит в состав инсталлятора Linux Mandrake, и именно ее вы используете, когда формируете разделы на вашем винчестере при установке этого дистрибутива. По своим возможностям она очень напоминает Partition Magic, да и интерфейс мало чем отличается. Вы можете создавать и удалять разделы, изменять размер и тип файловой системы. В состав Red Hat Linux входит всем известный *Disk Druid.* Программа похожа на **Disk Drake,** но, на мой взгляд, менее удобна.

Пусть я рискую показаться читателю аскетом, но я предпочитаю использовать программу **fdisk.** Что и вам советую, так как **fdisk** — стандартная программа и, вне зависимости с каким дистрибутивом работаете, вы всегда сможете выполнить любые операцию по переразбиению жесткого диска на разделы, используя **fdisk.** Согласен, **fdisk** не умеет изменять размер раздела без потери данных, но лучше уж уметь использовать **fdisk,** чем пытаться запустить **Disk Drake,** работая со Slackware...

4.9. Монтирование дисков. Файл /etc/fstab

Как уже отмечалось ранее, прежде чем использовать файловую систему, ее нужно примонтировать к основной файловой системе. Определим правила работы со сменными носителями (СН) данных (CDROM, Floppy, Zip) и монтируемыми файловыми системами (ФС):

1. Прежде чем использовать СН (или ФС), его нужно примонтировать к корневой файловой системе. Каталог, через который будет производиться доступ к файлам СН (ФС), называется точкой монтирования.
2. Если вы хотите сменить СН, его нужно размонтировать, сменить на новый и смонтировать заново.

3. Если вы не хотите использовать СН (ФС), его нужно размонтировать. При останове системы размонтирование производится автоматически.
4. Вы не можете размонтировать СН (ФС), который в данный момент используется.

Для монтирования ФС предназначена программа **mount,** для размонтирования — **umount.** Общий формат вызова (наиболее часто используемый) следующий:

```
mount -t fs_type device mount_point
```

В качестве параметра fs_type программы **mount** указывается тип подключаемой файловой системы, некоторые из которых я позволю себе еще раз напомнить в табл. 4.6.

Основные типы файловых систем *Таблица 4.6*

Тип	Описания
ext2 или ext3	Файловая система Linux
Vfat	Файловая система Windows 9x
iso9660	Ее нужно использовать при монтировании CD-ROM
Ntfs	Всем известная NT Filesystem

Следующим параметром является устройство *(device).* В качестве устройства выступает носитель данных, например /dev/hdd. Далее, наконец, задается сама точка монтирования (*mount_point*). Примонтировать файловую систему вы можете к любому каталогу корневой файловой системы. Я рекомендую создать подкаталог в каталоге /mnt с осмысленным именем и к нему подмонтировать нужную вам файловую систему. Например, для монтирования дисковода A: вы можете использовать следующую команду:

```
mount -t vfat /dev/fdO /mnt/flopyy
```

При этом считается, что дискета отформатирована для файловой системы vfat.

Для монтирования привода CD-ROM вы можете воспользоваться следующей командой:

```
mount -t iso9660 /dev/hdd /mnt/cdrom
```

Привод CD-ROM подключен ко второй шине IDE как ведомый (Secondary Slave).

Для размонтирования достаточно указать точку монтирования или устройство в качестве параметра команды **umount.** Например, команда **umount /mnt/floppy** размонтирует диск A:. Информация об устройствах, смонтированных на данный момент, содержится в файле /etc/mtab. Программа **mount** имеет опции, представленные в табл. 4.7.

Параметры программы mount *Таблица 4.7*

Опция	Описания
-a	Монтирование всех файловых систем, указанных в файле /etc/fstab, кроме тех, для которых указан параметр noauto
-n	Монтирование без записи в файл /etc/mtab. Полезно, если каталог /etc доступен только для чтения
-r	Монтирование в режиме «только чтение»
-w	Монтирование в режиме «чтение/запись» (по умолчанию)
-t тип_ФС	Задает тип файловой системы

Вы можете комбинировать опции, например, команда **mount -a -t vfat** монтирует все ФС типа VFAT. Список файловых систем, которые поддерживает ядро вашей системы, находится в файле `/etc/filesystems` или в файле `/proc/filesystems`.

Для того, чтобы файловая система монтировалась автоматически при загрузке системы, нужно внести определенную запись в файл `/etc/fstab`. Формат записей в этом файле следующий:

device mount_point fs_type options **флаг_резервного_копирования флаг_проверки**

где:

device — устройство, которое нужно подмонтировать;

mount_point — точка монтирования;

fs_type — тип файловой системы;

options — набор опций монтирования (см. табл. 4.8);

флаг_резервного_копирования — если установлена (1), то программа dump включит данную ФС в архив при создании резервной копии (дампа). Если установлен (0), то резервная копия ФС создаваться не будет;

флаг_проверки — этот флаг устанавливает порядок, в котором файловые системы при монтировании будут проверяться на наличие ошибок. Поиск и исправление ошибок при этом осуществляется специальной программой fsck, которая запускается сценарием инициализации системы. Этот флаг означает очередь, в которой будет проверяться данная файловая система. Если для нескольких файловых систем указан один и тот же номер, то эти файловые системы, при подходе очереди, будут проверяться одновременно. Правильная настройка флагов проверки позволяет ускорить загрузку. Корневая файловая система всегда должна иметь значение флага проверки (1), которое означает, что ее необходимо проверять первой. Для всех остальных файловых систем рекомендуется устанавливать значение (2), которое позволит произвести их проверку одновременно, сразу же после проверки корневой файловой системы. Значение (0) указывается для файловых систем, проверку которых производить не нужно. К таким ФС относятся съемные файловые системы (носители Floppy, CD-ROM, и т.д.).

Опции монтирования ФС в файле /etc/fstab *Таблица 4.8*

Опция	Описание
exec	Разрешает запуск бинарных (выполняемых) файлов для данной файловой системы. Эта опция используется по умолчанию
noexec	Запрещает запуск бинарных (выполняемых) файлов для данной файловой системы
noauto	Данная файловая система не может быть смонтирована с помощью команды mount -a, то есть не может быть смонтирована при загрузке системы
auto	Данная файловая система будет автоматически смонтирована во время загрузки. Эта опция используется по умолчанию
ro	Монтирование в режиме «только чтение»
rw	Монтирование в режиме «чтение/запись». Эта опция используется по умолчанию
user	Разрешает пользователям монтировать/демонтировать данную файловую систему
nouser	Запрещает пользователям монтировать/демонтировать данную файловую систему. Эта опция используется по умолчанию
defaults	Использовать стандартный набор опций, установленных по умолчанию

А сейчас я себе позволю несколько небольших комментариев относительно опций монтирования. Опцию noexec полезно устанавливать для файловых систем, в которых вы не предполагаете запускать программы. Ее полезно установить для файловой системы **vfat**: запускать там нечего, а вот при копировании из нее файлов в файловую систему **ext2** не будет устанавливаться право на выполнение только что скопированного файла. О правах доступа поговорим немного позже.

Если вы установите опцию **noauto**, данную систему нельзя будет подмонтировать с помощью опции -a программы mount. Команда mount -a обычно выполняется при запуске системы, а значит, данная файловая система не будет подмонтирована автоматически. Это очень полезно для сменных устройств, например, дискет или магнитооптических дисков, когда нужно просто задать какие-нибудь параметры для данной файловой системы, но не монтировать ее. Ведь при запуске системы в приводе может не оказаться дискеты или магнитооптического диска. Опция user позволяет пользователю монтировать данную файловую систему. Обычно она используется вместе с опцией noauto для сменных дисков. Пример файла конфигурации файловых систем /etc/fstab приведен в листинге 4.1.

Листинг 4.1. Файл /etc/fstab

```
/dev/hda1   /           ext2    defaults        1   1
/dev/hda2   /den        ext2    defaults        0   2
/dev/hda3   /home       ext2    defaults        0   2
/dev/hda4   swap        swap    defaults        O   O
/dev/fdO    /mt/floppy  vfat    noauto,noexec   O   O
/dev/hdd    /mt/cdrom   iso9660 noauto,ro       O   O
none        /proc       proc    defaults        O   O
```

В первой строке содержится запись, задающая параметры монтирования корневого раздела «/» и указывающая, что устройство /dev/hdal имеет файловую систему ext2 и должно быть смонтировано со стандартным набором опций defaults, используемых по умолчанию. Кроме этого, в записи сказано, что необходимо создавать резервную копию данной файловой системы, и что устройство должно быть проверено на наличие ошибок при загрузке системы, причем в первую очередь.

Вторая и третья записи содержат информацию о том, что устройства /dev/hda2 и /dev/hda3 содержат файловую систему **ext2** и должны быть смонтированы со стандартными установками в каталоги /den и /home соответственно. Резервные копии данных файловых систем создавать не нужно, а проверку при загрузке ОС необходимо производить во вторую очередь, причем одновременно обеих.

Четвертая строка содержит запись о параметрах монтирования раздела подкачки (swap). Для этого, а также для всех последующих разделов указано, что не надо ни создавать их резервную копию, ни производить их проверку при загрузке.

В пятой и шестой строках монтируются устройство чтения дискет (Floppy) и CD-ROM. Последняя строка файла /etc/fstab определяет специальную файловую систему /proc, которой вообще не ставится в соответствие никакое устройство (none). Файловая система **/proc** предназначена для обеспечения интерфейса взаимодействия с внутренними структурами данных ядра.

В процессе настройки вы, наверное, заметите, что при монтировании файловой системы **vfat** вместо русских букв отображается не совсем то, что вам бы хотелось. Например, в лучшем случае вместо имени каталога Мои документы вы увидите ??? ??????????. Для перекодирования русскоязычных (и не только) имен файлов из одной кодировки в другую используются опции монтирования **iocharset** и **codepage**. Непосредственно для vfat нужно указать:

```
codepage=866, iocharset=koi8-r
```

4.10. Создание раздела (файла) подкачки

Рано или поздно при работе с Linux вам станет недостаточно оперативной памяти: потребности растут, а возможности (оперативная память) остаются прежними. В этом случае нужно купить дополнительные 128...256 Мб, тем более, что цены на память постоянно снижаются. Временным выходом из данного положения может послужить создание дополнительного файла или раздела подкачки. Еще раз замечу, что для Linux нет разницы, с чем работать: с файлом или с разделом.

Создайте раздел подкачки с помощью fdisk (тип раздела 82) и используйте команду **mkswap**, чтобы отформатировать его как раздел подкачки. Формат использования команды mkswap следующий:

```
mkswap -c раздел размер
```

Например, следующая команда создаст раздел свопинга размером 32 Мб (параметр -c используется для проверки «плохих» блоков):

```
mkswap -c /dev/hda3 32768
```

Если лишний раз переразбивать жесткий диск вам лень, можно создать не раздел, а файл подкачки, который будет впоследствии использоваться в качестве раздела подкачки. Для этого сначала создайте пустой файл /swap/sw-file (размер 32 Мб) с помощью команды dd:

```
dd if=/dev/zero of=/swap/sw-file bs=1k count=32768
```

Примечание.

*Эта команда читает данные с устройства /dev/zero и записывает их в файл /swap/sw-file. В качестве данных будет просто поток нулей, причем не чисел ноль (ANSII-код 48), а неотображаемых символов NULL (ANSII-код 0). Данные читаются и записываются блоками по 1 Кб (bs=1k), и общее количество блоков равно 32768. Таким образом, на выходе будет получен файл размером 32 Мб, заполненный символами NULL. Действия по созданию такого файла очень сходны с действиями, производимыми программой **fdisk** при создании нового раздела.*

После этого отформатируйте данный файл под своп:

```
mkswap /swap/sw-file 32768
```

Заметьте, что никто вам не мешает создать в этом файле файловую систему и использовать ее, например:

```
mke2fs -m 0 /swap/sw-file
```

затем:

```
mount -t ext2 /mnt/disk1 /sw/sw-file
```

Параметр -m задает процент блоков, которые будут зарезервированы для суперпользователя (по умолчанию — 5%).

После создания раздела (файла) подкачки, его нужно активизировать. Команда **swapon -a** включает все разделы свопинга (описанные в файле /etc/fstab), а команда **swapon <раздел>** включает только конкретный раздел. Команда **swapon** -a обычно помещается в сценарий загрузки системы. Обычно это /etc/rc.d/rc.sysinit для систем, использующих инициализацию типа SysV — RedHat, Mandrake, Debian (хотя RedHat и Mandrake используют несколько модифицированную схему инициализации, но суть та же) или /etc/rc/rc.S для BSD-подобных Linux-систем (Slackware).

В рассматриваемом случае для подключения раздела подкачки нужно выполнить команду **swapon /dev/hda3,** а для подключения файла подкачки необходимо выполнить команду **swapon /swap/sw-file**.

Обратите внимание, что файл подкачки не может быть автоматически активизирован с помощью команды **swapon -a,** так как он не может быть указан в файле /etc/fstab. Для того, чтобы файл подкачки /swap/sw-file автоматически активизировался при загрузке системы, команду **swapon /swap/sw-file** нужно включить в сценарий загрузки после команды **swapon -a.** В противном случае вам придется эту команду вводить каждый раз вручную после загрузки системы.

Убедиться, что ваш файл (раздел) подкачки активизирован, можно, выполнив команду **free** (см. рис. 4.3). Команда **free** выводит информацию об

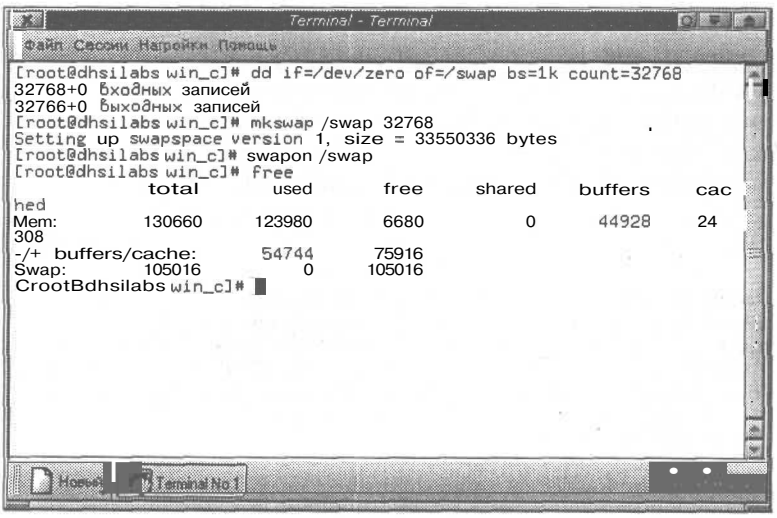


Рис. 4.3, *Создание и активизирование файла подкачки*

использовании памяти, в том числе и о виртуальной. В колонке total выводится общий размер памяти и подкачки (Mem и Swap соответственно), в колонке used — размер занятой памяти, а free — свободной.

После активизирования файла (раздела) подкачки общий размер свопа (Swap Total) должен увеличиться на размер только что активизированного раздела — в нашем случае на 32 Мб.

4.11. Использование LILO

Linux LOader (LILO) — программа, предназначенная для загрузки Linux и других операционных систем.

Существуют другие загрузчики, например, bootlin, bootact, bootstar, но они постепенно вытесняются LILO. Помимо LILO Linux еще можно загрузить с помощью loadlin, GRUB (загрузчик в Linux Mandrake) или **NTLoader**. Об использовании NTLoader и программы loadlin уже писалось ранее, во второй главе. Рассмотрим теперь подробнее LILO. Загрузчик LILO состоит из трех частей:

1. Программа записи начального загрузчика **lilo**.
2. Программа конфигурации **liloconf**.
3. Файл конфигурации `/etc/lilo.conf`.

Программа liloconf создает файл `/etc/lilo.conf`, который используется программой Шо для записи начального загрузчика. Обычно LILO записывают в MBR (Master Boot Record). Но иногда LILO устанавливают на первый сектор того раздела, где установлен Linux. Второй способ обычно используется, если нужно обеспечить загрузку Linux:

1. С помощью другого загрузчика, например, NTLoader.
2. На старых машинах без поддержки LBA.

При загрузке компьютера LILO выдает подсказку:

```
LILO
```

или

```
LILO boot:
```

После чего нужно ввести метку той операционной системы, которую вы хотите загрузить. Для загрузки Linux обычно следует ввести **linux**. Для просмотра всех доступных меток нажмите «Tab». Современные версии LILO обычно имеют удобное меню. Выбор меток осуществляется с помощью стрелок.

Иногда на экран только выдается подсказка:

```
LILO
```

Чтобы выбрать ядро, нужно нажать клавишу «Shift», после чего появится подсказка:

```
LILO boot:
```

и только теперь можно нажать «Tab». Если вы введете команду help, то получите список всех команд LILO.

Поведение LILO зависит от его настройки в файле `/etc/lilo.conf`, который является конфигурационным файлом LILO. При запуске Linux

можно передать ядру дополнительные параметры, например mem=1024M **устанавливает объем ОЗУ равным 1024 Мб. При этом можно самим сформировать строку параметров и записать** ее в lilo.conf **- эта строка будет передана ядру при загрузке Linux. С помощью LILO можно организовать загрузку других операционных систем (Windows, FreeBSD,..) и загрузку разных версий ядра одной ОС (имеется в виду Linux). В листинге 4.2 приведен пример файла** /etc/lilo.conf.

Листинг 4,2. Пример конфигурационного файла /etc/lilo.conf

```
#0перационная система: Linux Mandrake
#LILO version: 21.5
# Глобальные опции
# Загрузочное устройство (MBR на /dev/hda)
boot=/dev/hda
I «Карта» загрузки. Если этот параметр пропущен,
# используется файл /boot/map
map=/boot/map
# Устанавливает заданный файл как новый загрузочный сектор.
# По умолчанию используется /boot/boot.b
install=/boot/boot.b
# compact — не используйте этот режим. Обычно он
# используется при загрузке с дискеты
# Режим VGA: normal -- 80x25, ext -- 80x50
vga=normal
# Образ ядра по умолчанию. Если не задан, то используется
# первый в списке
default=linux
# Раскладка клавиатуры
keytable=/boot/ru4.klt
# Включен режим LBA32. На некоторых дисках может вызвать проблемы
# Обычно проблемы возникают на старых
# компьютерах без поддержки трансляции блоков (LBA)
lba32
# Включает ввод приглашения без нажатия на какую-нибудь клавишу.
# Автоматическая загрузка невозможна, если prompt установлен, а
# timeout — нет
prompt
# Задержка 5 секунд (в некоторых версиях используется delay)
timeout=50
# Подсказка, которая выдается при загрузке
message=/boot/message
# Цветовая схема
menu-scheme=wb:bw:wb:bw
# Пароль (ко всем образам)
# password=54321
```

```
# Пароль нужен для загрузки образа, если параметры задаются
# в командной строке (для всех образов)
# restricted
# Список образов. Максимум 16 вариантов
image=/boot/vmlinuz              # ядро
label=linux                      # метка (метки должны быть разными)
root=/dev/hda5                   # корневая файловая система
       append=" mem=64M"          # объявление параметров ядра
       vga=788
       read-only                 # монтирование корневой
# файловой системы в режиме «только чтение»
# Параметры vga, password, restricted могут быть как глобальными,
# так и отдельными для каждого образа.

# Т.е. вы можете закрыть паролем определенный образ
image=/boot/vmlinuz              # image — для Linux-систем
       label=linux-nonfb
       root=/dev/hda5
       append=" mem=64M"
       read-only
image=/boot/vmlinuz
       label=failsafe
       root=/dev/hda5
       append=" mem=64M failsafe"
       read-only
other=/dev/hdal          t other — какая-нибудь другая система
       label=windows  # для не Linux-систем параметр root не указывается
       table=/dev/hda # определяет устройство, содержащее таблицу разделов
other=/dev/fdO
       label=floppy
       unsafe              # не давать доступ к boot-сектору во время создания
# карты диска. Запрещает проверку таблицы разделов
# Параметры table и unsafe несовместимы.
```

Внимание!

Для того, чтобы изменения вступили в силу (если вы изменили файл конфигурации), нужно выполнить команду lilo.

При конфигурировании LILO вы можете дополнительно использовать следующие опции:

disk=параметры ..... определяет нестандартные параметры для заданного диска.

disktab=файл_таблицы ... задает имя таблицы параметров диска. По умолчанию это /etc/disktab и менять его не рекомендуется не рекомендуется.

ignore-table ..... игнорирование ошибочных таблиц разделов.

nowarn ..... запрещает сообщения о возможных неприятностях.

serial=параметры ..... разрешает управление с последовательной линии. Загрузчик берет ввод из заданного последовательного порта и из клавиатуры. Клавиша «break» по последовательной линии аналогична «Shift» на клавиатуре.

Параметры: `<port>[,bps[parity[bits]]]`

port = 0. .3 .... соответствует /dev/ttySO — /dev/ttyS3. Могут быть задействованы все 4 порта.

bps: ..... скорость порта в бодах. По умолчанию 2400. Поддерживаются значения: 110, 150, 300, 600, 1200, 2400, 4800, 9600.

parity: ..... контроль по четности. n — без четности, e — по четности, o — по нечетности.

bits: ..... число битов в символе — 7 или 8. По умолчанию 8, если нет контроля четности.

Параметры по умолчанию: 0,2400n8.

Опции **append, ramdisk, read-only, read-write, root, vga** могут быть заданы в разделе глобальных параметров. Они будут использованы, если они не заданы в конфигурационных секциях.

Программу LILO можно использовать вместе с параметрами, указанными в табл. 4.9. Эти параметры указываются в командной строке.

Параметры программы LILO *Таблица 4.9*

Параметр	Описание
-V	Выводит номер версии LILO
-P	Карта загрузки
-C имя_файла	Задает конфигурационный файл, который должен использоваться вместо стандартного файла /etc/lilo.conf
-r файловая_система	Устанавливает указанную файловую систему в качестве корневой (выполнит команду enroot перед выполнением каких-нибудь действий)
-и	Удаляет LILO
-I	Запрашивает полное имя файла ядра операционной системы Linux
-R	Устанавливает командную строку ядра Linux по умолчанию

Два небольших замечания:

1. Программа **chroot,** как вы уже заметили, используется для изменения корневой файловой системы.
2. Удалить LILO можно и с помощью команды DOS **fdisk /mbr.**

Параметр -q загрузчика LILO выводит карту загрузки системы. Обычно карта загрузки отображается при записи загрузчика LILO, например:

```
linux    *
windows
```

Если вам нужно просмотреть текущую карту загрузки без записи загрузчика, выполните команду Шо -q.

Кроме LILO сейчас популярен загрузчик **GRUB.** В Linux Mandrake этот загрузчик используется по умолчанию, хотя я рекомендую вам сменить его

на стандартный загрузчик Linux — LILO. GRUB «видит» такие файловые системы: ext2, ext3, ReiserFS, vfat и еще несколько других. Одной из особенностей загрузчика GRUB является возможность загружать операционную систему, которая не присутствует в меню.

Следует также упомянуть такие загрузчики:

Xboot.....крошечный и безопасный загрузчик. Xboot просто меняет активный раздел при загрузке. Обязательным требованием является установка всех загружаемых систем в первичный раздел (а не в расширенный). Это ограничивает число возможных установленных операционных систем до 4 (4 первичных раздела).

Symon.....бесплатный загрузчик. Имеет несколько недостатков, но работает вполне стабильно. Под недостатками следует понимать невозможность установить пароль на загружаемый раздел (как в LILO), а также некорректная работа с некоторыми версиями OS/2.

BootStar... коммерческий бутменеджер. Имеются версии инсталляторов для DOS и Windows. В его состав входят средства для переразбиения разделов, подобные Partition Magic. Несмотря на свое коммерческое происхождение, довольно нестабильно работает, и я не рекомендовал бы его вам использовать.

vpart.....загрузчик, корректно работающий с LVM OS/2 (LVM появился в версии OS/2 4.5).

xosl (Extended Operating System Loader) .... загрузчик, разработанный Гертом Восом (Geurt Vos), распространяется по лицензии GPL. Ознакомиться с xosl вы можете по адресу http://home.wanadoo.nl/geurt/download.html.

gag (GRAPHICAL BOOT MANAGER) .... еще один загрузчик, распространяемый по лицензии GPL. Загрузить его вы можете по адресу http://www.rastersoft.com/programas/gag/downeng.html.

4.12. Права доступа

Для каждого файла и каталога в ОС Linux задаются права доступа. Права доступа определяют, кто имеет доступ к объекту и какие операции над объектом он может выполнять. Под объектом следует понимать файл или каталог. Выполнять можно три основных операции: чтение, запись и выполнение.

Право на чтение файла означает, что его можно просматривать и печатать, а для каталога — что может отображаться список содержащихся в нем файлов. Право на запись для файла означает возможность его редактирования, а для каталога — возможность создания и удаления в нем файлов.

Если для файла установлено право выполнения, то его можно запускать как программу. Данная возможность используется при написании сценариев командных интерпретаторов. Право выполнения для каталога означает право доступа к каталогу, но не право на выполнение расположенных в нем файлов, как это может показаться исходя из названия режима доступа. Чтобы разобраться с правом выполнения для каталогов, проведите неболь-

шой эксперимент, выполнив несколько описанных далее действий. Все действия нужно проводить от имени обыкновенного пользователя, а не от имени суперпользователя root.

Создайте каталог:

```
mkdir dir1
```

Затем создайте в нем файл script:

```
#!/bin/sh
echo "Hello"
```

Измените права на чтение и выполнения для файла:

```
chmod 500 dir1/script
```

Попробуйте просмотреть содержимое каталога:

```
dirl ls -l dirl
```

А теперь измените права доступа к каталогу и повторите предыдущий шаг:

```
chmod 600 dirl
ls -1 dirl
```

Как результат установленных прав на чтение и запись для каталога вы получите сообщение:

```
Permission denied
```

Теперь попробуйте записать что-нибудь в каталог (мы же установили право на запись):

```
cat >> dirl/script
```

Получите то же сообщение:

```
Permission denied
```

В общем случае существует три категории пользователей: владелец, группа и прочие.

Владелец — пользователь, создавший файл. Само собой разумеется, для того, чтобы создать файл, вы должны иметь право записи в каталог, в котором вы создаете файл. При создании файла обычно устанавливаются права на чтение и запись для владельца, и только чтение для всех остальных пользователей.

Пользователи объединяются в группы, например, для работы над одним проектом. Владелец может разрешить или запретить доступ к файлам для членов группы.

Прочие — это все остальные пользователи.

Для каждой категории имеется свой набор прав доступа, просмотреть который вы можете с помощью команды **ls –l**:

```
ls -1 file.txt
-rw-r— 1 den group 300 Feb 10 12:00 file.txt
```

Строка **-rw-r**———характеризует набор прав доступа к файлу file.txt. Владельцем файла является пользователь den, который входит в группу group.

Первый символ — это тип файла. «-» означает файл, а «d» — каталог.

Следующие три символа «rw-» задают права доступа для владельца файла. Символ «r» -- это право на чтение, «w» -- на запись, а «x» -- на выполнение. Права задаются именно в таком порядке: чтение, запись, выполнение. Если право на какой-нибудь вид доступа отсутствует, то ставится «-». В данном случае отсутствует право на выполнение.

Второй трехсимвольный набор задает права доступа для группы, а третий — для прочих пользователей. В нашем примере (r—) члены группы имеют право только на чтение, а другие пользователи вообще не имеют никакого доступа к файлу (—).

Для изменения прав доступа используется программа **chmod.** При этом права доступа можно задавать двумя способами: символьным и абсолютным. Рассмотрим сначала символьный метод, а потом — абсолютный.

В рамках символьного способа изменения прав вызов программы **chmod** имеет следующий вид:

chmod права <файл|каталог>

Параметры программы **chmod** указаны в табл. 4.10.

Права доступа (символьный метод) *Таблица 4.10*

Опция	Описание
+	Устанавливает право доступа
-	Отменяет право доступа
=	Присваивает набор прав доступа
r	Право на чтение файла или каталога
w	Право на запись файла или каталога
x	Право на выполнение
u	Устанавливает право доступа для пользователя, который создал файл и является его владельцем
g	Устанавливает права доступа для группы
o	Устанавливает права доступа для прочих пользователей
a	Устанавливает права доступа для владельца, группы и прочих пользователей
s	Устанавливает бит смены идентификатора пользователя или группы
t	Устанавливает sticky-бит

Бит смены идентификатора пользователя или группы является вариантом права выполнения x. Право на чтение, запись и выполнение обозначается в этом случае не **rwx**, а **rws**. Так называемый *sticky-бит* позволяет оставить программу в памяти после ее выполнения. Устанавливать этот бит полезно для маленьких и часто используемых программ, чтобы ускорить их запуск.

Программа **chmod** никогда не изменяет права символических ссылок, но это не является особой проблемой, так как права ссылок никогда не используются. Изменить группу файла можно командой **chgrp**, а владельца — **chown**.

Теперь перейдем к абсолютному методу указания прав доступа, который, как мне кажется, несколько удобнее, чем символьный, поскольку не нужно помнить символику прав доступа. Этот метод еще называют методом двоичных масок.

Для изменения прав доступа абсолютным методом используется та же команда **chmod:**

chmod число <файл|каталог>

Число называется маской прав доступа и представляет собой число в восьмеричной системе, задающее наборы прав доступа. Напомню, что восьмеричная система — это система с основанием 8 (см. табл. 4.11). Не спешите переворачивать страницу, услышав слово «восьмеричная», все на самом деле намного проще, чем звучит.

Каждое число, задающее права доступа, состоит из трех разрядов, например, 760:

7.....первый разряд;

6.....второй разряд;

О.....третий разряд.

Первый разряд задает права доступа для владельца файла, второй — для группы, третий — для остальных пользователей. Одному разряду восьмеричной системы соответствует три разряда в двоичной.

Соответствие разрядов восьмеричной системы разрядам в двоичной системе *Таблица 4.11*

Восьмеричный формат	Двоичный формат
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Каждый двоичный разряд задает соответствующий ему тип доступа: первый — чтение, второй — запись, третий — выполнение. Будем считать, что разряды нумеруются слева направо. О — если данная операция запрещена, а 1 — если разрешена. Теперь все становится на свои места, например, право доступа, задаваемое числом 777 (111 111 111) означает право на чтение, запись и выполнение для всех пользователей.

Наиболее часто используется маска 644, разрешающая чтение и модификацию файла для владельца, и только чтение — для других пользователей. Иногда используется набор прав доступа, состоящий из четырех цифр. Старшая цифра обозначает флаги SETUID (4), SETGID (2), sticky-бит (1).

Какие же права доступа будут у только что созданного каталога командой **mkdir?** Обычно маска прав доступа равна 0777 минус значение, выводимое командой **umask**. Значение, выводимое командой umask, обычно равно 022. Следовательно, маска прав доступа будет равна 0777 - 0022 = 0755. И действительно, создайте каталог и введите команду **ls –l**:

```
mkdir new
ls -l
...
drwxr-xr-x 2 den den 4096 Янв 14 14:30 new/
...
```

Набор **rwx** (111) равен семи, а набор **r-x** (101) равен пяти в восьмеричной системе. Получается, что маска доступа к новому каталогу = 755. Установить значение **umask** можно с помощью самой этой команды. Просто выполните команду **umask 000** от имени суперпользователя и вы установите новое значение, равное нулю. Обратите внимание, команда **mkdir** автоматически устанавливает право выполнения (доступа) для каталога, иначе к каталогу нельзя будет получить доступ.

***Права доступа** SUID и SGID*

Кроме рассмотренных выше прав доступа в системе Linux имеются еще два специальных права доступа — **SUID** (Set User ID root) и **SGID** (Set Group ID root). Их существование связано с тем, что некоторые программы (**pppd**, **smbmount**, SVGA-программы) требуют для своей работы привилегий пользователя root.

Допустим, что вы хотите, чтобы другие пользователи могли устанавливать PPP-соединение, для этого нужно установить право доступа **SUID** для программы-демона `/usr/sbin/pppd`. Делается это так:

```
chmod u+s /usr/sbin/pppd
```

После этого демон **pppd** будет запускаться с привилегиями **root,** даже . если он запущен самым обыкновенным пользователем. Запуск **pppd** в таком режиме необходим для того, чтобы обыкновенный пользователь смог настроить необходимые интерфейсы и таблицу маршрутизации ядра, то есть подготовить систему для PPP-соединения. Обычно на такие изменения (изменение таблицы маршрутизации ядра, конфигурирование интерфейсов) есть право только у пользователя root.

То же самое можно сказать и о программе **smbmount** — для ее работы тоже нужно установить право доступа **SUID.**

Казалось бы, все хорошо: и пользователи довольны, и вам не нужно каждый раз подходить к серверу, если нужно установить коммутируемое соединение или примонтировать общий ресурс. Однако следует учитывать, что программы, требующие установления **SUID** (или **SGID)** для своей работы являются потенциальными дырами в системе безопасности. Представьте такую ситуацию: у вас в системе установлена программа **superformat**, которая предназначена для форматирования дисков. Создание файловой системы, пусть даже на дискете, -- это привилегированная операция, требующая права доступа **root.** При установке этой программы для нее сразу устанавливается право **SUID,** чтобы разрешить пользователям форматировать дискеты. Пользователь запускает ее для форматирования диска. Программа запускается, получает права root, форматирует дискету и нормально завершает работу. А теперь представим, что программа некорректно завершает свою работу — по ошибке, например, произошло переполнение стека (такие случаи известны при работе с этой программой). Что же при этом произошло: программа **supermount** получила права **root** и некорректно завершила работу. **В результате чего обыкновенный пользователь получил права root!** Неквалифицированный пользователь с правами **root** — это намного хуже, чем просто крах системы. Нельзя с уверенностью сказать что произойдет, если пользователь получит права **root.** Выходит, что не нужно быть профессиональным хакером — достаточно просто уметь форматировать дискеты.

Помните о потенциальной опасности при работе с такими программами и, по возможности, избегайте использования прав **SUID и SGID.**

Ради справедливости нужно заметить, что ряд системных программ (в частности **pppd)** разрабатывался с учетом прав **SUID и SGID,** и эти программы являются максимально защищенными, хотя полной уверенности в этом нет. Поэтому использовать право **SUID** нужно только в самых крайних случаях.

Я позволю себе сделать еще несколько замечаний относительно прав доступа **SUID и SGID:**

1. Лучше не использовать программы, требующие привилегированные права доступа, на сервере, точнее, не разрешать обыкновенным пользователям их использовать. Использование права доступа SUID вы можете себе позволить только на своей домашней машине, например, для установления того же коммутируемого соединения, чтобы каждый раз при подключении к Интернет не вводить команду su.
2. Если все-таки нужны привилегированные права, используйте программу **sudo** (или su). Конечно, это не так удобно, но зато гораздо безопаснее.
3. Перед использованием программ, требующих права доступа root, убедитесь в их надежности. Если программа получена из ненадежного источника, лучше ее не использовать. Под надежным источником подразумеваются сайты или FTP-серверы разработчиков дистрибутивов Linux. Желательно получить исходный код такой программы, чтобы убедиться, что она не производит каких-либо несанкционированных действий.
4. Нет ни одной причины, по какой нужно было бы разрешить использование SUID-программ в домашних каталогах пользователей. Для разделов, в которые разрешена запись обыкновенным пользователям, установите опцию **nosuid** в файле `/etc/fstab`.

4.13. Обслуживание файловой системы

Обслуживание файловой системы в ОС Linux сводится к двум операциям:

1. Проверка.
2. Дефрагментация.

Проверка и восстановление файловой системы Linux выполняется программой **fsck.** Перед проверкой файловая система должна быть смонтирована в режиме «только чтение».

Программа **fsck** умеет проверять и другие типы файловых систем, но для исправления ошибок лучше использовать родные для этой файловой системы программы, предназначенные для проверки и исправления ошибок. Программа **fsck** автоматически проверяет файловые системы при загрузке Linux в соответствии с указаниями, содержащимися в файле etc/fstab. Для программы **fsck** можно использовать опции, указанные в табл. 4.12.

Параметры программы fsck *Таблица 4. /2*

Опция	Описание
-A	Проверка всех файловых систем, указанных в файле /etc/fstab. Корневая файловая система будет проверена первой
-C	Показывать индикатор выполнения проверки (только для файловой системы ext2)
-P	Используется вместе с опцией -A. Проверять корневую файловую систему параллельно вместе с другими файловыми системами, а не перед ними. Не очень безопасный вариант проверки, поэтому лучше его не использовать
-R	Используется вместе с опцией -A. При этом проверяться будут все файловые системы, кроме корневой
-1 тип_ФС	Используется с опцией -A. Указывает только какого типа файловые системы следует проверять. В поле <тип_ФС> указывается тип файловой системы. Если перед типом файловой системы поставить префикс, то проверяться будут ФС всех типов, кроме указанного
-s	Производить проверку файловых систем последовательно, а не параллельно, как это делается по умолчанию (за исключением корневой ФС)

Дефрагментация

Для повышения производительности файловой системы я использую программу **defrag,** написанную Полем Харгровом (Paul H. Hargrove, hargrove@sccm.Stanford.edu). Программа умеет проверять файловые системы типов `ext2`, `minix`, `xia`.

4.14. Подключение магнитооптического диска

Я решил написать этот раздел в силу большой распространенности магнитооптических дисков. Первые магнитооптические диски подключались к контроллеру SCSI, что не способствовало их широкому распространению из-за довольно высокой стоимости. После выпуска первых устройств с интерфейсом IDE цены на магнитооптические устройства значительно снизились.

Подключение магнитооптического привода подобно подключению жесткого диска. При этом следует помнить простое правило: не нужно подключать к одной шине магнитооптический привод и жесткий диск. Логика проста: магнитооптические диски обладают довольно низкой производительностью по сравнению с жестким диском, и использование двух этих устройств на одной шине снизит общую производительность дисковой подсистемы.

После подключения не забудьте выполнить процедуру AUTODETECT для данного устройства. ОС Linux определит магнитооптический привод как обыкновенный жесткий диск с интерфейсом IDE. Если вы поспешили и, запустив Windows, чтобы полюбоваться новой буквой в списке доступных дисков, отформатировали ваш магнитооптический диск, то, скорее всего, в Linux он будет работать некорректно. Для обеспечения нормальной работы магнитооптического привода в Linux запустите **fdisk** для Linux и удалите все разделы, которые создала Windows. Затем создайте один первичный раздел и командой t измените его тип на FAT32.

Хочу отметить, что FAT32 может работать довольно медленно, но позволит сэкономить около 80 Мб дискового пространства при использовании магнитооптического диска размером 640 Мб.

4.15. Использование стримера

ОС Linux, как и UNIX, обладает богатыми возможностями по созданию и сопровождению резервных копий с помощью стримеров. **Стример** — это потоковый накопитель на магнитной ленте. Стримеры работают в безостановочном режиме, обеспечивают запись и считывание данных с ленты сплошным потоком. Основным преимуществом стримеров является их дешевая стоимость, но они имеют ряд недостатков:

1. Работать с жестким диском или магнитооптическим диском намного удобнее, чем со стримером.
2. Медленная скорость передачи данных.

Хотя второе никак не относится к более дорогим стримерам с интерфейсом SCSI.

Здесь следует упомянуть о типах стримеров относительно их интерфейса. Существует два типа стримеров: стримеры, использующие интерфейс SCSI, и стримеры, использующие интерфейс FDC. Первые из них довольно дорогие. Это объясняется дороговизной самого контроллера SCSI. Хотя в последнее время наблюдается снижение цен на контроллеры и устройства SCSI. Эти стримеры подключаются к шине SCSI.

Второй тип, использующий интерфейс FDC, подключается к контроллеру гибких дисков. Это более дешевый и медленный вариант и поэтому, если вы собираетесь использовать стример в профессиональных целях, лучше приобрести стример с интерфейсом SCSI. Второй тип более подходит для домашнего применения.

4.15.1. Подключение стримера с интерфейсом SCSI

ОС Linux поддерживает все возможные стримеры с интерфейсом SCSI. Это объясняется интеллектуальностью контроллера SCSI. Вы также можете использовать интерфейс LUN (Logical Unit Number), который является расширением интерфейса SCSI, для подключения стримера с автоматической заменой ленты.

Для подключения стримера вам потребуется перекомпилировать ядро системы, включив опцию SCSI **Tape Support.** Также вам нужно установить тип контроллера SCSI в подразделе **SCSI Low-Level drivers.** Возможно, нужно будет включить режим **Probe all LUNs on each device.** После перезагрузки в вашей системе появится устройство /dev/st0.

4.15.2. Подключение стримера с интерфейсом FDC

В зависимости от типа вашего стримера вам нудно включить опцию **QIC-02 tape support** или **Ftape (QIC-80/Trawan) support.** О том, какую из этих опций нужно использовать, вы можете прочитать в документации, поставляемой со стримером. Данные опции находятся в разделе **Character devices.** После перезагрузки должно появиться устройство /dev/nrtf0.

4.15.3. Управление стримером

Управление стримером выполняет программа **mt.** Она входит в состав пакета **mt-st,** который обычно входит в состав дистрибутива. Эта программа точно есть в дистрибутивах Red Hat и Mandrake Linux. Программа **mt** использует устройство /dev/nftape, которое является ссылкой на /dev/nrft0. Если вы используете стример с интерфейсом SCSI, вам нужно изменить ссылку на /dev/stO.

После подключения стримера необходимо подготовить ленту к работе. Вся подготовка состоит из перетяжки ленты и ее форматирования. При перетяжке с поверхности пленки снимаются статические заряды. Перетяжку можно выполнить командой:

```
mt-st -f /dev/nftape retension
```

А вот форматирование вам придется выполнять с помощью программы для DOS, которая поставляется со стримером. Можно, конечно, использо-

вать и другие программы. Стабильно работают **Conner Backup Basics, Norton Backup,** а также **QICstream**. По завершении этого процесса требуется инициализировать ленту:

```
mt-st -f /dev/nftape erase
```

Вот теперь можно приступать к резервированию данных. Например, если вы хотите записать на ленту содержимое своего домашнего каталога, вы можете использовать следующую команду:

```
tar cfz /dev/nftape /home/den
```

Здесь я использовал команду tar, которая и предназначена, по правде говоря, для работы с лентой (это видно из ее названия — Tape Archive). Опция z указывает программе tar о необходимости сжать данные. Для записи без сжатия достаточно опций **cf.** Для восстановления архива с ленты вы можете использовать команду:

```
tar xzf /dev/nftape
```

Если вы не использовали сжатие данных, то вам нужно применить команду **xf.**

Проверить целостность архива на ленте можно с помощью команды:

```
tar df /dev/nftape
```

Для того, чтобы поместить на ленту два или более архивов tar, вы должны использовать программу **mt-st** для позиционирования головки и перемотки ленты. Вам нужно будет использовать tar вместе с **mt-st.** Например, чтобы перейти на две отметки начала файла формата tar на ленте, можно использовать команду:

```
mt-st -f /dev/nftape fsf 2
```

Для возврата назад необходимо использовать операцию **bsf** вместо **fsf.**

При работе с mt-st доступны операции, описание которых приведено в табл. 4.13.

Операции программы mt-st *Таблица 4.13*

Операция	Описание
eof	Поместить маркер конца файла в текущую позицию ленты
asf n	Перейти к файлу с номером n
fsf n	Перемотка ленты вперед на n файлов
bsf n	Перемотка ленты назад на n файлов
fsr n	Перейти на n записей вперед
bsr	Перейти на n записей назад
seek n	Поиск блока с заданным номером n
eom	Переход к концу записей на ленте. Используется для дописывания файлов на ленту
rewind	Перемотка ленты в начало
offline	Перемотка пленки в начало и извлечение ленты из накопителя
retension	Перетяжка ленты для снятия статических зарядов с поверхности пленки
erase	Стирание содержимого ленты

4.16. Стратегия резервного копирования

Успешно, во всяком случае, я на это надеюсь, разобравшись с технической стороной создания резервных копий, переходим к организационным вопросам. А именно, вам нужно определиться с ответами на следующие вопросы:

1. Какая информация будет резервироваться (архивироваться)?
2. Когда будет происходить создание резервных копий?
3. Кто этим будет заниматься?
4. Как часто будет производиться архивирование?

Какая информация будет резервироваться (архивироваться)?

В первую очередь вам нужно архивировать данные пользователей, то есть каталог /home. Эти данные относятся к наиболее критичной категории данных. Восстановить систему вы сможете в течение максимум двух-трех часов, а вот данные пользователей уже не восстановишь...

На втором месте — это файлы настройки системы, находящиеся в каталоге /etc. Архивирование этих данных позволит существенно сэкономить время, которое вам потребуется на восстановление системы после сбоя.

И, наконец, на третьем месте — это дистрибутивы программ, которые не входят в состав дистрибутива Linux. Эти данные, как правило, не нуждаются в частом обновлении.

Конечно, можно создать полную копию всей корневой файловой системы, но в результате вы получите один большой архив, на обновление которого будет затрачена уйма времени.

Когда будет происходить создание резервных копий?

Самое удачное время для этого мероприятия — ночь. Почему именно ночь?

1. Систему можно настроить на автоматическое обновление архива.
2. Операция архивирования, как правило, не требует вмешательства оператора.
3. Вряд ли пользователи будут довольны дополнительной нагрузкой на систему в рабочее время.
4. Существует вероятность того, что в конце дня данные на жестком диске изменятся, а так как копия создавалась днем или утром, то новые данные не попадут в архив.

Еще следует учитывать, что если вы используете стример с интерфейсом FDC, средняя скорость архивирования которого составляет 4 Мб/мин, то для архивирования 1 Гб вам потребуется около 4 часов. Для использования сжатия данных потребуется дополнительное время. Также нужно помнить о необходимости смены кассет с лентой, но при использовании стримера с интерфейсом FDC и лентой на 1 Гб об этой проблеме можно забыть: при использовании сжатия вы сможете поместить на ленту около 2 Гб, а весь процесс займет около 6 часов. А в случае со стримером SCSI процесс архивирования займет не более часа и его можно выполнить в конце рабочего дня.

Кто этим будет заниматься?

В случае, если процессу архивации подлежит ваш домашний компьютер, то этой ответственной задачей будете заниматься вы сами. На предприятии (особенно большом) необходимо определить, кто будет архивировать данные с каждого сервера сети: не будете же вы бегать по зданию со стримером, контролируя процесс создания резервных копий? В идеале, за каждым сервером должен быть закреплен человек, ответственный за процесс создания архива и поддержанию его в должном состоянии.

Как часто будет производиться архивирование?

Для ответа на этот вопрос я предлагаю к вашему рассмотрению шестидневную схему архивации. Для этого вам потребуется шесть кассет (или шесть магнитооптических дисков, но помните о максимальной емкости магнитооптического диска). На этих кассетах сделайте надписи: Пт1, Пт2, Пн., Вт., Ср., Чт. Начните создание копий в пятницу вечером и придерживайтесь расписания, приведенного в табл. 4.14.

Расписание резервного копирования • *Таблица 4.14*

День	Кассета	Операция
Пятница	Пт1	Создание резервной копии всего диска
Понедельник	Пн.	Создание копии новых и обновленных данных
Вторник	Вт.	Создание копии новых и обновленных данных
Среда	Ср.	Создание копии новых и обновленных данных
Четверг	Чт.	Создание копии новых и обновленных данных
Пятница	Пт2	Создание резервной копии всего диска

4.17. Использование программы cpio

Для создания архивов на магнитной ленте или жестком диске, а также для извлечения файлов из архивов используется программа **cpio.**

Программа **cpio** может работать в трех оперативных режимах. При этом режим работы задается указанной опцией. Помимо опций могут использоваться параметры, которые управляют работой **cpio** в заданном режиме. Формат вызова программы cpio зависит от режима, в котором она должна работать. Опции программы и соответствующий им формат вызова представлены в табл. 4.15, а параметры — в табл. 4.16.

Опции программ cpio *Таблица 4.15*

Опция и формат вызова	Описание
-o Формат вызова: cpio -o[параметры] список_файлов [имя_архива]	Копирование в архив всех файлов, которые указаны. Файлы в списке указываются по одному в строке
-i Формат вызова: cpio -i[параметры] [шаблоны] имя_архива	В режиме, задаваемом этой опцией, программа cpio будет извлекать файлы из архива. При этом извлекаться будут только те файлы, чьи имена совпадут с одним из указанных шаблонов. Если ни одного шаблона не указано, то из архива будут извлекаться все файлы. Шаблон также может включать в себя символы подстановки
-P Формат вызова: cpio -p[параметры] каталог	Копирование будет производиться в указанный каталог

Параметры программы cpio *Таблица 4.16*

Параметр	Описание
-0	Этот параметр позволяет включить в архив файлы, имена которых содержат символ новой строки. Используется вместе с опциями -o и -p
-a	Устанавливает текущее время в качестве времени последнего доступа к файлу
-A	Присоединение файлов к существующему архиву на диске
-b	Заменяет местами байты и полуслова
-B	Устанавливает размер блока в 5120 байт. По умолчанию используется 512 байт
-c	Чтение и запись заголовка как текста ASCII
-C n	Устанавливает размер блока в n байтов
-d	Предварительно создает каталоги в случае необходимости
-E файл	Используется вместе с опцией -i. При этом в качестве шаблонов, по которым отбираются файлы для извлечения, будут использоваться строки, содержащиеся в указанном файле. Таким образом, указываемый файл есть файл списка шаблонов
-F файл	Использует указанный файл в качестве архива.
-H тип	Устанавливает тип формата архива: bin — устаревший двоичный формат; crc — формат Unix SysV Release 4, использующий подсчет контрольных сумм (CRC); hpodc формат Hewlett-Packard; newc — формат Unix SysV Release 4, может использоваться для файловых систем, количество дескрипторов в которых превышает 65536; odc — формат POSIX.1; tar формат tar; ustar — формат POSIX.1 tar
-I файл	Указанный файл будет использоваться в качестве исходного архива. Используется вместе с опциями -i и -p
-l	Создает ссылки вместо копирования файлов в каталог при использовании опции -p
-L	При встрече символической ссылки копироваться будет не сама ссылка, а файл, на который она указывает. Используется совместно с опциями -o и -p
-m	Сохраняет время последнего изменения файла
-M строка	Выводит указанную строку при смене носителя
-n	Выводит GID (идентификатор группы)
-O файл	Вывод в указанный файл
-r	Переименование файлов, новые имена будут запрошены у пользователя
-s	Замена байтов местами. Используется с опцией -i
-S	Замена полуслов местами. Используется с опцией -i
-t	Режим тестирования. Выводит содержимое архива, который должен быть создан, но сам архив при этом не создается
-u	Перезапись файлов без подтверждения
-v	Вывод имени всех файлов
-V	Выводит точку вместо имени файла

Давайте рассмотрим несколько примеров использования программы **cpio.** Создать архив можно с помощью опции -o программы **cpio.** Программа **cpio** будет читать имена файлов, которые следует поместить в архив со стандартного ввода. По умолчанию используется бинарный формат архива, поэтому формат архива нужно задать опцией -H. Будем использовать формат **tar.** Введите команду:

```
cpio -o -H tar -O arc.tar
```

Затем введите имена файлов, которые вы хотите добавить в архив, например:

```
/opt/ctrl/ctrl.c
/opt/ctrl/ctrl.html
```

По окончании ввода будет создан архивный файл arc.tar. Программа **cpio** создаст архив с сохранением структуры каталогов. Чтобы убедиться в этом, запустите файловый менеджер tc и просмотрите содержимое архива arc.tar.

Конечно, вводить имена файлов вручную не совсем приятное занятие. Для автоматизации ввода можно использовать средства перенаправления ввода/вывода. Например, для архивирования текущего каталога введите команду:

```
ls | cpio -o -H tar -O current_dir.tar
```

Для извлечения файлов из архива введите команду:

```
cpio -i -H tar < current_dir.tar
```

В режиме извлечения файлов программа cpio читает со стандартного ввода имя архива.

4.18. Повышение производительности жесткого диска

Существенно повысить производительность жесткого диска поможет программа **hdparm**. Я увеличил скорость операции чтения своего жесткого диска Quantum Fireball ATA66 с 3,75 Мб/с до 14 Мб/с, а жесткий диск IBM ATA100 (модель точно не помню) удалось «разогнать» до 30,1 Мб/с!

Рассмотрим использование программы hdparni на примере. Для начала запустим ее в режиме теста, зарегистрировавшись в системе как root:

```
t hdparm -t /dev/hda
Timing buffered disk reads: 64 MB in 17.08 seconds = 3.75 MB/sec
```

Взглянув на отображенную информацию, можно заметить: «Маловато, однако». Чтобы понять, почему так получается, введем команду:

```
# hdparm /dev/hda
```

и получим в ответ

```
/dev/hda:
multcount = 0 (off)
I/O support = 0 (default 16-bit)
unmaskirq = 0 (off)
using_dma = 0 (off)
keepsettings = 0 (off)
nowerr = 0 (off)
readonly = 0 (off)
readahead = 8 (on)
```

Из этого можно сделать вывод, что все параметры выключены и используется шестнадцатиразрядный доступ к диску. Давайте попробуем немного «разогнать» наш жесткий диск.

```
# hdparm - dlm2c3ul /dev/hda
```

Теперь разберемся, что же мы сделали этой командой. Во-первых, мы включили DMA, затем разрешили передавать более одного слова за такт, а также включили тридцатидвухбитный доступ к диску (команда с). Кстати, параметр **ul** полезен и в тех случаях, когда у вас начинает «заикаться» **xmms** во время прослушивания музыки.

Вот теперь опять запустим **hdparm** в режиме теста. В зависимости от жесткого диска у нас должно получиться не менее 14 Мб/с. Думаю, по сравнению с предыдущим показателем разница существенна.

Можно использовать параметры X33 и X66 для включения режимов передачи данных UDMA33 и UDMA66 соответственно. Если при использовании режимов X33 и X66 производительность снизилась, используйте режим X68. Для сохранения параметров контроллера IDE используйте команду:

```
# hdparm -k 1 /dev/hda
```

При перезагрузке системы параметры IDE теряются, поэтому команду «разгона» винчестера нужно поместить в сценарий запуска системы. Сценарии загрузки рассматриваются в следующей главе. Сейчас просто добавьте команду вызова **hdparm** в файл `/etc/rc.d/rc.local`. Этот способ является наиболее универсальным, поскольку он позволяет установить отдельные параметры для разных жестких дисков, если у вас их несколько. Второй, менее универсальный, способ заключается в редактировании файла `/etc/sysconfig /harddisks`, в котором можно задать общие параметры для всех жестких дисков.

Есть еще один «подводный камень», который состоит в следующем: при пробуждении системы в нормальное состояние после «сна» параметры контроллера также сбрасываются. Этого можно избежать, если подправить файл конфигурации демона **apmd**, который отвечает за управление питанием. Параметры контроллера IDE, которые устанавливаются при переходе системы в «спящий» режим и выходе из него, задаются строками **HDPARM_AT_SUSPEND и HDPARM_AT_RESUME** в файле конфигурации `/etc/sysconfig/apmd`.

Файлы конфигурации, расположенные в каталоге `/etc/sysconfig`, имеются только в системах, подобных Red Hat — это Red Hat Linux, Mandrake Linux, SuSE Linux, ASP Linux, Back Cat Linux, ABI Linux и другие.

С помощью команды **hdparm** можно не только повысить скорость обмена данными, но, как вы заметили, и снизить ее. Особенно это полезно при прослушивании аудиокомпакт-дисков. В самом деле, зачем прослушивать аудиокомпакты на приводе 52x? К тому же высокоскоростной CDROM слушком шумит. «Притормозить» привод можно такой командой:

```
# hdparm -E 2 /dev/hdd
```

В данном примере мы устанавливаем вторую скорость, то есть 300 Кб/с.

4.19. Создание массивов RAID

Идея надежности хранения данных волновала, волнует и будет волновать не одно поколение системных администраторов и пользователей. Используемые в ОС Linux файловые системы **ext2 и ext3** обладают достаточной степенью надежности, но зачастую этого мало.

Если существует вероятность потерять данные в результате выхода из строя жесткого диска, то единственным выходом из данной ситуации является использование массивов жестких дисков RAID. **RAID** (Redundant Array of Independent Disk или Redundant Array of Expensive Disk) — матрица независимых дисков с избыточностью. Под избыточностью подразумевается резерви-

рование и дублирование данных. В зависимости от уровня RAID, предоставляются различные способы объединения дисков в массив (см. табл. 4.17).

Наиболее часто используются массивы уровней 0,1 и 5. Иногда встречаются комбинированные способы объединения данных в массив, например, 5+1.

Уровни RAID *Таблица 4.17*

Уровень RAID	Описание
0	Обеспечивает распределение блоков данных по нескольким дискам. Предназначен для хранения больших объемов данных, не умещающихся на одном диске. Этот уровень не обеспечивает избыточности, при использовании этого массива диски просто объединяются в цепочку. Емкость массива равна суммарной емкости всех дисков, образующих массив
1	Обеспечивает технологию зеркального копирования. Диски дублируют друг друга. Емкость массива равна емкости самого меньшего из дисков
2	Запись на разные диски производится методом битового чередования малых блоков данных с добавлением кодов исправления ошибок
3	То же, что и уровень RAID 2, но контрольные коды записываются на отдельный диск
4	Представляет собой совокупность взаимосвязанных данных, которые записываются на один диск, а контрольные коды — на другой
5	На этом уровне используются контрольные суммы и данные записываются «вперемешку» на все диски. При выходе из строя одного из дисков потерянные данные восстанавливаются с помощью контрольной суммы. Общая емкость массива вычисляется по формуле min_size*(n-1), где min_size — объем наименьшего из дисков, а n — количество дисков в массиве. Минимальное количество дисков равно трем

Организация массива RAID доступна не каждому из-за все еще высокой стоимости на контроллеры RAID. Хотя производители материнских плат пытаются поправить это, выпуская материнские платы со встроенными контроллерами RAID, но такие контроллеры довольно неуниверсальны и обладают слабыми возможностями.

ОС Linux поддерживает программные контроллеры RAID. Применение программных контроллеров имеет как свои преимущества, так и недостатки. К достоинствам относится возможность использования дисков с различными интерфейсами, например, SCSI и IDE, для организации массива — программному контроллеру все равно, с чем работать. Недостатком является дополнительная нагрузка на центральный процессор — он выполняет всю работу по обеспечению функционирования массива RAID.

Итак, приступим к созданию массива RAID. Вам потребуется любой дистрибутив с поддержкой программного контроллера RAID (Software RAID). Такой возможностью обладают практически все современные дистрибутивы. Для включения поддержки RAID вам нужно перекомпилировать ядро. Если ваше ядро поддерживает RAID, при загрузке системы вы должны увидеть примерно следующее:

```
md driver 0.90.0 MAX_MD_DEVS=256, MAX_REAL=12
raid5: measuring checksumming speed
raid5: MMX detected, trying high-speed MMX checksum routines
    pII_mmx :   980.694 MB/sec
    p5_mmx :    999.744 MB/sec
    8regs :     753.237 MB/sec
    32regs :    444.246 MB/sec
```

```
using fastest function: p5_mmx (999.744 MB/sec)
md.c: sizeof(mdp_super_t) = 4096
Partition check:
hda: hda1 hda2 < hda5 hda6 hda7 hda8 >
autodetecting RAID arrays
autorun ...
... autorun DONE.
```

Если перезагружать систему вам не хочется, проверить поддержку RAID вы можете с помощью команды:

```
dmesg | less
```

Программа dmesg выводит на стандартный вывод сообщения ядра во время загрузки системы.

Включить поддержку RAID можно в разделе Block device конфигуратора ядра (make menuconfig). Данная опция называется RAID n support, где n — это номер уровня массива RAID. После этого нужно установить пакет raidtools, в состав которого входят программы raidhotadd, raidhotremove, mkraid и другие.

Для организации массива уровня RAID 1 нужно выделить два раздела и изменить тип этих разделов на Linux raid autodetect. Обратите внимание, я написал «два раздела», а не «два диска», так как конфигурируется программный контроллер. Конечно, лучше, чтобы эти разделы располагались на разных дисках, в противном случае от нашего массива будет мало толку.

Теперь отредактируйте файл /etc/raidtab (см. листинг 4.3).

Листинг 4.3. Файл /etc/raidtab (уровень 1)

```
# Имя устройства RAID
raiddev /dev/md0
# Уровень
raid-level 1
chunk-size 8
persistent-superblock 1
# Число дисков в массиве
nr-raid-disk 2
# Число дисков, которые будут использоваться в качестве замены,
если
# один из дисков выйдет из строя
nr-spare-disk 0
# Определяем первый диск RAID
device /dev/hdb1
raid-disk 0
# Определяем второй диск RAID
device /dev/hdc1
raid-disk 1
```

После этого нужно создать устройство /dev/md0, для чего выполните следующую команду:

```
mkraid /dev/md0
```

В некоторых случаях нужно будет использовать дополнительные параметры, о которых вы можете прочитать в справочной системе (man **mkraid**). В случае, если инициализация прошла успешно, в файле /proc/mdstat вы увидите примерно следующее:

```
Personalities: [raid1]
read_ahead 1024 sectors
mdO: active raidl hdc1[1] hdb1[0]
```

Теперь рассмотрим, как создать массив уровня RAID 5. Для этого используйте конфигурационный файл, текст которого приведен в листинге 4.4.

Листинг 4.4. Файл /etc/raidtab (уровень 5)

```
raiddev /dev/md0
raid-level 5
nr-raid-disk 3
nr-spare-disk 0
persistent-superblock  1
parity-algorithm left-symmetric
chunk-size 64
device /dev/hdb1
raid-disk 0
device /dev/hdc1
raid-disk 1
device /dev/hdd1
raid-disk 2
```

После успешной инициализации вы можете использовать массив как один самый обыкновенный диск, то есть создавать и удалять разделы, монтировать эти разделы к корневой файловой системе.

Для извлечения диска из массива используется команда **raidhotremove**. Извлечение может понадобиться, если один из дисков вышел из строя. В этом случае я рекомендую использовать диски с возможностью «горячей» замены. В противном случае вам придется останавливать машину для замены диска. После замены на новом диске следует создать разделы так же как и на диске, который вышел из строя, и только после этого выполнить команду **raidhotadd**. В качестве параметров программы **raidhotremove и raidhotadd** используют имя массива (/dev/md0) и номер диска, извлекаемого из массива.

4.20. Форматирование дискет в Linux

В других книгах, посвященных ОС Linux, этой теме обычно уделяется мало внимания. Хотя эта тема никак не относится к организации сервера, я решил все-таки рассмотреть процесс форматирования дискет более подробно, потому что в ближайшее время они еще будут использоваться.

Я использую программу **kfloppy**, которая входит в состав KDE и в особых комментариях не нуждается. В качестве альтернативы вы можете использовать программы **fdformat** и **superformat**. Первая из них (**fdformat**) форматирует

дискеты только в Linux-формате (**ext2fs**). Вызов программы осуществляется следующим образом:

```
fdformat [-n] device
```

Опция -n запрещает проверку дискеты при форматировании.

device - это или /dev/fdO (**A:**) или /dev/fd1 (**B:**).

Более гибкой является программа **superformat**. Она может форматировать дискету как в Linux-формате, так и создавать файловую систему DOS. На самом деле она вызывает **mformat** из **mtools** для создания файловой системы **msdos**. Параметры программы **superformat** указаны в табл. 4.18. Формат использования программы **superformat** следующий:

superformat параметры

Параметры программы superformat *Таблица 4.18*

Параметр	Описание
-2	Форматирование дисков большой емкости для работы с программой 2mf
-B	Проверка диска с помощью программы mbadblocks
-d устройство	Форматирование диска в указанном устройстве. По умолчанию используется /dev/fdO
--dd	Форматирование дисков двойной плотности (Double Density)
-D устройство	Указание устройства в формате DOS для передачи программе mformat (a: или b:)
-f	Запрет проверки диска
-H n	Установка количества головок (по умолчанию 2)
--hd	Форматирование дисков высокой плотности (High Density)
-1	Не использовать 2т
--no2m	Не использовать 2т
-s n	Установка количества секторов. Аргумент n обозначает не количество физических секторов, а количество логических 512-байтных секторов
-t n	Установка количества дорожек. Значение по умолчанию — 40 или 80 в зависимости от устройства и плотности диска
-v n	Установка уровня отладки. Допустимые значения 1, 2, 3, 6 и 9
-V	Проверка диска после завершения форматирования всего диска. По умолчанию после форматирования каждой дорожки производится ее проверка

С помощью этой программы можно увеличить емкость дискет, используя нестандартные форматы (см. табл. 4.19). Однако за качество работы этих дискет я не ручаюсь. К тому же я очень не рекомендую использовать дискеты нестандартных форматов в качестве загрузочных.

Нестандартные форматы дискет *Таблица 4.19*

Размер дискеты	Емкость устройства	Стандартная емкость дискеты	Число дорожек	Число секторов	Емкость дискеты, байт
5.25"	360 Кб	360 Кб	41	10	409.088
5.25"	1.2 Мб	360 Кб	81	10	816.640
5.25"	1.2 Мб	1.2Мб	81	18	1.476.096 (1.45 Мб)
3.5"	720 Кб	720 Кб	81	10	816.640
3.5"	1.44 Мб	720 Кб	81	10	816.640
3.5"	1.44 Мб	1.44 Мб	81	21	1.723.904 (1.7 Мб)

Пример:

```
superformat -d /dev/fdO -t 81 -s 21
```

Если дискета работает крайне нестабильно, попробуйте уменьшить число секторов до 20.

5

Процессы

5.1. Системные вызовы fork() и exec()

Процесс в Linux (как и в UNIX) — это программа, которая выполняется в отдельном виртуальном адресном пространстве. Когда пользователь регистрируется в системе, под него автоматически создается процесс, в котором выполняется оболочка (shell), например, /bin/bash.

В Linux поддерживается классическая схема мультипрограммирования. При этом Linux поддерживает параллельное (или квазипараллельное при наличии только одного процессора) выполнение процессов пользователя. Каждый процесс выполняется в собственном виртуальном адресном пространстве, т.е. процессы защищены друг от друга и крах одного процесса никак не повлияет на другие выполняющиеся процессы и на всю систему в целом. Один процесс не может прочитать что-либо из памяти другого процесса (или записать в нее) без «разрешения» на то другого процесса. Санкционированные взаимодействия между процессами допускаются системой.

Ядро предоставляет системные вызовы для создания новых процессов и для управления порожденными процессами. Любая программа может начать выполняться, только если другой процесс ее запустит или произойдет какое-то прерывание (например, прерывание внешнего устройства).

В связи с развитием **SMP** (Symmetric Multiprocessor Architectures) в ядро Linux был внедрен механизм **нитей или потоков** управления (threads). **Нитями** также называют «легковесные» процессы. Другими словами, **нить** -- это процесс, выполняемый в виртуальной памяти, которая используется вместе с другими нитями одного и того же «тяжеловесного» процесса. Такой «тяжеловесный процесс» обладает отдельной виртуальной памятью и может иметь несколько «легковесных» процессов.

Потоки (или **нити)** позволяют решать в рамках одной программы одновременно несколько задач.

Операционная система предоставляет программе некоторый интервал процессорного времени. Когда программа переходит в режим ожидания какого-либо события (например, сигнала) или освобождает процессор, операционная система передает управление другой программе. Распределяя время центрального процессора, операционная система распределяет его не между программами, а между **потоками.** Исходя из всего этого, **потоки — это**

наборы команд, имеющие возможность получать время процессора. Время процессора выделяется **квантами. Квант** — это минимальное время, на протяжении которого **поток (нить)** может использовать процессор.

Когда вы вводите команду, интерпретатор производит поиск указанной программы в каталогах, которые перечислены при определении переменной окружения PATH. При этом будет выполнена первая найденная программа с указанным именем.

Если интерпретатору (shell) встречается команда, соответствующая выполняемому файлу, интерпретатор выполняет ее, начиная с точки входа **(entry point).** Для С-программ **entry point** — это функция **main.** Точка входа для каждой среды разработки различна. Запущенная программа тоже может создать процесс, т.е. запустить какую-то программу и ее выполнение тоже начнется с функции **main.** Затем с помощью системного вызова **fork()** создается адресное пространство — подготавливается «место» для нового процесса, а потом с помощью вызова **exec()** в это адресное пространство загружается программа. Таким образом, каждый новый процесс выполняется в своей индивидуальной среде.

Для создания процессов используется системный вызов: **fork()**. Вызов **fork()** создает новое адресное пространство, которое полностью идентично адресному пространству основного процесса. Другими словами, вызов **fork()** создает новый процесс. После выполнения этого системного вызова вы получаете два абсолютно одинаковых процесса — основной и порожденный. Функция **fork()** возвращает 0 в порожденном процессе и PID (Process ID — идентификатор порожденного процесса) — в основном. PID — это целое число.

Теперь, когда процесс создан, можно запустить в нем программу с помощью вызова **exec.** Параметрами функции **exec** являются имя выполняемого файла и, если нужно, параметры, которые будут переданы этой программе. В адресное пространство порожденного с помощью **fork()** процесса будет загружена новая программа и ее выполнение начнется с точки входа (адрес функции **main).**

В качестве примера рассмотрим следующий фрагмент программы:

```
if (fork()==0) wait(0);
else execl("ls", "ls", 0); /* порожденный процесс */
```

Теперь рассмотрим более подробно, что же делается при выполнении вызова **fork():**

1. Выделяется память для описателя нового процесса в таблице процессов.
2. Назначается идентификатор процесса PID.
3. Создается логическая копия процесса, который выполняет **fork()** — полное копирование содержимого виртуальной памяти родительского процесса, копирование составляющих ядерного статического и динамического контекстов процесса-предка.
4. Увеличиваются счетчики открытия файлов (порожденный процесс наследует все открытые файлы родительского процесса).
5. Возвращается PID в точку возврата из системного вызова в родительском процессе и 0 — в процессе-потомке.

5.1.1. Общая схема управления процессами

Каждый процесс может порождать полностью идентичный процесс с помощью **fork()**. Родительский процесс может дожидаться окончания выполнения всех своих процессов-потомков с помощью системного вызова **wait.** В любой момент времени процесс может изменить содержимое своего образа памяти, используя одну из разновидностей вызова **exec().** Каждый процесс реагирует на сигналы и, естественно, может установить собственную реакцию на сигналы, производимые операционной системой. Приоритет процесса может быть изменен с помощью системного вызова **nice.**

Сигнал — это способ информирования процесса ядром о происшествии какого-то события. Если возникает несколько однотипных событий, процессу будет подан только один сигнал. Сигнал означает, что произошло событие, но ядро не сообщает, сколько таких событий произошло.

Примеры сигналов:

1. Окончание порожденного процесса (например, из-за системного вызова **exit** (см. ниже)).
2. Возникновение исключительной ситуации.
3. Сигналы, поступающие от пользователя, при нажатии определенных клавиш.

Установить реакцию на поступление сигнала можно с помощью системного вызова **signal:**

```
func = signal(snum, function);
```

где: snum..... номер сигнала;

function..... адрес функции, которая должна быть выполнена при поступлении указанного сигнала.

Возвращаемое значение — адрес функции, которая будет реагировать на поступление сигнала. Вместо **function** можно указать ноль или единицу. Если был указан ноль, то при поступлении сигнала **snum** выполнение процесса будет прервано аналогично вызову **exit.** Если указать единицу, данный сигнал будет проигнорирован, но это возможно не для всех процессов.

С помощью системного вызова **kill** можно сгенерировать сигналы и передать их другим процессам. Обычно **kill** используется для того, чтобы принудительно завершить («убить») процесс:

```
kill(pid, snum);
```

где: pid..... идентификатор процесса;

snum.... номер сигнала, который будет передан процессу (см. табл. 5.1).

Pid состоит из идентификатора группы процессов и идентификатора процесса в группе. Если вместо **pid** указать нуль, то сигнал **snum** будет направлен всем процессам, относящимся к данной группе (понятие группы процессов аналогично группе пользователей). В одну группу включаются процессы, имеющие общего предка. Идентификатор группы процесса можно изменить с помощью системного вызова **setpgrp**. Если вместо **pid** указать -1, то ядро передаст сигнал всем процессам, идентификатор пользователя которых равен идентификатору текущего выполнения процесса, посылающего сигнал. Номера сигналов приведены в табл. 5.1. Сигналы (точнее, их номера) описаны в файле singnal.h.

Номера сигналов *Таблица 5.1*

Номер	Название	Описание
01	SIGHUP	Освобождение линии (hangup)
02	SIGINT	Прерывание (interrupt)
03	SIGQUIT	Выход (quit)
04	SIGILL	Некорректная команда (illegal instruction). Не переустанавливается при перехвате
05	SIGTRAP	Трассировочное прерывание (trace trap). Не переустанавливается при перехвате
06	SIGIOT или SIGABRT	Машинная команда ЮТ. Останов ввода/вывода
07	SIGBUS	Ошибка на шине
08	SIGFPE	Исключительная ситуация при выполнении операции с вещественными числами (floating-point exception)
09	SIGKILL	Уничтожение процесса (kill). Не перехватывается и не игнорируется
10	SIGUSR1	Определяемый пользователем сигнал 1
11	SIGSEGV	Некорректное обращение к сегменту памяти (segmentation violation)
12	SIGUSR2	Определяемый пользователем сигнал 2
13	SIGPIPE	Запись в канал, из которого некому читать. Обрыв потока
14	SIGALRM	Будильник
15	SIGTERM	Программный сигнал завершения
16	SIGSTKFLT	Сбой стека
17	SIGCHLD (или SIGCLD)	Изменение статуса дочернего процесса
18	SIGCONT	Продолжение работы после сигнала STOP. Не перехватывается и не игнорируется
19	SIGSTOP	Сигнал СТОП. Не перехватывается и не игнорируется
20	SIGTSTP	Сигнал останова клавиатуры
21	SIGTTIN	Фоновое чтение из терминала (tty)
22	SIGTTOU	Фоновая запись на терминал (tty)
23	SIGURG	Критическое состояние сокета
24	SIGXCPU	Превышенный предел процессорного времени
25	SIGXFSZ	Превышенный предел размера файла
26	SIGVTALRM	Сигнал виртуального будильника
27	SIGPROF	Сигнал профилирующего будильника
28	SIGWINCH	Изменение размера окна
29	SIGIO	Разрешение ввода/вывода
30	SIGPWR	Сбой питания
31	SIGSYS	Некорректный параметр системного вызова

Для нормального завершения процесса используется вызов:

```
exit(status)
```

где **status** — это целое число, возвращаемое процессу-предку для его информирования о причинах завершения процесса-потомка.

Вызов **exit** может задаваться в любой точке программы, но может быть и неявным, например, при выходе из функции **main** (при программировании на С) оператор **return 0** будет воспринят как системный вызов **exit(0)**.

5.2. Перенаправление ввода/вывода

Практически все операционные системы обладают механизмом перенаправления ввода/вывода, и Linux не является исключением из этого правила. Обычно программы вводят текстовые данные с консоли (терминала) и

выводят данные на консоль. При вводе под консолью подразумевается клавиатура, а при выводе — экран монитора. Клавиатура и экран монитора — это, соответственно, стандартный ввод и вывод **(stdin** и **stdout).** Любой ввод/вывод можно интерпретировать как ввод из некоторого файла и вывод в файл. Работа с файлами производится через их *дескрипторы.* Для организации ввода/вывода в UNIX используются три файла: **stdin** (дескриптор 0), **stdout** (дескриптор 1) и **stderr** (дескриптор 2).

Символ > («больше») используется для перенаправления стандартного вывода в файл. Например:

```
$ cat > newfile.txt
```

В этом примере стандартный вывод команды **cat** будет перенаправлен в файл `newfile.txt`, который будет создан после выполнения этой команды. Если файл с этим именем уже существует, то он будет перезаписан. Нажатие Ctrl + D остановит перенаправление и прервет выполнение команды **cat.**

Символ < («меньше») используется для переназначения стандартного ввода команды. Например, при выполнении команды `cat < file.txt` в качестве стандартного ввода будет использован файл `file.txt`, а не клавиатура.

Символ >> используется для присоединения данных в конец файла *(append)* стандартного вывода команды. Например, в отличие от случая с символом >, выполнение команды **cat >> newfile.txt** не перезапишет файл в случае его существования, а добавит данные в его конец.

Чтобы перенаправить весь стандартный поток ошибок в какой-нибудь файл, используйте переадресацию **2> имя_файла** или **2» имя_файла.** В первом случае стандартный поток ошибок будет передан в файл или на устройство, а во втором — поток ошибок будет добавлен в файл, если такой существует. При использовании переадресации **2>&1** стандартный поток ошибок будет перенаправлен на стандартный вывод интерпретатора **Bourne** (здесь 1 и 2 — дескрипторы файлов). Для перенаправления стандартного потока ошибок в файл вы можете также использовать переадресацию **>& имя_файла** (интерпретатор **C-Shell).**

В командных интерпретаторах **Korn и C-Shell** можно использовать переадресацию **>! имя_файла.** При этом файл не будет перезаписан, если он существует.

Символ | используется для перенаправления стандартного вывода одной программы на стандартный ввод другой. Например, `ps -ax | grep httpd`.

Можно также использовать переадресацию |&. В этом случае стандартный поток ошибок будет передан по каналу другой команде интерпретатора.

5.3. Команды управления процессами

Команда ps

Команда **ps** предназначена для вывода информации о выполняемых в текущий момент процессах. Данная команда имеет много параметров, о которых вы можете прочитать в руководстве **(man ps).** Здесь я опишу лишь наиболее часто используемые мною (см. табл. 5.2).

Параметры программы ps *Таблица 5.2*

Параметр	Описание
-a	Отобразить все процессы, связанные с терминалом (отображаются процессы всех пользователей)
-e	Отобразить все процессы
-t список_терминалов	Отобразить процессы, связанные с указанными терминалами
-и идентификаторы_пользователей	Отобразить процессы, связанные с данными идентификаторами
-д идентификаторы_групп	Отобразить процессы, связанные с данными идентификаторами групп
-x	Отобразить все процессы, не связанные с терминалом

Например, после ввода команды ps -a вы увидите примерно следующее:

```
PID TTY      TIME CMD
1007 tty1     00:00:00 bash
1036 tty2     00:00:00bash
1424 ttyl     00:00:02 mc
1447 pts/0    00:00:02 mpg123
2309 tty2     00:00:00 ps
```

Для вывода информации о конкретном процессе вы можете воспользоваться командой:

```
# ps -ax | grep httpd
698 7    S  0:01 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1261 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1262 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1263 ?   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1264 ?   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1268 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1269 ?   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1270 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1271 ?   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1272 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1273 7   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
1280 ?   S  0:00 httpd -DHAVE_PHP4 -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_A
```

В приведенном выше примере используется перенаправление ввода/вывода между программами **ps** и **grep,** в результате чего будет отображена информация обо всех процессах, содержащих в строке запуска **«httpd».** Данную команду **(ps -ax | grep httpd)** я написал только лишь в демонстрационных целях -- гораздо проще использовать параметр -C программы **ps** вместо перенаправления ввода/вывода и параметр -e вместо **-ax.**

Команда top

Эта команда предназначена для вывода информации о процессах в реальном времени. Процессы сортируются по максимальному занимаемому процессорному времени, но вы можете изменить порядок сортировки (см. **man top).** Команда также сообщает о свободных системных ресурсах.

```
# top
7:49pm  up 5 min,  2 users,  load average: 0.03, 0.20, 0.11
56 processes: 55 sleeping, 1 running, 0 zombie, 0 stopped
CPU states:  7.6% user,  9.8% system,  0.0% nice, 82.5% idle
Mem  130660K av,  94652K used, 36008K free, OK shrd,  5220K buff
Swap:72256K av,   OK used,     72256K free     60704K cached
PID USER   PRI  NI SIZE  RSS   SHARE   STAT  %CPU  %MEM  TIME   COMMAND
1067 root  14   0  892   892   680     R     2.8   0.6   0:00   top
   1 root  0    0  468   468   404     S     0.0   0.3   0:06   init
   2 root  0    0  0     0     0       SW    0.0   0.0   0:00   kflushd
   3 root  0    0  0     0     0       SW    0.0   0.0   0:00   kupdate
   4 root  0    0  0     0     0       SW    0.0   0.0   0:00   kswapd
```

Просмотреть информацию об оперативной памяти вы можете с помощью команды **free,** а о дисковой — с помощью команды **df.** Информация о зарегистрированных в системе пользователях доступна по команде **w.**

Существует графический аналог программы **top** — **gtop** (см. рис. 5.1).

Изменение приоритета процесса — команда nice

Формат использования:

nice [-коэффициент понижения] команда [аргумент]

Команда **nice** выполняет указанную команду с пониженным приоритетом, коэффициент понижения указывается в диапазоне 1..19 (по умолчанию он равен 10). Суперпользователь может повышать приоритет команды, для этого нужно указать отрицательный коэффициент, например −10. Если указать коэффициент больше 19, то он будет рассматриваться как 19.

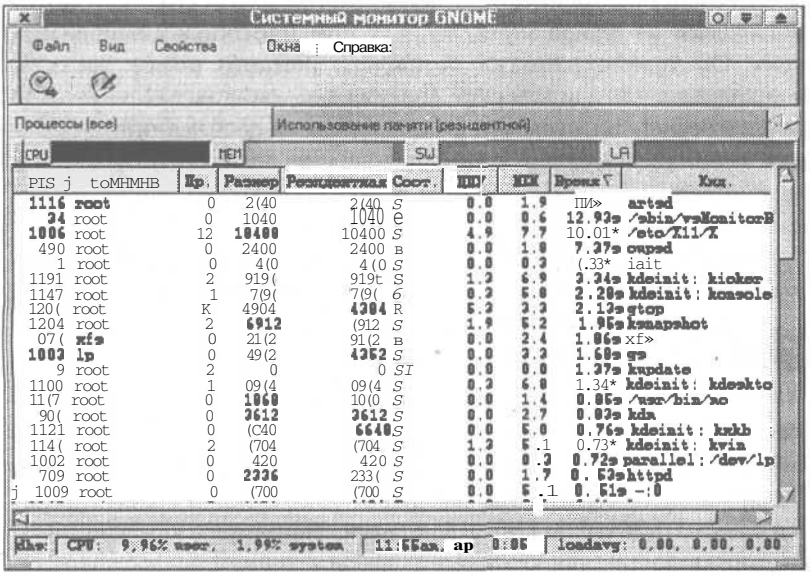


Рис. 5.1. *Программа gtop*

Команда nohup — игнорирование сигналов прерывания

Формат использования:

nohup команда [аргумент]

Команда nohup выполняет запуск команды в режиме игнорирования сигналов. Не игнорируются только сигналы SIGHUP и SIGQUIT.

Команда kill — принудительное завершение процесса

Формат использования:

kill [-номер сигнала] PID

где PID.....идентификатор процесса, который можно узнать с помощью команды ps.

Команда kill служит для принудительного завершения процесса. При этом процессу с указанным идентификатором (PID) посылается сигнал завершения. В качестве параметра можно указать номер сигнала, который следует отсылать. По умолчанию отсылается сигнал SIGTERM.

Команды выполнения процессов в фоновом режиме — jobs, fg, bg

Команда jobs выводит список процессов, которые выполняются в фоновом режиме, fg — переводит процесс в нормальные режим («на передний план» — foreground), а bg — в фоновый. Запустить программу в фоновом режиме можно с помощью конструкции **<команда> &**

5.4. *Создание расписаний*

Механизм расписаний Linux (UNIX) предоставляет удобные и мощные средства для обслуживания системы. Например, можно создать расписание резервирования данных в нерабочее время или обновления системы через Интернет.

Для запуска команд по расписанию используется демон **crop.** Он запускается автоматически из файла /etc/init.d при многопользовательском уровне запуска. Об уровнях запуска поговорим немного позже. Каждую минуту **cron** пробуждается и проверяет содержимое файлов crontab. Для какой-то определенности назовем эти файлы файлами расписаний.

Демон **crop** имеет свою буферную область (каталог /var/spool/cron/), в которой находятся файлы crontab — файлы расписаний. Имена файлов расписаний соответствуют именам пользователей из файла /etc/passwd. Если **crop** найдет файлы расписаний, он загрузит их в память. К этим файлам не должно быть прямого доступа, редактировать их можно с помощью программы **crontab.**

Когда демон **cron** выполняет команду, он посылает по почте сообщение владельцу файла crontab или пользователю, указанному в переменной MAILTO файла crontab.

Программа crontab

Программа **crontab** используется для редактирования файлов расписаний отдельных пользователей. Программа позволяет устанавливать, удалять, редактировать и просматривать файлы расписаний. Например, для установки файла расписаний используется команда:

crontab -u user file.cron

Если не использовать опцию -и, то будет установлен файл расписания для пользователя, запустившего программу.

Каждый пользователь может иметь файл расписания. Для того, чтобы использовать эту возможность, пользователь должен быть прописан в файле /var/spool/cron.allow, если такой существует. Программу **crontab** можно запускать с опциями, приведенными в табл. 5.3.

Примечание.

Последние версии демона crond используют файл /etc/cron.allow вместо файла /var/spool/cronallow и файл /etc/cron.deny вместо файла /var/spool/crondeny. В файле /etc/cron.allow содержится список разрешенных пользователей, в файле /etc/cron. deny — запрещенных.

Опции программы crontab *Таблица 5.3*

Опия	Описание
-l	Выводит текущий файл расписания
-r	Удаляет файл расписания
-e	вызывает редактор, указанный в переменной окружения $EDITOR, для редактирования файла расписания

Каждая строка файла расписания имеет такой формат:

время_выполнения действие

Время выполнения состоит из пяти полей. В первом поле задаются минуты (0...59), во втором — часы (0...23), в третьем — день месяца (1...31), в четвертом - - номер месяца (1...12), а в пятом день недели (0...6, О соответствует воскресенью). В любом из этих полей можно поставить звездочку, которая обозначает все возможные значения. Например, следующая запись означает, что архивирование каталога /etc будет производиться каждый день, кроме воскресенья, в семь часов утра:

```
О 7 * * 1-6 tar cfz /backup /etc
```

В системе используется системный файл расписания это /etc/crontab (см. листинг 5.1).

Листинг 5.1. Файл /etc/crontab

```
# Интерпретатор команд
SHELL=/bin/bash
# Путь для поиска команд
PATH=/sbin:/bin:/usr/sbin:/usr/bin
# Отчет о выполнении расписания будет отправлен
# пользователю root
MAILTO=root
# Домашний каталог
HOME=/
# run-parts
01 * * * * root run-parts  /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

Директива **run-part** означает, что будут выполнены все выполняемые файлы из указанного каталога.

Демон atd

Еще одним способом планирования задач является использование демона **atd.** Этот демон используется для отложенного выполнения заданий. Для постановки заданий в очередь используются команды **at** и **batch.** Чтобы добавить задание в очередь используйте команду:

at время дата

После этого введите все команды, которые хотите выполнить. Для окончания ввода нажмите Ctrl+D. Более подробно о формате задания времени и даты вы можете прочитать в справочной системе. Например, для выполнения команд в 13:00 введите команду:

at 1pm

Для просмотра очереди заданий, введите команду atq. В результате будут выведены задания для пользователя, запустившего команду. При запуске программы суперпользователем будет выведен список заданий для всех пользователей. Удалить задание вы можете командой **atrm.**

5.5. Уровни выполнения. Программа init

ОС Linux имеет шесть уровней выполнения, а также три уровня выполнения по требованию.

Программа **init** переключает систему в нужный режим работы (на нужный уровень выполнения), и ее имеет право использовать только пользователь root. Для переключения на уровень **n** достаточно ввести команду:

```
# init n
```

Иногда, при небольшом изменении файла /etc/inittab, нужно заново перечитать таблицу инициализации (**inittab**). Для этого воспользуйтесь командой:

```
init q
```

Помните, что вы в любой момент можете изменить файл /etc/inittab и перечитать его заново командой **init q.**

Для перехода на первый уровень выполнения можно также использовать параметр s программы **init:**

```
init s
```

Описание уровней приведено в табл. 5.4.

Уровни выполнения *Таблица 5.4*

Уровень	Описание
0	Останов системы
1	Административный (однопользовательский) режим. Обычно используется пользователем root для восстановления системы после сбоя
2	Многопользовательский режим, но без использования сети
3	Многопользовательский режим, допускается работа в сети
4	Не используется
5	Многопользовательский режим. Поддержка сети. Автоматический запуск системы X Window
6	Перезагрузка системы

Файл /etc/inittab описывает, какие процессы должны запускаться во время загрузки и на протяжении нормальной работы системы. Программа **init** переключает уровни выполнения системы. Корректными номерами уровней системы являются 0...6, а также A, B и C.

Каждая строка файла /etc/inittab должна быть записана в формате:

```
id:уровни_выполнения:действие:процесс
```

Поле «Id» (идентификатор)..... уникальная последовательность из четырех символов (в старых дистрибутивах длина имени идентификатора ограничена двумя символами).

Поле **«уровни_выполнения»**..... перечень уровней выполнения, для которых будет выполнено указанное действие.

Поле **«действие»**..... задает действие, которое будет выполнено.

Поле **«процесс»**..... определяет процесс, который будет выполнен.

В качестве значений поля «уровни_выполнения» могут быть указаны номера уровней выполнения без каких-либо разделителей. Например, значение данного поля 1235 означает, что указанное действие будет применено к уровням 1, 2, 3 и 5. В качестве дополнительных уровней, которые называются уровнями по требованию (ondemand), могут быть указаны уровни A, B и C.

В качестве действия может быть указано одно из действий, описанных в табл. 5.5

Действия над процессами, задаваемые в файле /etc/inittab *Таблица 5.5*

Действие	Описание
wait	Процесс будет запущен на одном из указанных уровней выполнения, и программа init будет ждать его завершения
respawn	Процесс будет перезапущен после его завершения
once	Процесс будет запущен всего один раз на одном из указанных уровней выполнения
boot	Процесс будет запущен во время загрузки системы. Поле «уровни_выполнения» будет проигнорировано
bootwait	То же, что и boot, но программа init будет ждать завершения процесса
off	Не выполняет никаких действий
ondemand	Процесс будет запущен в режиме по требованию, то есть он будет выполнен, когда будет вызван один из уровней по требованию (A, B, C)
initdefault	Определяет уровень выполнения по умолчанию. Если он не указан, при загрузке программа init попросит вас ввести уровень выполнения
sysinit	Процесс будет запущен во время загрузки, но перед выполнением процессов, которые запускаются с помощью действия boot или bootwait
powerwait	Процесс будет запущен, когда исчезнет напряжение в сети. Естественно, для корректной работы этой записи нужен источник бесперебойного питания, от которого система и получит уведомление об исчезновении напряжения. Программа init будет ждать завершения этого процесса
powerfail	То же, что и powerwait, но программа init не будет ждать завершения процесса
powerokwait	Процесс будет запущен сразу после того, как программа init получит сведения о том, что питание восстановлено
ctrlaltdel	С помощью этого действия можно установить реакцию системы на нажатие комбинации клавиш Ctrl+Alt+Del

Теперь рассмотрим листинг обычного файла /etc/inittab (см. листинг 5.2).

Листинг 5.2. Файл /etc/inittab

```
id:5:initdefault:
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Рассмотрим подробно приведенный пример. Самая первая строка определяет уровень выполнения по умолчанию. Очень рекомендую вам установить это значение. Во-первых, вводить уровень загрузки каждый раз при запуске системы не очень удобно. Во-вторых, если не установлен уровень выполнения по умолчанию, система не сможет запускаться автоматически: при загрузке она будет ждать ввода номера уровня выполнения. Для загрузки системы подойдут два уровня: или 3, или 5. Я рекомендую использовать третий уровень выполнения, использовать уровень выполнения 5 нужно только в том случае, если вы хотите, чтобы система X Window запускалась автоматически при старте системы.

Затем указываются программы, которые будут выполнены во время системной инициализации (sysinit). В нашем случае будет запушен сценарий загрузки системы `/etc/rc.d/rc.sysinit`.

Далее следуют описания действий для каждого из уровней выполнения. Например, для уровня выполнения номер 5 будет запущен сценарий `/etc/rc.d/rc` с параметром 5. Программа **init** будет ждать завершения процесса `/etc/rc.d/rc`.

Программа `/sbin/update` будет запускаться на каждом из уровней выполнения один раз.

Реакция на нажатие комбинаций клавиш Ctrl+Alt+Del устанавливается с помощью действия **ctrlaltdel**. Будет запущена программа **shutdown** для немедленной (now) перезагрузки **(-r)** системы. Задержка перед переключением на шестой уровень (перезагрузка) составит три секунды **(-t3)**.

Реакция на перебои с питанием устанавливается с помощью действий **powerfail и powerokwait**. В качестве реакции на событие **powerfail** будет запущена программа **shutdown** для останова системы (-h). Опция **-f** указывает, что при перезагрузке будет пропущена проверка файловой системы с помощью fsck. Задержка перед переключением на шестой уровень выполнения составит две минуты (+2). На всех активных терминалах пользователи увидят сообщение «Power Failure; System Shutting Down». Если в течение двух минут питание будет восстановлено, запустится программа **shutdown** для отмены останова системы **(-c)**. На терминалах пользователей будет отображено сообщение «Power Restored; Shutdown Cancelled».

Наглядным примером для действия **respawn** может послужить система X Window. Можете провести такой эксперимент: перейдите на уровень выполнения 5:

```
init 5
```

Если уровень 5 уже активен, этого делать, естественно, не нужно. Находясь в системе X Window, нажмите Ctrl+Alt+Backspace — эта комбинация клавиш используется для экстренного останова системы X Window. После останова система X Window будет перезапущена.

5.6. *Сценарии загрузки системы*

Все Red Hat-подобные системы, в отличие от BSD-подобных (Slackware), используют систему инициализации SysV, хотя и несколько переработанную.

Примечание.

Система инициализации *— это набор файлов, необходимых для запуска операционной системы. Обычно система инициализации представляет собой сценарии загрузки системы. В процессе развития Unix (см. гл. 1) выделились два основных типа систем — BSD-подобные системы и SysV-совместимые. Первые были совместимы (полностью или частично) с операционной системой BSD, разработанной Калифорнийским университетом, а вторые использовали в качестве своего предка операционную систему Unix System V, разработанную компанией AT&T. Системы инициализации BSD и SysV отличаются набором входящих в них файлов и их назначением.*

Запуск и останов демонов осуществляется с помощью сценариев, расположенных в каталоге `/etc/rc.d/init.d`. Демон — это программа, которая в фоновом режиме периодически выполняет какие-нибудь действия. Например, демон **ftpd** непрерывно проверяет наличие пользовательских запросов на соединение по протоколу FTP.

Сценарии каталога /etc/rc.d выполняются автоматически при запуске системы. В этом каталоге есть несколько подкаталогов rcN.d, где N — это номер уровня выполнения. В большинстве случаев у вас будет установлен уровень 5 в качестве уровня по умолчанию. Этот уровень соответствует многопользовательскому режиму с автоматическим запуском системы X Window. В каталоге /etc/rd.d/rc5 .d находятся символические ссылки на сценарии, расположенные в каталоге /etc/rc.d/init.d.

Для запуска какого-нибудь демона нужно выполнить соответствующий ему сценарий в каталоге /etc/rc.d/init.d с опцией **start.** Для останова нужно запустить тот же сценарий, но с опцией **stop**.

Чтобы обеспечить автоматический запуск какого-нибудь сервера, нужно создать сценарий для его запуска и поместить его в каталоге /etc/rc.d/init.d. Затем, в зависимости от уровня выполнения, в каталоге rcN.d нужно создать символическую ссылку на этот сценарий.

Для выбора демонов, которые будут запускаться автоматически при загрузке системы, обычно используют профамму **drakconf** в операционной системе Linux Mandrake (см. рис. 5.2) или программу **setup** в ОС Red Hat Linux.

Если вы хотите сами создать сценарий для запуска своего демона, можете воспользоваться приведенным ниже шаблоном (см. листинг 5.3).

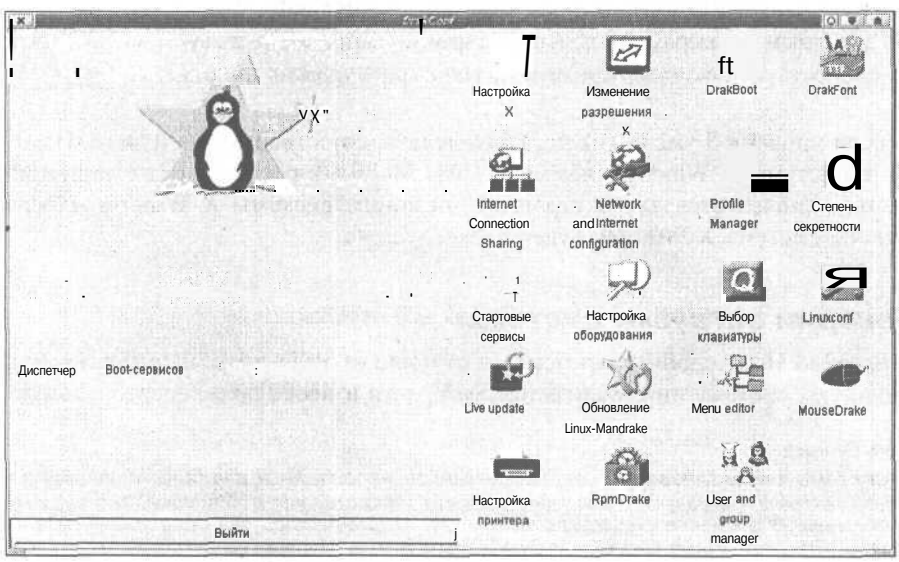


Рис. 5.2. *Конфигуратор DrakConf*

Листинг 5.3. Шаблон для запуска демона

```
#!/bin/bash
t Шаблон для запуска демона
# Подключаем библиотеку функций
. /etc/init.d/functions
# Определяемы параметры
```

```
case "$1" in
 start)
    # Запуск демона
    echo "Starting my_daemon..."
    daemon my_daemon
    touch /var/lock/subsys/my_daemon
    ;;
 stop)
    # Останов демона
    killproc my_daemon
    rm -f /var/lock/subsys/my_daemon
    rm -f /var/run/my_daemon.pid
    ;;
 status)
# Выводим статистику работы
    ;;
 restart|reload)
    # действия, выполняемые при перезагрузке демона
    :
    ;;
 *)
    # Произошел вызов без параметров
    echo "Usage: my_daemon {start|stop|status|restart|reload}"
    exit 1
esac
exit 0
```

Теперь рассмотрим весь процесс загрузки системы, начиная с самого начала: от включения питания. Итак, вы включаете питание, система выполняет процедуру самотестирования POST (Power On Self Test). Если во время самотестирования ошибок обнаружено не было, из главной загрузочной записи MBR (Master Boot Record) вызывается загрузчик операционной системы. Поиск загрузчика происходит на загрузочных устройствах в соответствии с последовательностью загрузки (Boot Sequence). Данная последовательность определяется в программе настройки компьютера — SETUP. Например, у вас может быть установлена такая последовательность загрузки: FLOPPY, HDD-0, CDROM

В этом случае система сначала будет искать загрузчик на дискете (диск A, устройство /dev/fd0). Если в дисководе нет дискеты, система перейдет к поиску загрузчика на первом жестком диске (HDD-0, устройство /dev/hda). Если же в дисководе есть дискета, но она не является загрузочной, вы получите соответствующее сообщение об этом. На этом этапе вы можете либо сменить дискету, либо вообще извлечь ее из дисковода, чтобы разрешить системе произвести поиск загрузчика операционной системы на жестком диске. Если и на жестком диске не будет обнаружен загрузчик, система перейдет к следующему элементу загрузочной последовательности — CDROM.

Предположим, что система нашла загрузчик на жестком диске. Загрузчик должен быть корректно установлен, иначе вы получите сообщение о невозможности загрузиться с данного носителя информации (жесткого диска). После этого управление будет передано программе LILO или любому другому загрузчику операционной системы Linux.

После того, как пользователь выберет нужное ему ядро, загрузка системы будет продолжена. Сначала будет загружаться ядро, а потом — программа **init.** Для полноты рассмотрения процесса загрузки я буду считать, что у нас установлен уровень управления 5.

Первыми будут выполнены процессы, которые указаны в действии **sysinit** файла /etc/inittab. Затем — процессы, перечисленные с помощью действий **boot и bootwait (см. п. 5.5).**

Обычно для действия **sysinit** выполняется сценарий загрузки /etc/rc.d/rc.sysinit:

```
si::sysinit:/etc/rc.d/rc.sysinit
```

На данном этапе загрузки системы (sysinit) выполняются следующие действия:

1. Устанавливается имя машины (hostname).
2. Конфигурируются параметры ядра.
3. Устанавливается раскладка клавиш и системный шрифт.
4. Активируются разделы подкачки.
5. Корневая система проверяется программой **fsck.** Если будут найдены ошибки, которые невозможно исправить автоматически, будет запрошен пароль администратора для входа в административный режим, что равноценно переходу на уровень выполнения 1. В этом режиме вы запустите программу **fsck** с параметром /, который означает проверку корневой файловой системы. После исправления всех ошибок введите команду **exit** для перезагрузки системы. Если программа **fsck** ошибок не обнаружила, файловая система монтируется в режиме чтение/запись.
6. Проверяются зависимости модулей ядра.
7. Выполняется проверка других файловых систем.
8. Монтируются локальные файловые системы.
9. Включаются квоты.
10. Подключается (не активизируется!) раздел подкачки. С этого момента система начинает использовать раздел подкачки.

После выполнения сценария загрузки /etc/rc.d/rc.sysinit выполняется сценарий /etc/rc.d/rc. Этому сценарию передается один параметр - номер уровня выполнения. В рассматриваемом случае - - это номер 5, поэтому будет выполнена команда:

```
/etc/rc.d/rc 5
```

Разумеется, данный сценарий будет выполнен при наличии в вашем файле /etc/initab строки:

```
l5:5:wait:/etc/rc.d/rc 5
```

Вы можете определить любое другое действие для уровня выполнения 5. Однако я не рекомендую вам этого делать: если вы написали свой сценарий загрузки, который работает лучше, чем предлагаемый разработчиками дистрибутива по умолчанию, значит, вам самое время написать свой дистрибутив! Запуск пятого уровня выполнения подразумевает запуск сценариев из каталога `/etc/rc.d/rc5.d/`.

После выполнения этого сценария будет выполнен сценарий `/etc/rc.d/rc.local`. Данный сценарий всегда выполняется последним, вне зависимости от уровня выполнения.

После запуска сценариев пятого уровня выполнения создаются виртуальные консоли и запускается менеджер дисплеев системы X Window (xdm).

5.7. Стандартные файлы протоколов (журналов)

В любой Unix-системе есть стандартные файлы протоколов (журналов), расположение которых может изменяться в зависимости от операционной системы. В Linux они находятся в каталоге `/var/log`. К стандартным протоколам относятся следующие файлы:

secure
auth.log
boot.log
dmesg
messages
syslog

В подкаталогах каталога /var/log находятся журналы (протоколы) других программ, например, в каталоге `/var/log/kernel` находятся журналы ядра, а в `/var/log/httpd` — журналы HTTP-сервера. По большому счету расположение журналов зависит от настройки соответствующих им программ, но в стандартном виде все они находятся в каталоге /var/log и его подкаталогах.

Назначение стандартных журналов представлено в табл. 5.6.

Подробно файлы журналов будут рассмотрены в следующем пункте (п. 5.8), в котором мы научимся управлять процессом протоколирования.

Стандартные журналы *Таблица 5.6*

Файл	Назначение
auth.log	Протоколирование аутентификации
user.log	Сообщения пользовательских программ
secure	Обычно в этот файл записываются сообщения об удаленном доступе к этой машине, например, сообщения от демона FTP о том, какие пользователи и когда регистрировались на данном сервере
messages	Остальные сообщения

5.8. Управление протоколированием

Этот раздел посвящен демону **syslogd**, а также управлению протоколированием сообщений системы и ядра с помощью этого демона. Прежде всего следует отметить, что демон находится в пакете sysklogd (если вы, конечно, используете Red Hat-совместимую систему), поэтому перед его использова-

нием нужно установить этот пакет. В большинстве случаев у вас пакет уже будет установлен, а демон syslogd запущен. Чтобы проверить этот факт, введите команду **syslogd.** Если демон запущен, то в ответ вы должны получить сообщение:

```
syslogd: Already running.
```

Убедиться в том, что демон запущен, можно также с помощью команды:

```
ps -ax I grep syslogd
```

Обратите внимание, что в пакет sysklogd на самом деле входят две программы: syslogd и **klogd**. Программа **syslogd** отвечает за протоколирование сообщений системы, а **klogd** — ядра.

5.8.1. Демон Syslogd

Демон **syslogd** (system logging-deamon) обеспечивает вид протоколирования, который поддерживается большинством программ. При этом демон **syslogd** пишет сообщения в файл /var/log/syslog. Записи в этом файле обычно содержат такие поля: **дата и время, хост, программа, сообщение.** Пример этого файла представлен ниже:

```
Jan 27 17:09:35 dhsilabs modprobe: modprobe: Can't locate module sound-service-1-0
Jan 27 17:09:35 dhsilabs modprobe: modprobe: Can't locate module sound-slot-1
Jan 27 17:12:28 dhsilabs kernel: VFS: Disk change detected on device ide1(22,64)
Jan 27 17:12:31 dhsilabs kernel: ISO 9660 Extensions: Microsoft Joliet Level 1
Jan 27 17:12:31 dhsilabs kernel: ISOFS: changing to secondary root
Jan 27 17:12:32 dhsilabs kernel: VFS: Disk change detected on device fd(2,0)
Jan 27 17:12:32 dhsilabs kernel: end_request: I/O error, dev 02:00 (floppy), sector 0
```

Например, из предпоследней записи мы можем узнать, что 27-го января 2002 года в 17:12 произошла смена носителя в устройстве fd, о чем нам любезно сообщило ядро системы (запись «программа» — kernel). А носитель (дискета) оказался подпорченным, о чем свидетельствует следующая запись — ошибка ввода/вывода (I/O error, dev 02:00 (floppy), sector 0).

Демон **syslogd** запускается автоматически при старте системы. Для его запуска предназначен сценарий /etc/rc.d/init.d/syslog. Как обычно, запустить демон самостоятельно вы можете с помощью команды: **/etc/rc.d/init.d/syslog start,** а остановить — **/etc/rc.d/init.d/syslog stop.** Для обеспечения автоматической загрузки нужно создать символическую ссылку на этот файл, например:

```
ls -s /etc/rc.d/rc5.d/@S30syslog /etc/rc.d/init.d/syslog.
```

В этом случае вы обеспечите запуск демона на пятом уровне запуска (автоматический запуск X Window). Если вы используете Linux Mandrake, включить и отключить автоматический запуск вы можете с помощью команды **drakxservices**. При использовании Linux Red Hat включите автозапуск демона с помощью конфигуратора **setup.** Демон **syslogd** можно запускать с опциями, указанными в табл. 5.7.

В табл. 5.7 указаны не все опции демона. Назначение всех остальных опций вы можете найти в справочной системе, введя команду **man syslogd.**

Опции демона syslogd *Таблица 5.7*

Опция	Описание
-a сокет	Этот параметр позволяет указать дополнительный сокет, который syslogd должен прослушивать
-d	Включает режим отладки. В этом режиме демон не будет использовать системный вызов fork(2) для переключения себя в фоновый режим и будет выводить больше отладочной информации
-f файл	Этот параметр определяет альтернативный файл конфигурации
-h	По умолчанию демон не перенаправляет сообщения, которые он получает от других узлов. Этот параметр позволяет перенаправить сообщения другим хостам, которые определены
-n	Этот параметр нужен, если syslogd запускается и контролируется программой init
-p сокет	Позволяет задать другой сокет Unix вместо /dev/log
-r	Позволяет принимать сообщения из сети. Данная опция появилась в версии syslogd 1.3
-v	Выводит версию демона syslogd

5.8.2. Сигналы

Демон **syslogd** реагирует на следующие сигналы: SYGTERM, SIGINT, SIGQUIT, SIGHUP, SIGUSR1, SIGCHLD. Реакция демона на сигналы описана в табл. 5.8.

Реакция демона на сигналы *Таблица 5.8*

Сигнал	Реакция
SIGTERM	Завершает работу демона
SIGINT, SIGQUIT	Завершает работу демона, если выключена отладка (debugging). Если же отладка включена, эти сигналы игнорируются
SIGUSR1	Включает или выключает отладку
SIGHUP	Перезапуск демона

5.8.3. Файл конфигурации

По умолчанию используется файл конфигурации /etc/syslog.conf. Кроме этого вы можете указать другой файл конфигурации с помощью опции -/ Давайте рассмотрим установки демона на примере обычного файла конфигурации (см. листинг 5.4).

Листинг 5.4.

```
# Протоколирование аутентификации. Файл протокола /var/log/auth.log
auth,authpriv.* /var/log/auth.log
# Префикс "-" используется, если вы хотите синхронизировать
# файл после каждой записи в него.
*.*;auth,authpriv.none -/var/log/syslog
# Сообщения пользовательских программ
user.* -/var/log/user.log

# Протоколировать все (кроме mail (почты)). Уровень info и выше.
# Частные (private) сообщения протоколироваться не будут (попе)
*.info;mail.none;authpriv.none -/var/log/messages

# Файл регистрации частных сообщений имеет ограниченный доступ.
# Обычно в этот файл записываются сообщения об удаленном доступе к ртой
```

```
# машине, например, сообщения от демона FTP о том, какие пользователи и
# когда регистрировались на данном сервере.
authpriv.* /var/log/secure

# Протоколирование почты
# Уровень отладки, информации и замечаний
mail.=debug;mail.=info;mail.=notice  -/var/log/mail/info
# Уровень предупреждений
mail.=warn  -/var/log/mail/warnings
# Уровень ошибок
mail.err  -/var/log/mail/errors

# Протоколирование демона cron. Уровни отладки, информации,
# предупреждений и ошибок
cron.=debug;cron.=info;cron.=notice  -/var/log/cron/info
cron.=warn  -/var/log/cron/warnings
cron.err  -/var/log/cron/errors

# Протоколирование ядра
kern.=debug;kern.=info;kern.=notice  -/var/log/kernel/info
kern.=warn  -/var/log/kernel/warnings
kern.err  -/var/log/kernel/errors

# Протоколирование очереди печати
lpr.=debug;lpr.=info;lpr.=notice  -/var/log/lpr/info
lpr.=warn  -/var/log/lpr/warnings
lpr.err  -/var/log/lpr/errors

# Протоколирование новостей
news.=debug;news.=info;news.=notice  -/var/log/news/info
news.=warn  -/var/log/news/warnings
news.err  -/var/log/news/errors

# Протоколирование демонов
daemon.=debug;daemon.=info;daemon.=notice  -/var/log/daemons/info
daemon.=warn  -/var/log/daemons/warnings
daemon.err  -/var/log/daemons/errors

f Критические сообщения
*.emerg *
# Сохранять ошибки почты и новостей (уровень err и выше)
# в отдельном файле
uucp,news.crit  -/var/log/spooler

# Загрузочные сообщения
local7.*  -/var/log/boot.log
```

Как вы уже заметили, файл конфигурации состоит из двух полей: **объект протоколирования и файл,** в который будут записываться сообщения, порождаемые этим объектом. Для каждого объекта можно указать один из уровней протоколирования: **debug, info, notice, warn, err.** Первые три относятся к информационным сообщениям. Уровень **warn** -- это предупреждения, а err — ошибки. Существуют специальные сообщения — критические. Обычно они выводятся прямо на консоль.

Как для обозначения объектов, так и для обозначения уровней протоколирования можно использовать символ *, который обозначает все объекты или все уровни. Например, если вы хотите протоколировать все сообщения демонов в файл /var/log/daemons, то используйте такую конструкцию: daemon.* /var/log/daemons.

Пример протоколирования всех сообщений уровня **emerg** (критический уровень) приведен выше. Если вы хотите отправлять сообщения не в файл, а в поименованный канал (FIFO), используйте символ | перед именем файла-потока.

5.8.4. Сетевое протоколирование

Сейчас разберемся, как обеспечить протоколирование в сети. Протоколирование в сети -- это перенаправление сообщений на демон **syslogd,** запущенный на другой машине, где они будут записаны на диск.

Для передачи сообщений используется протокол UDP. Он менее надежный, чем TCP, но отправление пакетов происходит несколько быстрее. Начните с того, что убедитесь, не закомментирована ли следующая строка в вашем файле /etc/services:

```
syslog 514/udp
```

Затем необходимо внести некоторые коррективы в файл конфигурации. Как и прежде, определите объекты протоколирования, а вместо файлов протоколов используйте параметр @hostname, где *hostname* — это имя компьютера, на который будут перенаправлены сообщения. Например, для перенаправления всех сообщений об ошибках на узел сети *hostname* можно использовать такую запись:

```
*.err @hostname
```

Для перенаправления всех сообщений используется запись:

```
*.* @hostname
```

Имя узла желательно указать в файле /etc/hosts, так как демон **syslogd** может быть запущен после сервера доменных имен или сервер DNS окажется недоступным.

Вы можете организовать центральный сервер протоколирования для всей вашей локальной сети. Для того, чтобы указать, какие хосты вы хотите протоколировать, используйте опцию -l **список_хостов**. В списке указываются простые имена машин, то есть без указания имени домена. Имена машин разделяются двоеточием (:). Возможно, вы также захотите использовать опцию -s для указания дополнительного сокета для прослушивания. Для перенаправления сообщений используйте опцию **-r** на машине-клиенте, при этом сообщения будут перенаправлены на сервер (см. табл. 5.7).

5.8.5. Демон klogd

Демон **klogd** предназначен для перехвата и протоколирования сообщений ядра Linux (klogd расшифровывается как kernel-logging deamon). В своей работе вы можете использовать параметры демона, указанные в табл. 5.9.

Параметры демона klogd *Таблица 5.9*

Параметр	Описание
-c n	Устанавливает уровень сообщений, которые будут выводиться на экран
-d	Режим отладки
-f файл	Записывать сообщения в указанный файл раньше демона syslogd
-i	Позволяет перезагрузить символьную информацию ядра о модулях
-I	Перезагружает статическую символьную информацию и информацию о модулях ядра
-n	Не переходить в фоновый режим. Этот параметр используется, когда демон управляется программой init
-o	Демон читает и протоколирует все сообщения, которые он найдет в буферах сообщений ядра. После одного цикла чтения/протоколирования демон завершает работу
-s	Заставляет демон klogd использовать системные вызовы для обращений к буферам сообщений ядра
-k файл	Использует указанный файл в качестве файла, содержащего символьную информацию ядра
-v	Выводит версию и завершает работу

Для просмотра сообщений ядра используется команда **dmesg**. Обычно она используется так:

```
dmesg | less
```

Данная команда выводит сообщения ядра при запуске системы. С помощью параметра -c этой команды можно очистить *ring-буфер* ядра. Параметр -n задает уровень сообщений, которые будут выводиться на консоль.

По умолчанию демон **klogd** вызывается системным вызовом для того, чтобы препятствовать отображению всех сообщений на консоль. Это правило не распространяется на критические сообщения ядра *(kernel panic),* так как эти сообщения все равно будут отображены на консоли.

Демон реагирует на сигналы: SIGHUP, SIGKILL, SIGINT, SIGTERM, SIGTSTP, SIGUSR1; SIGUSR2, SIGCONT. Сигналы SIGTSTP и SIGCONT используются для начала и завершения протоколирования сообщений ядра. Сигналы SIGUSR1 и SIGUSR2 аналогичны опциям -i и -I соответственно. То есть первый перезагружает информацию о модулях, а второй статическую информацию и информацию о модулях. Использовать сигнал GIGUSR1 (как и все остальные) можно так: `# kill -USR1 PID`.

5.8.6. Параметры ядра

Параметр **debug** ядра Linux задает уровень отладки. Сообщения ядра (важные и не очень) передаются через функцию **printk()**. Если сообщение очень важное, то его копия будет передана на консоль, а также демону **klogd** для регистрации сообщения на жестком диске. Сообщения передаются на консоль, потому что иногда невозможно запротоколировать сообщение на жестком диске (например, отказ диска).

Предел того, что будет отображаться на консоли, задается переменной **console_loglevel**. По умолчанию на консоли отображается все, что выше уровня DEBUG (7). Список уровней можно найти в файле `kernel.h`.

6

Русификация Linux

6.1. Русификация консоли

Выбор языка системы производится при установке системы или при помощи соответствующих конфигураторов: keyboarddrake в Linux Mandrake и **setup** в Linux Red Hat. В случае некорректного отображения русских символов нужно подправить файл `/etc/inputrc` (см. листинг 6.1), изменив следующие строки:

Листинг 6.1. Фрагмент файла inputrc

```
# 8Bits supports.
set meta-flag on
set convert-meta off
set input-meta on
set output-meta on
```

Для русификации **Midnight Commander** в меню **Options/Display bits** необходимо включить поддержку восьмибитного ввода и вывода. После чего требуется сохранить текущую конфигурацию.

6.2. Русификация системы X Windows

Как ни странно, при выборе русскоязычного варианта установки русские шрифты устанавливаются на диск, но не прописываются в конфигурационных файлах. Конечно, оконная среда **KDE** (K Desktop Environment), как и **Gnome,** в большинстве случаев использует эти шрифты, но некоторые элементы меню или подписи значков отображаются совсем не так, как нужно.

Для корректного отображения русскоязычной информации на экране в системе X Window нужно сделать следующее. Зарегистрировавшись как пользователь **root,** открыть в любом текстовом редакторе файл конфигурации XFree86 — `/etc/x11/XF86Config` и *перед* строкой

```
FontPath "unix/:-1"
```

добавить строку

```
FontPath "/usr/X11R6/lib/fonts/cyrillic/"
```

Естественно, пакет **XFree86-cyrillic-fonts** должен быть предварительно установлен. Проверить наличие этого пакета можно с помощью команды **rpm -qi XFree86-cyrillic-fonts**. Желательно проверить также каталог, в который был установлен данный пакет — возможно у вас он будет другим.

Теперь нужно перезапустить X-сервер. Для этого нажмите Ctrl+Alt+Backspace. Если X-сервер у вас не запускается автоматически при загрузке системы, то сейчас нужно запустить его, введя команду **startx**.

Если все же ваша программа не понимает русский язык, нужно прописать русскоязычный шрифт в ее конфигурационном файле или запускать ее с параметром **-Wt font.** Параметр **font** является именем шрифта, который вы хотите использовать. Подобрать нужный шрифт можно с помощью команды **xfontsel.** Украиноязычные шрифты в кодировке KOI8-U можно скачать с моей домашней странички — **http://dkws.narod.ru.**

В ОС Linux Mandrake имеет возможность очень быстрого преобразования TrueType-шрифтов Windows, если вас не устраивают стандартные экранные шрифта Linux. Конвертирование можно произвести с помощью программы **DrakConf.** Подключить TTF-шрифты можно и вручную, но рассмотрение этого вопроса выходит за рамки данной книги.

6.3. Русификация принтера

Напомню последовательность действий по подключению принтера:

1. Запустите **DrakConf.** Нажмите на кнопку «Настройка принтера», далее нажмите «OK» (см. рис. 6.1). В ОС Red Hat Linux роль конфигуратора **DrakConf** выполняет программа **printtool**.
2. Выберите тип принтера (локальный, удаленный, Netware, SMB). Укажите порт и модель принтера. Потом уточните некоторые параметры (активизируйте режим «Исправлять ступенчатую печать») и распечатайте пробную страницу.

Ваш принтер почти готов к работе, однако при печати русскоязычного текста на бумаге вы увидите все что угодно, кроме русских букв. Это происходит из-за того, что при создании Postscript-образа страницы в состав программы **ghostscript** не включены русские KOI8-шрифты. Их можно скачать по адресу **ftp://ftp.kapella.gpi.ru/pub/cyrillic/psfonts**.

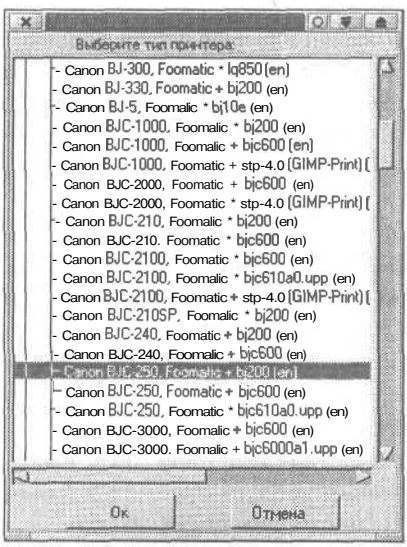


Рис. 6.1. *Подключение принтера*

Необходимые вам файлы:

1. gs-type1_koi8_fonts.tar — 614783 bytes
2. gs-type1_koi8_afm.tar — 29062 bytes

Также рекомендую скачать русифицированный **Fontmap** (1k). Далее распакуйте архив ***fonts.tar** в каталог `/usr/share/fonts/default/ghostscript`, а русифицированный **Fontmap** в каталог `/usr/share/ghostscript/5.10`.

7

Настройка сети

7.1. Установка сетевой платы. Настройка параметров сети

Модуль сетевой платы уже должен быть установлен, кроме случая, когда сетевая плата приобреталась после установки системы. Запустите конфигуратор **DrakConf** (см. рис. 7.1).

Запустите определение оборудования, чтобы убедиться, что сетевая плата распознается системой. Для этого щелкните на кнопке «Настройка оборудования» (см. рис. 7.2) и согласитесь на определение устройств ISA (Detect ISA devices).

Нажмите на кнопку «Настройка сети» (или выполните команду **netconf** — кому как нравится). Далее в окне **Network configurator** щелкните по кнопке «Basic host information» и в открывшемся окне введите имя машины, а затем на вкладке **Adaptor 1** (см. рис. 7.3) активизируйте адаптер (Enabled). После

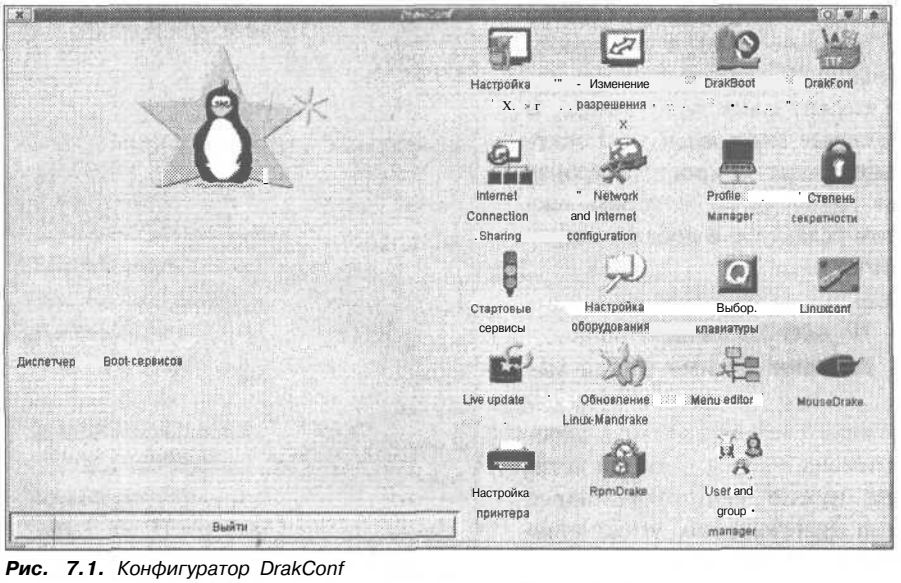


Рис. 7.1. *Конфигуратор DrakConf*

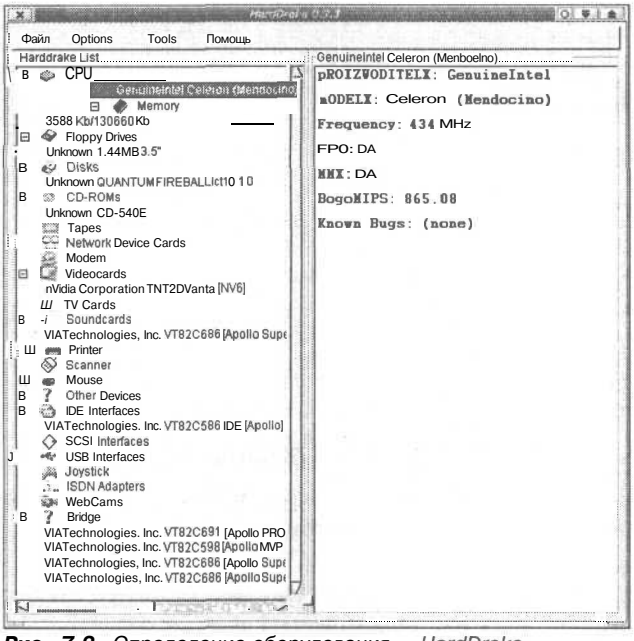


Рис. 7.2. *Определение оборудования — HardDrake*

этого введите информацию о своей сети и о своей плате (IP-адрес, сетевую маску, IO Port, Irq). В поле **NetDevice** укажите тип сетевого устройства —пусть будет **eth0** (от Ethernet), а в поле **Kernel Module** — имя модуля ядра, которое соответствует вашему сетевому адаптеру (например, модуль ne2k-pci соответствует плате NE2000 PCI).

Внимание!

Если вы используете сетевую плату PCI (например, ***ne2k-pci),*** *IO Port и IRQ устанавливать не нужно! Большинство сетевых плат совместимо с NE2000 или NE2000-PCI.*

Вернитесь теперь в окно конфигуратора сети (рис. 7.4) и настройте DNS. С этой целью активизируйте DNS, введите IP-адреса сервера и перечислите нужные вам домены. Всю необходимую для этого информацию можно узнать у администратора.

Если у вас небольшая домашняя сеть, то скорее всего, сервера DNS у вас не будет, а для преобразования IP-адресов в имена машин служит файл /etc/hosts. В этом случае ваша задача становится еще проще — просто откройте файл /etc/hosts в любом текстовом редакторе и добавьте строку типа:

IP_Addr имя_компьютера псевдоним

где: **IP_Addr**.....ваш IP-адрес;
hostname.....имя вашей машины.

Также следует добавить адреса и имена машин в вашей сети. Затем нужно установить адрес шлюза (gateway) по умолчанию (Routing and gateways).

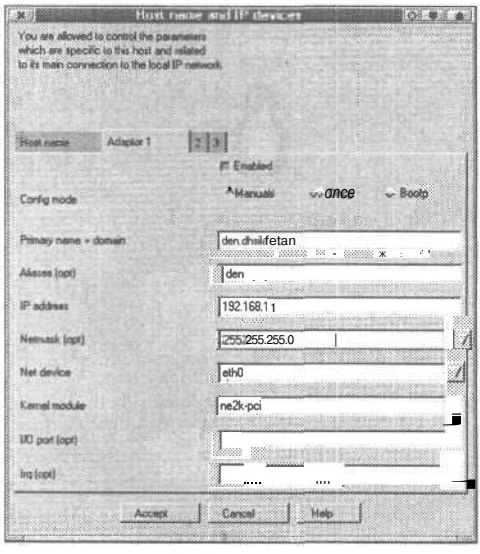


Рис. 7.3. *Параметры сетевого интерфейса*

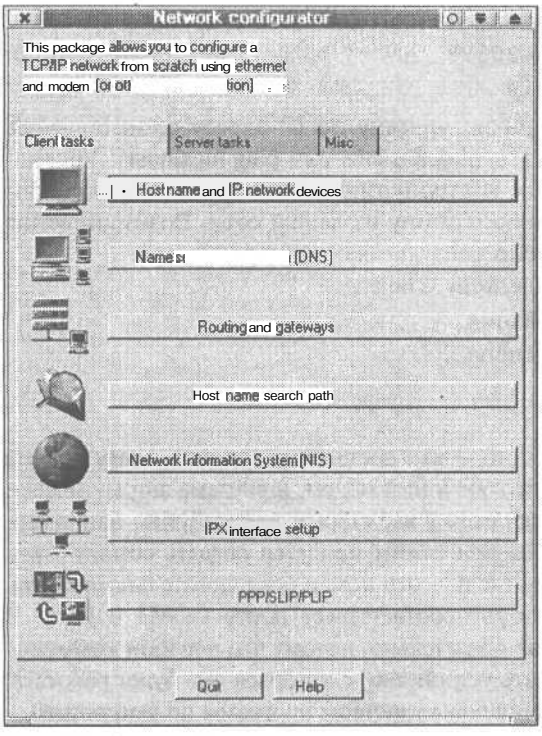


Рис. 7.4. *Окно конфигуратора сети*

При использовании сервера доменных имен еще нужно установить порядок поиска адресов. Это можно сделать в окне **Name service access** сетевого конфигуратора **(Host name search path): hosts, dns.** Это означает, что система сначала будет использовать локальную базу данных адресов, а затем обращаться к серверу DNS. Не отключайте режим **Multiple IPs for one host.**

Настройки системы DNS хранятся в файлах `/etc/hosts.conf` и `/etc/resolv.conf`. Если же конфигуратор **DrakConf** у вас недоступен (либо у вас не запущен сервер X, либо вы используете другую версию Linux), то вышеописанные действия можно сделать вручную.

Добавьте модуль сетевой платы:

`insmod rtl8139.o` (для Realtek 8139)
`insmod ne2k-pci.o` (для NE2000 PCI)

Эти же модули вам нужно добавить в файл `/etc/conf.modules`.

Программа **ifconfig** используется для конфигурации сетевого интерфейса, а **route** — таблицы маршрутизации.

ifconfig eth0 192.168.1.1 up — «поднимаем» сетевой интерфейс.

А более корректно это будет выглядеть так:

```
/sbin/ifconfig ethO ${IPADDR} broadcast ${BROADCAST} netmask ${NETMASK}
```

Теперь добавьте вашу сетевую плату в таблицу маршрутизации:

```
/sbin/route add -net ${NETWORK} netmask ${NETMASK} ethO
```

Укажите шлюз по умолчанию:

```
/sbin/route add default gw ${GATEWAY} netmask 0.0.0.0 metric 1
```

Перезапустите демон **xinetd** (или **inetd)** и проверьте настройки сети. Для проверки воспользуемся командой:

```
ping 127.0.0.1
```

127.0.0.1 — адрес обратной петли, т.е. все пакеты, которые отправляются на этот адрес, на самом деле не выходят за пределы локальной машины и возвращаются к ней. Этот адрес зарезервирован для служебных целей и может служить для проверки конфигурации сети. Если у вас возникли проблемы с этим адресом, активизируйте сервис **network.** При правильной настройке ваша таблица маршрутизации должна выглядеть подобным образом:

```
[root@dhsilabs /etc]# route
Kernel IP routing table
```

```
Destination     Gateway         Genmask         Flags   Metric  Ref    Use Iface
192.168.1.1     0.0.0.0         255.255.255.0   U       0       0        12 ethO
127.0.0.1       0.0.0.0         255.0.0.0       U       0       0        1 lo
```

Теперь можно пропинговать свою машину по IP-адресу ее интерфейса ethO и по ее имени (ping 192.168.1.1, ping **dhsilabs** или ping **localhost**). Убедившись, что проблем с локальными настройками не возникает, можно пропинговать какую-нибудь удаленную машину из вашей сети. Возникновение проблем на этом этапе обусловлено следующим:

1. Неправильность настроек на удаленной машине.
2. Неисправность сетевого оборудования.
3. Удаленная машина просто выключена...

7.2. Подключение модема

Модем подключается очень просто — вам достаточно вставить плату модема в корпус компьютера или подключить внешний модем, и система автоматически определит и установит его. В случае, если у вас старый дистрибутив, например, Red Hat Linux версии 6 и ниже, то вам самим придется создать ссылку /dev /modem на устройство `/dev/ttySn`, где п — это номер последовательного порта. Напомню, что устройство /dev/ttySO соответствует порту COM1 в DOS. В принципе, создавать ссылку даже не обязательно, потому что в любой коммуникационной программе можно указать устройство, с которым она будет работать. Устройство `/dev/modem` используется большинством программ по умолчанию.

Для проверки работоспособности модема можно использовать программу **minicom**. Это обычная терминальная программа. Перед ее запуском необходимо установить параметры программы. Это можно сделать командой minicom -s (см. рис. 7.5). При этом нужно изменить только устройство, которое будет использоваться в качестве модема.

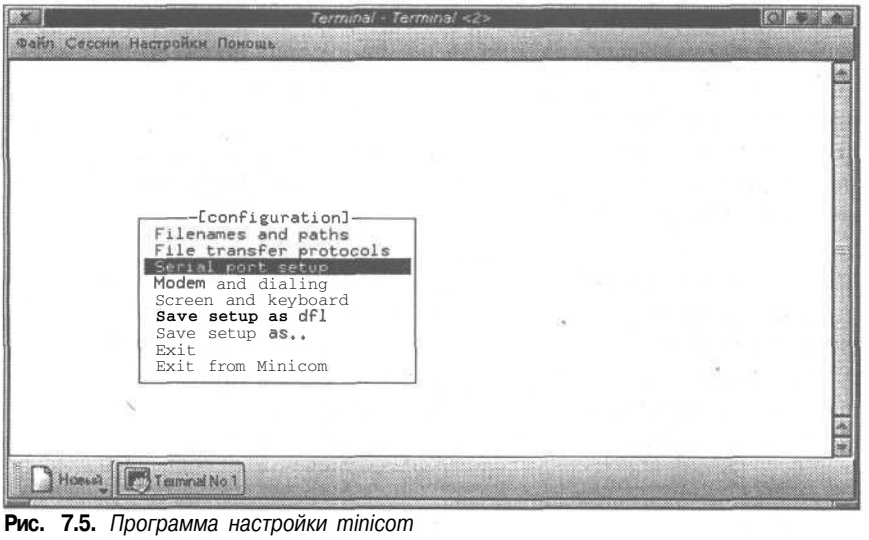


Рис. 7.5. *Программа настройки minicom*

Для тестирования модема обычно используются стандартные AT-команды. Инициализировать модем можно командой ATZ, поднять и положить трубку — **ATH1 и ATHO** соответственно, а набрать номер — **ATDPномер**, используя импульсную систему набора номера, и **ATDTномер**, используя тональную систему.

Ссылку можно также создать программой **modemtool** в RedHat, а в Linux Mandrake нужно воспользоваться все тем же конфигуратором **DrakConf**.

Будет справедливо отметить, что Linux не работает с программными модемами для Windows (win-модемы). А вот модемы, которые подключаются к шине USB, в ОС Linux использовать можно. Только для этого нужно включить в ядро поддержку шины USB и USB-модемов.

В последнее время некоторые производители программных модемов (например, Lucent) обратили внимание и на Linux-пользователей. Компания Lucent выпустила драйверы под Linux для своего модема. Данные драйверы находятся на компакт-диске, поставляемым с модемом. Однако я не рекомендую использовать программные модемы на сервере, так как это снизит производительность системы.

7.3. Подключение к Интернет

Рассмотрим, как можно подключить Linux к Интернет. Подключение Linux к Интернет имеет больший смысл, нежели Windows, так как Linux намного лучше и, что самое главное, быстрее работает с сетевыми устройствами. Лично у меня соединение с моим провайдером при использовании Linux работает гдето в два раза быстрее и не простаивает, как при работе в Windows.

Для настройки подключения можно пойти по пути наименьшего сопротивления и использовать все тот же конфигуратор или же программу **kppp,** но здесь я эти варианты рассматривать не буду. Попробуйте настроить все самостоятельно, как это делали все нормальные люди до создания KDE. Впрочем, на втором варианте можно было бы и остановиться. Программа **kppp** (см. рис. 7.6) входит в состав KDE и является стандартным дайлером, то есть программой, которая устанавливает соединение.

Как и любой другой дайлер, **kppp** выполняет следующие функции:

1. Устанавливает соединение с провайдером.
2. Регистрирует пользователя в удаленной системе.
3. Инициализирует PPP-соединение.

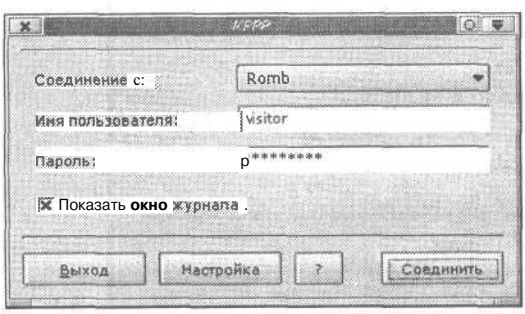


Рис. 7.6. *Программа kppp*

Интерфейс программы предельно прост. Однако, хотелось бы дать несколько рекомендаций по работе с этой программой:

1. Вам нужно изменить имя устройства, которое будет использовано **kppp,** то есть имя модема и уста-

новить его скорость. Но не подумайте только, что если вы установите скорость 56700, то ваш модем на 9600 заработает с большей скоростью.

2. Измените команду набора номера в окне «Команды модема» (см. рис. 7.7). По умолчанию используется тональный набор (команда ATDT). Для работы с импульсной системой набора номера используйте команду ATDP (рис. 7.8).
3. Нужно изменить права доступа к демону /usr/sbin /pppd, позволяющие пользователю его запускать. В противном случае запускать программу kppp вам придется так: su -c kppp.

Вот собственно и все, что касается программы kppp.

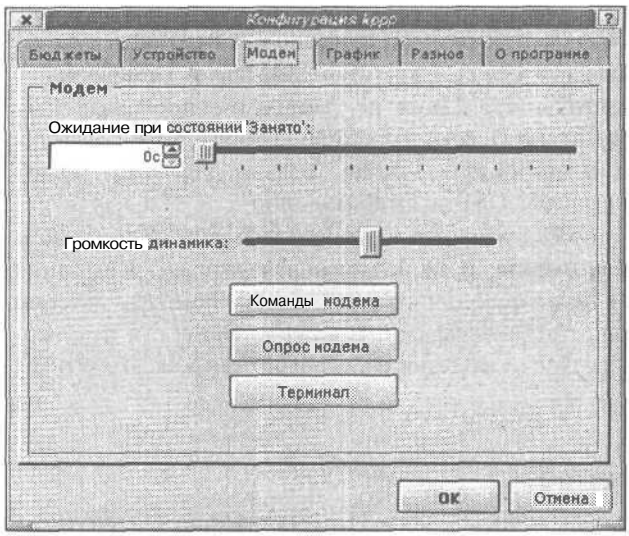


Рис. 7.7. *Вкладка «Модем»*

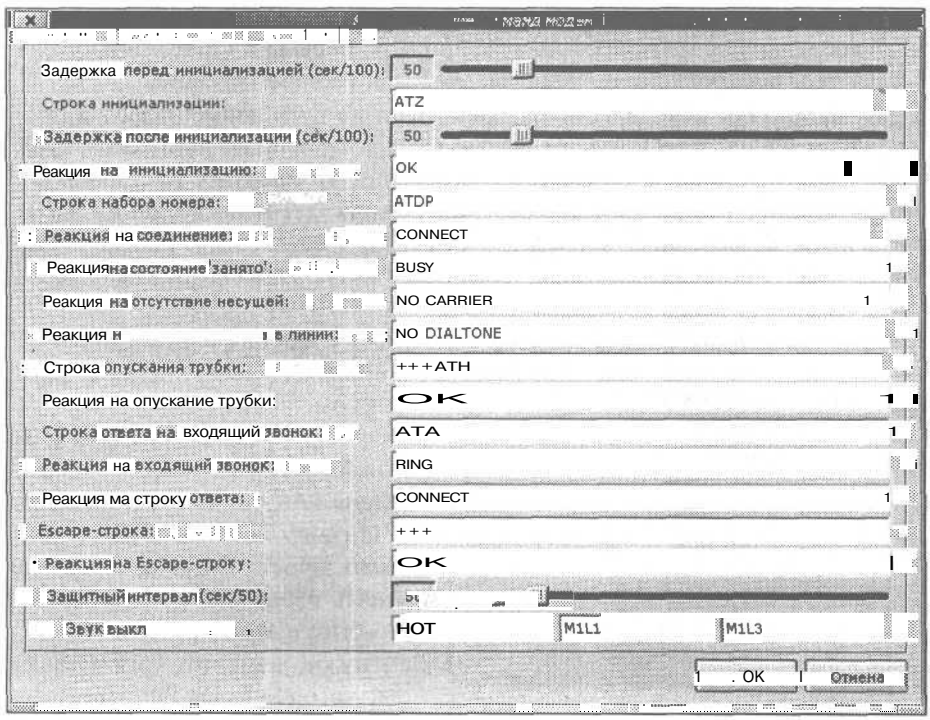


Рис. 7.8. *Команды модема*

Обычно соединение с провайдером устанавливается по протоколу PPP (Point to Point Protocol), и только в очень редких случаях по устаревшему протоколу SLIP (Serial Line Internet Protocol), поэтому протокол SLIP в этой книге рассматриваться не будет.

PPP — протокол «точка-точка» (Point-to-Point Protocol). Используется для обмена данными между компьютерами и удаленными локальными сетями.

Протокол PPP является составным, то есть он объединяет в себе сразу несколько протоколов. Основными из них являются:

» IPCP (IP Control Protocol) — внутренний протокол, обеспечивающий возможность сжатия данных и динамическое назначение IP-адреса удаленному пользователю.

» HDLC (High level Data Link Control) — протокол, осуществляющий кодирование данных перед отправкой их на линию.

• LCP (Link Control Protocol) -- протокол, предоставляющий средства по настройке, созданию и тестированию удаленного соединения.

» NCP (Network Control Protocol) — протокол, осуществляющий настройку протоколов сетевого уровня, вкладываемых в PPP.

В Linux PPP-соединение обеспечивается демоном **pppd,** конфигурационным файлом которого является файл /etc/ppp/options. Описание используемых в этом файле опций приведено далее.

Соединение по протоколу PPP, как правило, происходит по обычным телефонным линиям и может быть как с использованием аутентификации или без нее. Без аутентификации обычно устанавливаются соединения по выделенным линиям, о которых поговорим в следующей главе. Что касается аутентификации, то ее существует несколько способов:

1. **Терминальный.** Представляет собой интерактивный вход в сеть с использованием имени пользователя и пароля, передаваемых открытым текстом. Данный способ подразумевает создание сценария для автоматической регистрации.
2. PAP (Password Authentication Protocol) — протокол аутентификации по паролю. Протоколом PAP реализуется двухэтапный метод подтверждения прав доступа одного компьютера к другому (клиента к серверу). На первом этапе клиент посылает серверу запрос, в котором указывается имя пользователя и пароль. На втором этапе сервер обрабатывает эту информацию и отсылает ответ, в котором указывается удостоверен клиент или нет. Хотя PAP-аутентификация является лучше терминальной, но также пересылает пароль открытым текстом, что делает ее доступной для злоумышленников.
3. **CHAP** (Challenge Handsnake Authentication Protocol) — протокол аутентификации по запросу/ответу. Этот протокол предоставляет более безопасный механизм аутентификации. При этом проверка права доступа осуществляется в три этапа:
   * на первом этапе аутентификации сервер посылает клиенту специальный запрос;
   » на втором этапе клиент с помощью специальной функции MD5 вычисляет и отсылает серверу ответ. При формировании ответа учитываются имя пользователя, пароль и присланный сервером **запрос;**
   » на третьем этапе сервер сравнивает полученный от клиента ответ с предполагаемым и выдает соответствующий ответ, разрешая или запрещая доступ пользователю.

7.3.1. Терминальный способ

Не будем вдаваться в технические подробности, а опишем настройку соединения каждым способом. Начнем с самого простого терминального метода. Может быть, вы когда-нибудь подключались к провайдеру из-под Windows, устанавливая режим «Открыть окно терминала после набора номера»? Так вот это как раз этот способ аутентификации. Итак, рассмотрим подключение к одному из моих местных провайдеров, использующему как раз терминальный способ. Для этого потребуется следующая информация, которую можно узнать у провайдера, а жители города Кировограда могут использовать данный пример без изменений (см. табл. 7.1).

Параметры подключения к провайдеру *Таблица 7.1*

Параметр	Значение
Телефон для доступа	083
Имя пользователя для входа в сеть	dialup
Имя пользователя для доступа к серверу POP (почта)	name
Пароль для доступа в сеть	Password
Ваш IP-адрес	0.0.0.0
IP-адрес сервера DNS	194.183.166.3
Шлюз по умолчанию	194.183.166.3
Домен	frk.kr.ua (или host.kr.ua)
Прокси:порт	proxy.frk.kr.ua:8080

Пока ничего не забыл, нужно сделать несколько небольших замечаний.

Во-первых, имя пользователя для входа в сеть и для доступа к почтовику могут быть одинаковыми, а могут и отличаться. Все зависит от вашего провайдера. С точки зрения безопасности лучше иметь два различных имени и два разных пароля. Как вы видите, в данном случае используются два разных имени. В гл. 17 будут рассмотрены оба способа конфигурирования сервера входящих звонков.

Во-вторых, пароль я специально написал в таком виде: «PassWord». Этим я хочу показать, что при аутентификации учитывается регистр букв.

Далее, если у вас динамический IP-адрес (что скорее всего), то пишите 0.0.0.0 вместо нормального адреса. Кроме основного IP-адреса сервера DNS, может использоваться еще и вторичный сервер DNS. Этот адрес также лучше уточнить у провайдера. IP-адрес шлюза по умолчанию обычно совпадает с IP-адресом сервера DNS, так как обычно это один и тот же компьютер.

В свойствах браузера можно выбрать тип подключения прямой или через прокси-сервер. В данном случае нам все равно, потому что наш провайдер использует прозрачный прокси-сервер, а имя прокси-сервера я привел исключительно в демонстрационных целях.

Прежде чем приступить к настройке протокола PPP, установите необходимое программное обеспечение. Далее, я сначала опишу установку из RPM, а потом полностью компилирование **pppd** из исходников. Установить **PPP** вы можете с помощью команд:

```
# mount /mnt/cdrom
# rpm -ih /mnt/cdrom/Mandrake/RPMS/ppp*
```

Нужно также позаботиться о том, чтобы ядро поддерживало PPP. В большинстве случаев поддержка PPP уже встроена в ядро. Проверить наличие поддержки PPP можно с помощью команды: **dmesg | grep PPP.**

Желающие могут установить PPP из исходников. Последнюю версию можно скачать по адресу **ftp://ftp.linuxcare.com.au/pub/ppp/** . Установку PPP при этом можно выполнить с помощью таких команд:

```
t ./configure
# make
# make install
```

Естественно, что до этого необходимо сначала перейти в каталог с распакованными исходными текстами ppp.

Далее, возвращаясь к рассматриваемому примеру, установите имя машины:

```
# hostname name.frk.kr.ua
```

Затем может потребоваться подправить файл /etc/hosts . conf (хотя в большинстве случаев этого делать не надо):

```
order hosts, bind
multi on
```

Первая строка означает, что сначала адрес узла сети будет просматриваться в локальной базе данных -- в файле /etc/hosts, а затем будет произведено обращение к серверу DNS (если нужного адреса там нет). Вторая строка разрешает использовать сразу несколько IP-адресов. Теперь подправим файл /etc/hosts:

```
127.0.0.1    localhost.localdomain localhost
0.0.0.0      name.frk.kr.ua  name
```

Сейчас необходимо указать системе, какими серверами DNS нужно пользоваться. Эта информация хранится в файле /etc/resolv.conf

```
domain frk.kr.ua
nameserver 194.183.166.3
```

Можно также добавить еще и адрес вторичного сервера (добавив еще одну директиву **nameserver**). Всего можно указать четыре адреса.

Как уже отмечалось выше, процесс подключения состоит из нескольких этапов:

1. Соединение с сервером провайдера.
2. Регистрация.
3. Инициализация PPP-соединения.

Программы-дайлеры выполняют все эти функции, но в нашем случае эти функции выполняют следующий набор файлов:

```
демон pppd
/usr/sbin/ppp-on
/usr/sbin/ppp-off
/etc/ppp/ppp-on-dialer
/etc/ppp/options
```

Скрипт **ppp-on** предназначен для инициализации PPP-соединения, **ppp-off** -- для выхода из сети. Далее приведен пример сценария ppp-оп (листинг 7.1).

Листинг 7.1. Пример файла ppp-on

```
#!/bin/sh
PHONE=083
ACCOUNT=mylogin
PASSWORD=mypassword
LOCAL_IP=0.0.0.0        # если не динамический, то укажите
                        # нужный адрес
REMOTE_IP=0.0.0.0       # обычно 0.0.0.0
NETMASK=255.255.255.0
export PHONE ACCOUNT PASSWORD
DIALER=/etc/ppp/pp-on-dialer # полное имя сценария набора номера
exec /usr/sbin/pppd debug lock modem crtscts /dev/ttyS2 38400 \
    asyncmap 20A0000 escape FF kdebug 0 $LOCAL_IP:$REMOTE_IP \
    noipdefault netmask $NETMASK defaultroute connect $DIALER
```

Обратите внимание, что в рассматриваемом примере модем подключен к порту COM3, то есть к устройству /dev/ttys2, максимальная скорость работы для которого - - 38400 бит/с. Следующий файл: **ppp-on-dialer** (см. листинг 7.2).

Листинг 7.2. Пример файла ppp-on-dialer

```
#!/bin/sh
exec chat -v \
    TIMEOUT 3 \
    ABORT '\nBUSY\r' \
    ABORT '\nNO ANSWER\r' \
    ABORT '\nRINGING\r\n\r\nRINGING\r' \
    '' \rAT \
    'OK-+++\c-OK' ATH0 \
    TIMEOUT 30 \
    OK ATDP$PHONE \
    CONNECT '' \
    name:-name: $ACCOUNT \
password: $PASSWORD
```

Обратите внимание на выделенную строку: если у вас тональный набор номера, измените ее на OK ATD$PHONE. На этом практически все! Осталось только немножко подправить файл /etc/ppp/options (см. листинг 7.3).

Листинг 7.3. Пример файла /etc/ppp/options

```
# Первые две строки являются обязательными,
# хотя опцию domain вы можете
# использовать на свое усмотрение
lock
domain frk.kr.ua
# Оптимизация работы демона pppd
# Чтобы демон не уходил в фоновый режим
-detach
# Использовать линии управления модемом
modem
# Управление потоком данных
crtscts
# Устанавливает маршрут по умолчанию
defaultroute
asyncmap 0
# использовать максимальный размер передаваемого пакета в 552 байт
mtu 552
# использовать максимальный размер принимаемого пакета в 552 байт
mru 552
```

Теперь можно устанавливать соединение:

```
ppp-on
```

Можно также пропинговать провайдера:

```
ping 194.184.166.3
```

Для отключения введите ppp-off.

7.3.2. PAP- и CHAP-аутентификация

Большинство провайдеров и корпоративных PPP-серверов используют протокол PAP. Если используется протокол PAP, то вместо того, чтобы регистрироваться на таком сервере, используя имя пользователя и пароль, когда их ввод запрошен сервером, PPP-сервер, использующий PAP, не требует обычного ввода имени и пароля для входа в систему. Вместо этого информация установления подлинности пользователя идет как часть протокола управления связи (LCP), который является первым шагом установления связи PPP.

Если ваш провайдер использует протокол PAP или CHAP, внесите следующие изменения в файл /etc/ppp/options:

```
имя_пользователя логин
# Если вы используете протокол PAP
+pap
-chap
# Если вы используете CHAP
#+chap
#-pap
```

В зависимости от протокола аутентификации вам потребуется изменить файлы /etc/ppp/pap-secrets или /etc/ppp/chap-secrets. Файл /etc/ppp/pap-secrets в рассматриваемом случае должен выглядеть примерно так:

```
# Secrets for authentication using PAP
# client    server  secret    acceptable local IP addresses
  user      *       Password
```

Последнее поле остается пустым, так как используется динамический IP-адрес.

Протокол CHAP требует, чтобы не только вас опознала удаленная машина, но и вы опознали ее. Поэтому в файле /etc/ppp/chap-secrets нужно указать имя сервера в поле «server». Это имя вы можете уточнить у провайдера.

Примечание.

Настроить Интернет-соединение можно также с помощью конфигуратора ***DrakConf*** *в Linux Mandrake (см. рис. 7.9).*

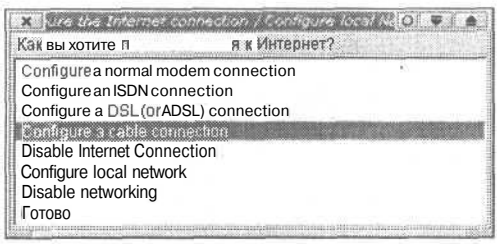


Рис. 7.9. *Настройка соединения с помощью конфигуратора DrakConf*

7.4. Настройка DSL-соединения

Для начала хотелось бы дать небольшое вступление. Многие телекоммуникационные компании разных стран мира начали внедрять различные варианты цифровых абонентских линий (DSL -- Digital Subscriber Line). Наиболее распространенной из них является технология асимметричной цифровой линии (ADSL). Кроме нее также используются службы симметричной цифровой линии (SDSL), цифровой линии с переменной скоростью (RADSL), сверхбыстрой цифровой линии (Very high-speed DSL, VDSL).

ADSL-модемы подключаются к обоим концам короткой линии между абонентом и АТС. При этом используется обычный телефонный провод. ADSL-модем использует полосу пропускания телефонного провода в виде трех каналов: быстрый канал передачи данных из сети в компьютер, менее быстрый дуплексный канал передачи данных из компьютера в сеть и простой канал телефонной связи, по которому передаются обыкновенные телефонные разговоры. Все три канала работают на разных частотах. Таким образом, вы можете одновременно работать в Интернет и разговаривать по телефону. Передача данных в канале «сеть-абонент» осуществляется со скоростью от 1.5 до 6 Мбит/с, в канале «абонент-сеть» -- от 16 Кбит/с до 1 Мбит/с. Скорость зависит от длины и качества линии. Асимметричный режим скорости передачи данных применяется потому, что обычно пользо-

ватель загружает из сети данные, а не закачивает данные в сеть. Поэтому выгоднее больший диапазон частот в полосе пропускания выделить под скачивание из сети.

Теперь переходим непосредственно к настройке.

7.4.1. Настройка соединения DSL в Linux Mandrake

Предположим, что вы используете Linux Mandrake. Если это не так, то ниже (п. 7.4.2) будет рассмотрен более универсальный способ конфигурирования, подходящий для любого другого дистрибутива.

Итак, начинать следует с установки пакетов **ppp, dhcpcd, pppoe-linuxconf, rp-pppoe**. При этом все действия нужно производить, зарегистрировавшись в системе под именем **root.** Перейдите в каталог с пакетами, обычно это `/mnt/cdrom/Mandrake/RPMS:`

```
cd /mnt/cdrom/Mandrake/RPMS
```

Установите пакеты:

```
rpm -i ppp-2.4.0-3mdk.i586.rpm
rpm -i dhcpcd-1.3.19pl1-1mdk.i586.rpm
rpm -i pppoe-linuxconf-1.2_1.21.1-1mdk.i586.rpm
rpm -i rp-pppoe-1.7-3mdk.i586.rpm
```

Примечание.

Версии пакетов у вас могут отличаться.

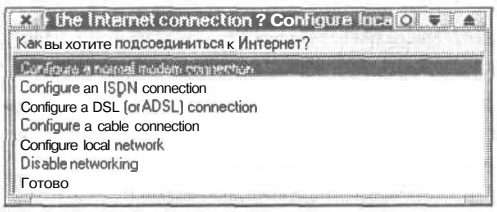


Рис. *7.10. Настройка сетевого соединения*

Теперь запустите конфигуратор **DrakConf** и выберите **Network and Internet configuration** (см. рис. 7.10). Далее выберите пункт **Configure DSL (or ADSL) connection** и укажите страну. На самом деле просто выберите «другие страны», если, конечно, вы не находитесь во Франции.

После этого необходимо указать способ подключения. Обычно здесь требуется выбрать **использование PPPOE** (use pppoe). Но, возможно, оборудование вашего провайдера не поддерживает **PPPOE,** в этом случае вам нужно будет выбрать **don't use pppoe.** Данную информацию опять-таки можно уточнить у вашего провайдера. Затем нужно ввести информацию о провайдере (см. рис. 7.11) и выбрать устройство DSL.

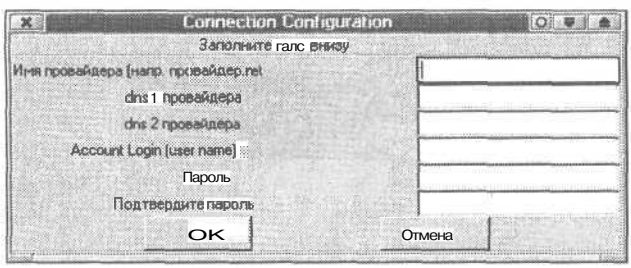


Рис. *7.11. Настройка соединения*

Примечание.

PPPOE *— Point to Point Protocol Over Ethernet — если дословно, протокол точка-точка через Ethernet. Большинство провайдеров, предоставляющих ADSL-доступ, используют именно этот протокол.*

По окончании установки в стартовые сервисы будет добавлен сервис **adsl**, который при запуске системы автоматически подключается к провайдеру.

7.4.2. Настройка соединения DSL в другом дистрибутиве

Для начала нужно установить **ppp, dhcpcd, rp-pppoe** и, если нужно, **pppoe-linuxconf.** Если вы используете Red Hat-совместимую систему, установку пакетов можно производить так же, как и в случае с Linux Mandrake. При использовании другого дистрибутива (например, Slackware) вам нужно прочитать рекомендации по установке вышеуказанного программного обеспечения самостоятельно.

Большинство провайдеров, предоставляющих ADSL-доступ, используют протокол **PPPOE. PPPOE** — Point to Point Protocol Over Ethernet — если дословно, протокол точка-точка через Ethernet.

Ваш дистрибутив должен обладать ядром версии не ниже 2.2.9 и демоном **pppd** версии 2.3.10 или выше. Если версия **pppd** у вас ниже необходимой, вы получите сообщение:

```
pppd: unknown option pty.
```

Существуют два метода установки **pppoe** — QuickStart и обычный. Если вам повезет, первый из них у вас будет работать.

QuickStart

После распаковки архива, зарегистрировавшись под **root**'ом, введите **./go** Скрипт **go** самостоятельно откомпилирует, установит и настроит **pppoe.** От вас только потребуется ввести информацию о провайдере.

Обычный метод установки

Обычно только четвертый шаг установки требует выполнения действий пользователем **root,** однако я рекомендую с самого сначала зарегистрироваться в системе как **root,** а потом производить установку. Если ваша система сконфигурирована таким образом, что вы не можете войти в систему как **root,** используйте команду **su,** чтобы воспользоваться правами суперпользователя.

Обычный метод установки состоит, если можно так сказать, из четырех шагов:

1. Распаковка

```
tar xzvf rp-pppoe-nnn.tar.gz
```

2. Настройка

```
./configure
```

3. Компиляция

```
make
```

4. Установка

```
make install
```

Теперь остается только все это настроить. Прежде чем приступить к настройке **pppoe,** убедитесь, что ядро «видит» вашу Ethernet-карту, которая будет использоваться вместе с модемом. Проверить это можно с помощью команды:

ifconfig ethO

Если эта сетевая плата единственная в вашей системе, N = 0. При этом вы должны будете увидеть примерно следующую строку (и еще несколько после нее):

ethO Link encap:Ethernet HWAddr XX:XX:XX:XX:XX:XX

где XX:XX:XX:XX:XX:XX — аппаратный адрес. Если вы его увидите, то карта сконфигурирована правильно. В противном случае настроить сетевую плату можно с помощью программы netconf.

Внимание!

При настройке сетевой платы не нужно назначать ей IP-адрес. Также не нужно ее конфигурировать во время загрузки (протокол BOOTP).

Сейчас самое время отредактировать файл /etc/ppp/pap-secrets, если ваш провайдер использует PAP-метод аутентификации, или /etc/ppp/chap-secrets — при использовании CHAP. Как уже отмечалось выше, обычно эти файлы имеют следующий формат:

имя_пользователя сервер парольIP-адрес

Введите, допустим: 'pupkin' * '123456' * при условии, что ваш логин pupkin и пароль 123456. Убедитесь, что ваш файл /etc/ppp/options пуст! В некоторых случаях, например, при выделенном соединении, этот файл обязательно должен содержать информацию. Тогда хотя бы не используйте директиву **lock.**

При использовании **pppoe** вам нужно немного изменить файл конфигурации /etc/ppp/pppoe.conf. Для чего откройте этот файл в любом текстовом редакторе и измените всего лишь два параметра:

ETH=eth1 — здесь измените eth1 на имя вашей сетевой платы, используемое для ADSL-соединения.

USER=bxxxxnxnx@sympatico.ca — здесь укажите правильный ID-пользователя.

Всю необходимую информацию можно узнать у провайдера. Далее следует обычная настройка сети. На всякий случай я опишу здесь все, чтобы вы лишний раз не перелистывали всю книгу. Итак, отредактируйте файл /etc/resolv.conf, добавив в него две строчки:

nameserver first_DNS
nameserver second_DNS

где first_DNS — это IP-адрес первичного сервера DNS вашего провайдера, а **second_DNS** — вторичного.

Можете также в начале файла добавить директиву **domain <имя_домена>** или **search <список_доменов>**.

Вместе с **pppoe** поставляется демонстрационный сценарий настройки **Firewall.** Для его установки сделайте следующее:

1. Возможно, вам придется немного отредактировать этот сценарий, указав нужные вам параметры.
2. Скопируйте его в каталог /etc/rc.d/init.d/firewall.
3. Введите **chkconfig firewall on**
4. Запустите firewall: **sh /etc/rc.d/init.d/firewall start**

Вышеуказанный способ работает только на Red Hat-совместимых дистрибутивах.

При условии, что установка прошла корректно, в Red Hat-совместимых системах уже будет обеспечена автоматическая установка ADSL-соединения во время загрузки (будет создан скрипт /etc/rc.d/init.d/adsl). Для того, чтобы вручную включить автоматическую установку соединения, введите **chkconfig —add adsl**. Если вы используете другой дистрибутив, например Slackware, то в этом случае добавьте в конец файла /etc/rc.d/rc.local строку **/usr/sbin/adsl-start**.

После этого вас можно поздравить — основная настройка уже выполнена. Теперь перейдем к дополнительной настройке. Если же работа ADSL-соединения вас и так устраивает, то вы можете смело перейти к следующей главе.

MTU (Maximum Transmit Unit) — это пакет информации максимального размера, который может быть передан или получен в нефрагментированном виде. Значение MTU указывает размер этого пакета. В общем случае увеличение этого параметра теоретически приводит к увеличению скорости обмена данными, а уменьшение — к повышению надежности. Необходимым условием использования повышенных значений MTU является поддержка их провайдером. Если заданное вами значение превышает предельное значение, установленное провайдером, то будет производиться дополнительная процедура разбивки и доследующего восстановления информации, что замедляет процесс передачи в целом. Если же заданное значение ниже предельно допустимого для провайдера значения, то клиент не использует до конца возможностей, предоставляемых оборудованием провайдера. Для рассматриваемого случая обычно используется значение MTU=1460.

Если ваша локальная сеть настроена для работы через **Firewall,** вы должны понизить значение MTU до 1452. При этом вы можете или установить значение MTU=1452 на всех интерфейсах сети или воспользоваться параметром **pppoe** «-m 1412». Использовать этот параметр -т гораздо проще, но требует некоторого дополнительного процессорного времени.

Если вы хотите вручную конфигурировать все узлы вашей сети, то в Linux это можно сделать следующей командой, которую желательно поместить в сценарий загрузки системы:

```
ifconfig ethO ratu 1452
```

Примечание.

От Windows, сами понимаете — не уйдешь, поэтому дам пару советов по настройке и этой ОС. При этом вам придется изменить пару параметров в реестре:

Ключ:
`[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Class\NetTrans\xxxx]`
***xxx** — ID интерфейса*
***Параметр:** MaxMTU*
Тип: REG_SZ
Дополнительная информация по настройке операционной системы Windows находится ПО адресу: http://support.microsoft.com/support/kb/articles/q120/6/42.asp

Если при установлении соединения вы получите сообщение *Message too long,* то это значит, что надо еще уменьшить значение MTU, допустим с 1452 до 1412.

Для управления ADSL-соединением вы можете использовать следующие команды:

adsl-start ..... запуск adsl;
adsl-stop ..... останов adsl.

Вести статистику вашего ADSL-соединения вы можете с помощью стандартного сценария, который входит в состав пакета **pppoe.** Для его работы необходим интерпретатор **Perl.**

7.5. *Настройка выделенных линий*

Протокол PPP может использоваться как для установления коммутируемых соединений, например dial-up, так и для организации выделенных (leased) линий. Для начала разберемся, что же такое выделенная линия? Сети с коммутацией пакетов (каналов) можно разделить на две группы: сети с динамической и постоянной коммутацией. В первом случае пользователь может установить соединение по собственной инициативе. Коммутация выполняется на время сеанса связи, а затем связь разрывается (по инициативе пользователя или по инициативе сервера). Во втором случае динамическая коммутация не осуществляется, а вместо этого пользователю (пользователям) разрешается заказать соединение на длительный период времени. Режим постоянной коммутации называется выделенной (dedicated) или арендуемой (leased) линией.

Будем считать, что необходимые вам пакеты уже установлены. Сам процесс установки пакетов **ppp** был описан в п. 7.3. Нужно также позаботиться о том, чтобы ядро поддерживало PPP. В большинстве случаев поддержка PPP уже встроена в ядро. Проверить наличие поддержки PPP можно с помощью команды: **dmesg | grep PPP.**

Предположим, вам нужно настроить два выделенных канала. Для определенности первый модем будет подключен к /dev/ttySO, а второй — к /dev/ttyS1. Файл /etc/ppp/options будет содержать глобальные настройки. Файлы конфигурации для модемов /dev/ttySO и /dev/ttyS1 будут называться соответственно /etc/ppp/options.ttSO и /etc/ppp

/option.ttyS1. **Отредактируйте файл** /etc/ppp/options.ttyS0, **как это показано в листинге 7.4.**

Листинг 7.4, Файл /etc/ppp/options.ttyS0

```
# Устройство
/dev/ttySO
# Скорость
57600
noauth
mru 1500
# ваш интерфейс: удаленный интерфейс
192.168.99.1:192.168.99.2
# маска подсети
netmask 255.255.255.0
bsdcomp O
chap-interval 15
debug
crtscts
mtu 552
mru 552
-detach
```

С устройством и скоростью все понятно. Особое внимание обратите на следующую запись: **192.168.99.1:192.168.99.2.** Между вашими модемами (собственно между «вашим» и тем, что на другой стороне) создается подсеть 192.168.99.0 (netmask 255.255.255.0). По окончанию конфигурирования выделенной линии вам нужно настроить маршрутизацию (см. man route). Если же линия у вас одна, то вы можете в конец файла /etc/options.ttyS0 добавить команду **defaultroute.** Эта запись добавляет маршрут в системную таблицу маршрутизации, используя удаленную сторону э качестве шлюза. Обычно эта запись удаляется при завершении соединения. Теперь разберемся со всеми остальными командами, задающими параметры соединения (см. табл. 7.2).

Параметры соединения *Таблица 7.2*

Команда	Описание
noauth	Не требует удаленную сторону назвать себя перед тем, как начнется обмен пакетами. Используйте параметр auth — если аутентификация нужна
crtscts	Использовать аппаратное управление потоком данных (напр., RTS/CTS), чтобы управлять потоком данных на последовательном порту
xonxoff	Использовать программное управление потоком данных (напр., XON/XOFF), чтобы управлять потоком данных на последовательном порту
mru n	Устанавливает значение MRU [Maximum Receive Unit] в n байт при договоренности. Демон pppd запросит удаленную сторону отправлять пакеты не более чем по n байт. Минимальное значение MRU 128. Значение MRU по умолчанию 1500. Для медленных соединений рекомендуется 296 (40 байт для заголовка TCP/IP + 256 байт данных)
mtu n	Устанавливает значение MTU [Maximum Transmition Unit) в n байт. Пока другая сторона не попросит меньшее значение при договоре о MRU, pppd будет требовать у сетевого кода ядра отправлять пакеты данных не более, чем по n байт через сетевой интерфейс PPP

Продолжение табл. 7.2

Команда	Описание
chap-interval интервал	С этой опцией pppd будет заново вызывать удаленную сторону каждые _интервал_ секунд
debug	Увеличить уровень отладки (то же что -d). Если эта опция есть, то pppd будет записывать в журнал все прибывшие и отправленные пакеты в читабельной форме. Пакеты будут регистрироваться в файлах протоколов через syslog. Эта информация может быть перенаправлена в файл соответствующей установкой /etc/syslog.conf. Если pppd скомпилирован с разрешенной extra-отладкой, он будет записывать сообщения в журнал, используя средство local2 вместо daemon
-detach	Не переходить в фоновый режим (иначе pppd будет это делать, только если указано последовательное устройство)

Оптимальными значениями **mru** и **mtu** являются 542 и 552 соответственно. Однако, для получения максимальной производительности, поэкспериментируйте со значениями этих параметров. Помимо вышеуказанных команд, для настройки ppp вы можете использовать команды, приведенные в табл. 7.3.

Дополнительные параметры соединения *Таблица 7.3*

Команда	Описание
connect программа	Задает программу для настройки линии
disconnect программа	Запустить данную программу после того, как pppd завершил связь
asyncmap	В качестве значения данного параметра указывается async-карта символов - 32-bit hex; каждый бит - символ, который надо представить в виде escape-последовательности, чтобы pppd мог его принять. 0x00000001 - это мака для '\x01', а 0x80000000 - маска для '\x1f'
local	Не использовать линии управления модемом
modem	Использовать линии управления модемом
lock	Указывает, что демон pppd должен создать файл блокировки для последовательного порта
passive	Разрешить опцию «passive» в LCP. С этой опцией pppd будет пытаться инициировать соединение; если ответ от другой стороны не принят, то pppd будет пассивно ожидать правильный LCP-пакет от другой стороны (вместо выхода, как делается без этой опции)
silent	С этой опцией pppd не будет передавать LCP-пакеты для инициации соединения, пока не придет правильный LCP-пакет от другой стороны (как опция «passive» в старых версиях pppd)
-all	Не разрешать договариваться о любых опциях LCP и IPCP (будут использоваться значения по умолчанию)
-am	Запретить договариваться о asyncmap
-ip	Не договариваться об IP-адресе (адрес должен быть указан или в options, или в командной строке)
-mru	Запретить договариваться о MRU (Max Receive Unit)
-pc	Запретить сжатие полей протокола
+pap	PAP-аутентификация
-pap	Отказаться от PAP-аутентификации
+chap	CHAP-аутентификация
domain имя_домена	Добавить имя домена к имени машины
name имя_машины	Установить имя машины (в целях аутентификации)
user имя	Установить имя пользователя для аутентификации этой машины на другой стороне, используя PAP. Нельзя использовать вместе с параметром name
login	Использовать базу данных паролей для идентификации удаленной стороны, используя PAP
idle n	Если соединение не используется в течение n секунд, то оно будет разорвано
speed	Задет скорость обмена с модемом (пропускную способность порта)

После того как настройка ppp завершена, можно перейти к редактированию inittab. Добавьте в /etc/inittab следующие строки:

```
7:2345:respawn:/usr/sbin/pppd file /etc/ppp/options.0 > /var/log/pppS0.log
8:2345:respawn:/usr/sbin/pppd file /etc/ppp/options.1 > /var/log/pppS1.log
```

7 и 8 — это порядковые номера, **1...6** уже заняты для системных консолей ttyl-tty6. 2345 - - уровни запуска. Весь inittab должен выглядеть примерно так, как это показано в листинге 7.5:

Примечание.

В процессе загрузки операционная система Linux вызывает фоновый процесс ***init*** *(PID= 1), который обрабатывает конфигурационный файл /etc/inittab и затем запускает сценарий запуска системы /etc/rc.d/rc.sysinit. Этот сценарий, в зависимости от режима запуска (уровня выполнения), запускает разные демоны (фоновые задачи). Уровень выполнения задается в файле inittab. В нашем примере (см. листинг 7.5) уровнем выполнения по умолчанию является третий уровень. Перейти из одного уровня на другой можно с помощью команды* ***init N,*** *где* ***N*** *— номер уровня. Подробнее об этом и многом другом, связанном с начальной загрузкой системы, вы можете прочитать в п 5.5.*

Листинг 7.5. Файл inittab для выделенной линии

```
id:3:initdefault:
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
t Things to run in every runlevel.
ud::once:/sbin/update
#  Trap  CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
7:2345:respawn:/usr/sbin/pppd file /etc/ppp/options.0 > /var/log/pppS0.log
8:2345:respawn:/usr/sbin/pppd file /etc/ppp/options.1 > /var/log/pppS1.log
  # Run xdm in runlevel 5
  # xdm is now a separate service
  x:5:respawn:/etc/X11/prefdm -nodaemon
```

Не забудьте надлежащим образом настроить ваши модемы! Для этого вам нужно прочитать документацию по вашему модему. Обычно для инициализации модемов используется AT-команда AT&L1. Однако эта команда поддерживается не всеми модемами.

7.6. Перед настройкой сервера

Перед тем как перейти к настройке сервера, нужно решить пару организационных вопросов. Например, где расположить наш сервер и какое железо предпочтительнее использовать. Вы уже произвели базовую настройку операционной системы, ваш будущий сервер прекрасно, как я надеюсь, работает в вашей локальной сети. Теперь самое время решить, где его расположить и кто будет иметь доступ (кроме вас, естественно) к нему. Очень важно правильно расположить сервер. Теперь это не рабочая станция, практически не представляющая для злоумышленника никакого интереса, а СЕРВЕР ВАШЕЙ СЕТИ!!!

Во-первых, сервер должен быть размещен в отдельном помещении, доступ в которое ограничен. В идеальном случае в серверную комнату никто, кроме вас и, возможно, вашего помощника, не должен входить. Запомните: чем меньше посторонних (да и сотрудников) имеют доступ к серверному помещению, тем меньше проблем у вас будет. Простое нажатие на «Reset» может вызвать простой сервера (а значит и всей сети) от одной-двух минут до двух-трех часов. Например, вы установили пароль в программе первоначальной настройки компьютера (SETUP) и каждый день, придя на работу, вводите его для запуска сервера. Допустим, что во время вашего отсутствия кто-то нажал на «Reset» или просто выключил сервер. Так как пароль знаете только вы, это может вызвать простой сети на все время вашего отсутствия.

В некоторых материнских платах можно программно отключить кнопку перезагрузки (например, Chaintech CT 6ATA2). Я рекомендую использовать эту возможность, если она есть. В крайнем случае, можно просто отключить «Reset», разобрав компьютер. Перед этим (на всякий случай) прочитайте документацию по материнской плате.

Приведенный пример несколько некорректен, так как серверы обычно работают в бесперебойном режиме и не выключаются даже ночью, но это зависит от специфики вашего предприятия. Например, если сервер используется только для выхода в Интернет сотрудников сети, то его работа ночью просто не имеет смысла. Сейчас рассмотрим более серьезные последствия непредвиденной перезагрузки сервера.

То, что ваша сеть будет временно недоступна и простаивать — это, как говорится, только вершина айсберга. Если ваш сервер обслуживает корпоративную базу данных, неожиданное выключение питания или перезагрузка может повлечь за собой куда более весомые потери — потерю информацию, а значит, времени на ее восстановление (если такое вообще окажется возможным) и, естественно, средств. Выходит, что самым «продвинутым» хакером может стать обыкновенный электрик, который просто выключит напряжение на щитке возле серверной комнаты. Отсюда вторая истина: не пожалейте денег на хороший *источник бесперебойного питания* (ИПБ). В условиях наших электросетей ИБП быстро себя окупит. Существует несколько типов ИБП:

1. Резервные (backup).
2. Линейно-интерактивные.
3. Онлайновые.

Не буду утруждать вас различиями между этими видами ИБП, а скажу, что в нашем случае подойдет третий тип. Именно онлайновые ИБП обеспечивают бесперебойную работу сервера, а также различного чувствительного оборудования, например, измерительного. Стоимость таких ИБП примерно в два-три раза выше резервных ИБП такой же мощности. Если ваши средства ограничены, подойдут и линейно-интерактивные ИБП. В случае с рабочей станцией можно использовать ИБП мощностью 300 ВА (VA), но в нашем случае нужен ИБП как минимум на 1200...2000 ВА (VA). Из недорогих могу порекомендовать линейно-интерактивный ИБП Back-UPS PRO 1400 VA (670 W) производства компании APC.

Теперь перейдем ко второму вопросу — *аппаратному обеспечению.* Сразу нужно отметить, что комплектовать сервер нужно, исходя из тех задач, которые он должен выполнять. Если вы настраиваете шлюз для доступа к Интернет вашей локальной сети, в которой несколько пользователей, то подойдет и такая минимальная конфигурация: Pentium 166/200 (или PPRO 200) 128MB ОЗУ/4-6Gb HDD. А если вы собираетесь стать провайдером, то есть предоставлять доступ к Интернет на коммерческой основе другим организациям, такой конфигурации явно будет маловато. Прежде всего, нужно увеличить объем ОЗУ, установить быстродействующий жесткий диск или даже несколько дисков для обеспечения высокой скорости операций прокси-сервера. А подключить такой жесткий диск к старой материнской плате у вас, скорее всего, не получится, а если он и будет работать, то в два-три раза медленнее.

При конфигурировании сервера баз данных крупной организации, например, банка, вам не обойтись без двухпроцессорной машины. Поэтому сейчас рассмотрим «максимальную» конфигурацию.

Основным компонентом любого компьютера является материнская плата. В свою очередь, основным компонентом материнской платы является ее чипсет. В 1999-2000 годах популярным чипсетом для построения двухпроцессорных систем был Intel 440BX. Но сегодня он не обеспечивает должного уровня производительности. Во-первых, не обеспечивается поддержка частоты системной шины (FSB) 133 МГц, что не позволяет использовать с максимальной отдачей процессоры Intel Pentium III. Во-вторых, максимальный объем ОЗУ, поддерживаемый чипсетом Intel 440BX, равен 1 Гб. При работе с некоторыми базами данных этого может оказаться недостаточно. Этот чипсет также не поддерживает 64-разрядную шину PCI и интерфейс ATA 66 (не говоря уже о ATA/100). Я бы не порекомендовал его использовать при комплектации сервера.

Следующими серверными чипсетами компании Intel являются i820 и i840. Оба чипсета поддерживают частоту FSB 133 МГц, ATA/66, объем ОЗУ 1 Гб и 2 Гб соответственно. Но есть одно но: оба чипсета требуют использования памяти RDRAM. Системы, использующие *Rambus DRAM,* оказываются чрезвычайно дорогими, учитывая их довольно невысокое быстродействие. А использование SDRAM снижает быстродействие системы.

Я рекомендую использовать чипсеты ServerWorks семейства ServerSet III (IV). Несмотря на их высокую стоимость, они обладают куда более высоким

показателем цена/производительность, чем чипсет Intel 840. Чипсет ServerSet III поддерживает 16GB ОЗУ, частота FSB равна 133 МГц. Используются модули ECC Registered RAM. В качестве процессора можно установить два процессора Pentium III или четыре процессора Хеоп. Также обеспечивается поддержка 64-разрядная шина PCI. Чипсеты ServerSet III обладают встроенным IOAPIC (I/O Advanced Programmable Interrupt Controller).

Чипсеты семейства ServerSet IV поддерживают процессоры Intel Pentium IV, память DDR SDRAM (200 МГц), максимальный объем ОЗУ равен 64 Гб.

Производством материнских плат с чипсетом ServerSet III занимаются такие компании как SuperMicro (www.supermicro.com), Туап (www.tyan.com), а также некоторые другие. О стоимости таких плат говорить не стану, так как цены, скорее всего, изменятся к моменту выхода этой книги из печати.

Теперь перейдем к мозгу сервера — процессору. Как вы заметили, при обзоре чипсетов я ни разу не упомянул процессоры других производителей, кроме Intel. И не случайно, так как я рекомендую использовать именно процессоры Intel, которые обеспечивают более высокую степень надежности по сравнению с процессорами других производителей.

При покупке оперативной памяти следуйте простому правилу: чем больше, тем лучше. Лишняя память никогда не помешает. Покупайте такой объем ОЗУ, какой вы можете себе позволить. Только перед этим убедитесь, какой тип памяти поддерживает ваш чипсет, а также максимальный объем ОЗУ.

Лучше всего подойдет жесткий диск с интерфейсом SCSI, обеспечивающий 10000 rpm (rotates per minute). В крайнем случае подойдет и ATA/100(133) на 7200 rpm и двумя мегабайтами кэша. Если вы заботитесь о сохранности своих данных, нужно позаботиться об обеспечении должного уровня избыточности. Было бы очень хорошо, если ваша материнская плата обладала контроллером RAID. Если у вас нет средств на приобретение аппаратного контроллера RAID, ОС Linux предоставляет средства для создания программных контроллеров RAID — об этом говорилось в гл. 4. При использовании программных контроллеров нужно учитывать, что при этом увеличивается нагрузка на центральный процессор.

Хорошо зарекомендовали себя винчестеры IBM, WD, Quantum. Сейчас у меня установлено два винчестера Quantum. Единственный недостаток этих винчестеров -- пожалуй, высокий уровень шума, создаваемый ими при работе (особенно при открытом корпусе), но для сервера это не столь существенно. А в остальном эти винчестеры довольно быстрые и надежные.

Не покупайте дешевые сетевые платы для сервера. Такую роскошь, как пятидолларовую плату *Realtek,* можно установить разве что на рабочей станции. Серверу приходится обрабатывать запросы многочисленных пользователей, поэтому нужно покупать сетевой адаптер со встроенным процессором. При этом снижается нагрузка на центральный процессор и повышается производительность системы.

Если вы устанавливаете модемный пул, я бы порекомендовал использовать оборудование компании ZyXEL. Модемы ZyXEL зарекомендовали себя как очень надежные модемы, способные работать практически на любых линиях.

8

Конфигурирование сервера

8.1. Суперсерверы inetd и xinetd

В данной главе пойдет речь об общей настройке Интернет-суперсерверов **inetd и xinetd,** а также о настройке сервера **xinetd** для работы с протоколом **IPv6.**

Для начала все же определимся, почему **inetd(xinetd)** называется суперсервером? Да потому, что он отвечает за установление TCP-соединения, то есть он прослушивает пакеты и запускает необходимые программы для обработки информации. Таким образом, получается, что сервер **inetd(xinetd)** управляет другими серверами и потому называется суперсервером. Например, если в запросе клиента будет требование установить соединение с двадцать первым портом, то суперсервер вызовет сервер **ftp,** конечно, при условии, что соединение с 21-м портом разрешено (в противном случае клиент получит сообщение *Connection refused).*

По правде говоря, все не так просто как я описал — на практике все намного сложнее: за установление TCP-соединений отвечает демон **tcpd** (в более ранних версиях Linux его не было), программы-сервисы **(httpd, ftpd)** могут постоянно находиться в памяти (режим **standalone),** в этом случае они сами обрабатывают пакеты, и, соответственно, суперсервер их уже не вызывает.

8.1.1. Настройка сервера inetd

Для начала разберемся с настройкой **inetd.** Этот сервер использовался в дистрибутиве RedHat до версии 7, в более новых версиях он заменен на **xinetd** (описание этого суперсервера приведено далее в п. 8.1.4...8.1.7). При конфигурирования **inetd** вам потребуется отредактировать два файла — `/etc/inetd.conf` и `/etc/services`. Первый, собственно, и есть файл конфигурации суперсервера, а во втором перечислены все сетевые службы, которые доступны в вашей системе. Формат файла `/etc/services` следующий:

```
Имя_службы  Порт/Протокол  Псевдоним_службы
```

Листинг 8.1. Фрагмент файла /etc/services

```
pop-2          109/tcp    postoffice      # POP version 2
.pop-2         109/udp
pop-3          110/tcp                    # POP version 3
pop-3          110/udp
```

postoffice в данном случае является псевдонимом. Для некоторых служб могут потребоваться использование нескольких протоколов (как в листинге 8.1 для POP3 используется два протокола — TCP и UDP) и/или нескольких портов (см. листинг 8.2).

Листинг 8.2. Несколько портов для сервиса ftp (RedHat)

```
ftp-data       20/tcp
ftp            21/tcp
```

В других версиях ftp может потребоваться только одна запись — ftp 21/tcp.

Из соображений безопасности лучше закомментировать символом # сервисы, которые вы не планируете использовать, например, если у вас роутер, то зачем вам **sendmail** (port 25)?

Теперь переходим к файлу /etc/inetd.conf. Каждая запись в этом файле имеет следующий формат:

Имя **Тип_сокета** Протокол Флаги Пользователь Путь Аргументы

где: Имя..... имя сетевой службы, которое должно быть указано в файле /etc/services.

Тип **сокета** ... в этом поле указывается тип сокета, то есть тип технологии доставки данных для указанной службы. Наиболее часто используются значения: stream (поток) — для протокола TCP, **dgram** (датаграмма) — для протокола UDP и raw – непосредсвенно для протокола IP.

Протокол..... имя протокола.

Флаги..... с помощью флагов указывается статус ожидания. В качестве значения этого поля указывается ключевое слово wait или nowait. Если указано wait, то суперсервер inetd будет ожидать завершения работы данной сетевой службы на данном сокете, прежде чем перейти в состояние ожидания других запросов других служб на подключение к этому сокету. То есть в данной ситуации суперсервер перестает «слушать» порт до тех пор, пока на нем не завершит работу уже запущенная сетевая служба. Значение nowait приводит к обратной ситуации: после запуска описываемой сетевой службы суперсервер продолжает прослушивать сокет в ожидании других подключений. Обычно для служб с типом сокета stream устанавливается статус ожидания nowait, а для служб с типом сокета dgram — статус ожидания wait.

Пользователь .. в этом поле указывается имя пользователя, с правами которого запускается описываемая сетевая служба (соответствующий ей сервер).

Путь.....в этом поле указывается полное имя сервера (включая путь к нему), который должен быть запущен для обслуживания данного соединения.

Аргументы.....все оставшиеся поля воспринимаются как аргументы для запускаемого сервера.

Ниже (см. листинг 8.3) приведен пример записи в файле `/etc/inetd.conf`.

Листинг 8.3. Фрагмент файла /etc/inetd.conf

```
ftp stream tcp nowait root/usr/sbin/tcpd in.ftpd
```

где: ftp.....имя сетевой службы;

stream.....задает тип сокета **stream** (потоковый сокет);

tcp.....протокол (указан протокол tcp, так как именно этот протокол использует служба FTP в качестве протокола транспортного уровня);

nowait.....суперсервер продолжает «слушать» порт после выполнения одного сервера для определенного порта;

root.....сервер FTP будет запущен с правами root;

/usr/sbin/tcpd.. сервер, который будет вызван для обработки соединения;

in.ftpd.....аргумент, то есть программа, которую должен выполнить **tcpd** после проверки некоторой информации (о ней немного позже).

Еще вы можете написать и так, для прямого вызова службы ftp (ProFTP):

```
ftp stream tcp nowait root /usr/sbin/in.proftpd in.proftpd
```

В данном случае сразу будет вызван демон **ProFTP**. Запись **in.proftpd** является ссылкой на **proftpd**. Если будете использовать такой вызов **ProFTP**, позаботьтесь о том, чтобы **proftpd** имел тип **inetd, а не standalone.**

8.1.2. Настройка tcpd

Демон **inetd** является довольно удобным в использовании средством для организации работы Интернет-сервера. И все было бы замечательно, если бы не одно «но». А это «но» заключается в том, что разработчики **inetd** очень мало уделили внимания защите, что является недопустимым в нашей суровой Интернет-действительности. Восполнить этот недочет призван демон **tcpd** (система TCP-Wrappers). Так что давайте теперь разберемся, что же представляет из себя демон **tcpd,** который является еще одним барьером в системе безопасности.

Демон **tcpd** аутентифицирует удаленных пользователей и проверяет корректность их запросов. С помощью этого демона можно ограничить запросы с удаленных компьютеров.

Файл hosts.allow содержит список хостов, которым разрешено подключаться к вашей системе, а hosts.deny — запрещено. Записи имеют формат **служба:хост.домен**. Если вы хотите разрешить или запретить доступ всем, используйте ALL. Запись ALL:ALL открывает или закрывает доступ к вашему компьютеру для всех остальных компьютеров и для всех видов сервисов (см. листинг 8.4).

Листинг 8.4. Файл /etc/hosts.allow

```
http:ALL
ftp:ALL
ALL:server.dhsilabs.com
```

В листинге 8.4 доступ к **http** и **ftp** разрешен всем, но доступ ко ВСЕМ сервисам разрешен только компьютеру server.dhsilabs.com.

8.1.3. Протокол IPv6

Думаю, что основной момент настройки понятен, и теперь переходим к протоколу IPv6. Схема 32-разрядной адресации протокола IPv4 привела к дефициту IP-адресов. В новой версии протокола IP (IPv6, ранее именовавшегося IPng — IP next generation) адрес состоит из 16-ти октетов и изображается в виде восьми пар октетов, разделенных двоеточиями. В версии 6 используется 128-разрядные адреса получателей и отправителей (это в 4 раза больше, чем в 4-ой версии). Адрес в формате IPv6 может выглядеть так:

```
3A3F:BC21:F133:56C4:A103:DB11:1000:400F
```

Заголовок IPv6-пакета разработан таким образом, чтобы минимизировать содержащуюся в нем информацию. Поля параметров и поля, которые не являются необходимыми, вынесены за пределы заголовка.

Протокол IPv6 подробно описан в RFC 1883, а IPv4 — в RFC 791.

8.1.4. Установка xinetd

Суперсервер **xinetd** является достойной заменой **inetd**. Этот суперсервер, помимо всего прочего, обладает встроенными механизмами защиты, которые для **inetd** выполняет специальный демон **tcpd.** К тому же **xinetd,** в отличии от **inetd,** поддерживает IPv6. Даже если вы пока не планируете переходить на IPv6, установка **xinetd** будет очень полезной из-за расширенных функций суперсервера. Сам **xinetd** появился в Red Hat начиная с 7-ой версии и обычно устанавливается во время установки системы. Если у вас он еще не установлен, сейчас самое время это сделать.

При этом рекомендую пойти по пути наименьшего сопротивления и установить **xinetd** из RPM-пакета, а можно и выкачать из Интернет последнюю версию **xinetd** по адресу **http://www.synack.net/xinetd** и установить его из исходных кодов.

После того, как вы распакуете исходники, введите **./configure**, перейдя предварительно в каталог с исходниками (при этом желательно иметь права root). Для сценария **configure** вы можете использовать параметры, представленные в табл. 8.1.

Параметры сценария configure *Таблица 8.1*

Параметр	Описание
--with-libwrap	Демон будет использовать tcp wrappers. При этом у вас уже должен быть установлен libwrap. С этой опцией xinetd будет сперва проверять ваши /etc/hosts.allow и /etc/hosts.deny файлы, и только после этого запускает свой механизм контроля доступа
--with-loadavg	Компилирует xinetd с опцией max_load. Этот параметр остановит сервис, когда нагрузка достигнет определенного уровня
--with-inet6	Включает поддержку IPv6

Внимание!

При включении IPv6 все IPv4-сокеты становятся IPv6-сокетами и соответственно ядро (это прежде всего) и все программы должны поддерживать IPv6. Поэтому будьте готовы к тому, что вам понадобится перекомпилировать свое ядро с поддержкой IPv6. Если ваше ядро не поддерживает IPv6, выкачайте последнюю версию ядра по адресу ***http://www.kernel.org.*** *Тем не менее, и после включения IPv6 запросы IPv4 также будут обрабатываться. Так что никакой аварийной ситуации не будет.*

Возвращаясь к настройке: теперь введите **make,** а затем — **make install.**

Файл конфигурации **xinetd** — `/etc/xinetd.conf` отличается по синтаксису от файла конфигурации **inetd.** Но если **inetd у** вас уже настроен и вам лень настраивать все заново, воспользуйтесь программой **itox:**

```
itox < /etc/inetd.conf > /etc/xinetd.conf
```

Такое использование команды **itox** верно при условии, что файлом конфигурации **inetd** является файл `/etc/inetd.conf`, а **xinetd** — `/etc/xinetd.conf`. Но в любом случае, вам не помешает разобраться с форматом `xinetd.conf`.

После установки суперсервера из RPM-пакета или его сборки из исходных текстов, **xinetd** (или **inetd)** добавляется в сценарий автозафузки системы. Напомню, что включить или отключить запуск **xinetd (inetd)** или любого другого сервиса всегда можно с помощью конфигуратора **drakxservices** в Linux Mandrake или **setup** в Linux Red Hat (см. рис. 8.1).

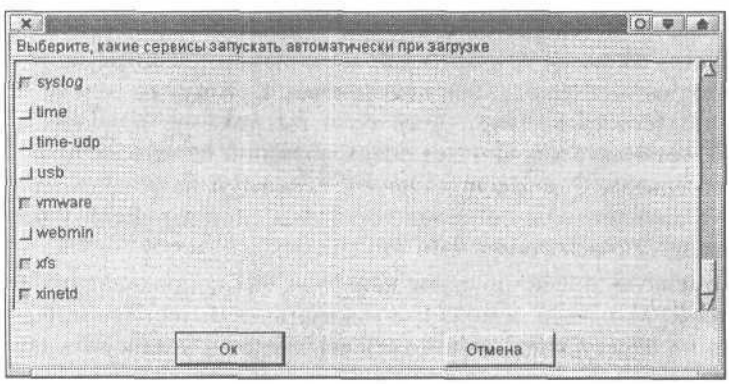


*Рис. **8.1.** Автозапуск суперсервера*

8.1.5. Настройка xinetd

Синтаксис файла xinetd.conf такой:

```
service <service_name>
    {
        <атрибут> <оператор_присваивания> <значение> <значение> ...
        <атрибут> <оператор_присваивания> <значение> <значение> ...
        ...
        <атрибут> <оператор_присваивания> <значение> <значение> ...
    }
```

Service_name -- это имя сервиса (login, shell, telnet, ftp, pop3 и т.д). Оператор присваивания может быть одним из следующих: «=», «+=», «-=». Большинство атрибутов может работать только с оператором «=» (равно). Назначение операторов следующее:

«=» ..... присвоить значение атрибуту;

«+=» ..... добавить еще одно значение;

«-=» ..... удалить значение.

Атрибут может иметь несколько значений, указанных через пробел. Некоторые параметры очень похожи на параметры **inetd.** Список всех атрибутов приведен в табл. 8.2.

Атрибуты сервиса для сервера xinetd *Таблица 8.2*

Атрибут	Описание
Id	Используется, если сервисы используют разные протоколы. Обычно совпадает с именем сервиса
Type	Может быть использована любая комбинация из следующих значений: RPC — если это сервис RPC (Remote Procedure Call). UNLISTED — если сервис не описан в файле /etc/rpc для rpc-сервисов или *в* /etc/services для не rpc. INTERNAL — если xinetd представляет этот сервис (для echo, time, daytime, chargen, и discard). Если RPC-сервисы у вас работают некорректно после установки xinetd, вы можете использовать для них старый inetd — inetd и его новая версия xinetd прекрасно уживаются вместе. В файле /etc/inetd.conf оставьте только RPC-сервисы, а остальное закомментируйте, а в /etc/xinetd.conf — наоборот
Flags	В качестве значения может быть использована любая комбинация из следующих значений: NODELAY — для tcp-сервиса будет установлен флаг сокета — TCP_NODELAY. Только для TCP-сервисов! DISABLE — отключить сервис. KEEPALIVE — установка флага сокета SO_KEEPALIVE. Только для TCP-сервисов! REUSE — установить флаг SO_REUSEADDR на сокет сервера. INTERCEPT — перехватывать пакеты или принимать соединения по порядку, проверяя, что они приходят из нужных мест. NORETRY — избегать повторных попыток в случае неудачи. IDONLY - соединение будет приниматься только от идентифицированных пользователей. На удаленной машине должен работать identification-сервер
disabled	Может принимать 2 значения, «yes» и «по». Если указать «yes», сервис запускаться не будет
socket_type	Тип сокета. Может принимать следующие значения: stream — сокет stream, обычно используется службами, работающими на основе протокола TCP dgram — сокет dgram, обычно используется службами, работающими на основе протокола UDP raw — сокет raw для сервисов, требующих прямого доступа к IP seqpacket — сокет seqpacket для сервисов, требующих надежную последовательную пересылку дейтаграмм

Продолжение табл. 8.2

Атрибут	Описание
protocol	Задает протокол, по которому будет работать сервер (tcp, udp, ..)
wait	Задает статус ожидания и может принимать два значения: yes и no, которые соответствуют значениям wait и nowait для сервера inetd (см. выше). Значение yes обычно устанавливается на сокетах dgram, а значение no на сокетах stream
user	Задает пользователя, от имени которого будет запущен сервер. Пользователь должен быть определен в файле /etc/passwd. По умолчанию сервер запускается от имени пользователя root
server	Указывает абсолютный путь к запускаемому серверу
server_args	Определяет аргументы, которые будут переданы серверу
log_on_failure	Определяет, какая информация будет писаться в файл отчета (протокол), если сервис по каким-либо причинам не запустился: HOST — записывать адрес удаленного хоста. USERID — если возможно, записывать идентификатор удаленного пользователя (используется протокол идентификации RFC 1413). ATTEMPT — записывать факт неудачной попытки. RECORD — записывать информацию с удаленного хоста, в случае невозможности запуска сервера
log_on_ success	Определяет, какая информация будет писаться в файл отчета (протокол) в случае удачного запуска сервиса. Можно комбинировать любые из следующих значений: PID — записывать идентификатор запущенного серверного процесса. HOST - записывать адрес удаленного хоста. USERID — если возможно, идентификатор удаленного пользователя (используется протокол идентификации RFC 1413) EXIT — записывать, каким образом был произведен выход. DURATION - - записывать продолжительность сессии
rpc_number	Определяет номер сервиса RPC
rpc_version	Определяет версию сервиса RPC
env	Задает значение атрибута. Атрибут представляет собой список строк типа: «name=value». Эти переменные будут добавлены в окружение перед тем, как сервер будет запущен
passenv	Это список переменных окружения из окружения xinetd, которые могут быть переданы серверу
port	Определяет порт сервиса. Если порт указан в файле /etc/services, то значение данного параметра должно совпадать с ним
redirect	Позволяет tcp-сервису делать перенаправление на другой хост. Значение задается в виде host:port
interface	Устанавливает интерфейс, на котором будет работать сервис. Синтаксис: interface=IP-адрес
bind	Это синоним параметра interface
banner	Определяет имя файла, который будет показываться при соединении с сервисом
banner_succe ss	Определяет имя файла, который будет показываться при удачном соединении
banner_fail	Определяет имя файла, который будет показываться при неудачном соединении
cps	Атрибут имеет два аргумента. Первый устанавливает количество соединений в секунду. Если это число будет превышено, сервис будет временно недоступен. Второй — число секунд, после которых сервис снова будет доступен
max_load	Определяет максимальную загрузку. При достижении максимума, сервер перестает принимать запросы на соединение. Значение параметра — число типа float
instances	Устанавливает число серверов, которые могут быть активны одновременно для сервиса (по умолчанию лимита нет). Значением этого атрибута может быть число, либо — UNLIMITED
nice	Устанавливает приоритет сервиса

Примечание.

RPC (Remote Procedure Call) — вызов удаленной процедуры. Используется в серверной части приложения. Механизм RPC скрывает от программиста детали сетевых протоколов нижележащих уровней.

Вам необязательно указывать все эти атрибуты для каждого сервиса. Можно указать только необходимые:

1. socket_type
2. user
3. server
4. wait

Параметр **protocol** указывается только для RPC-сервисов, а также для всех сервисов, которые не описаны в /etc/services. Параметр **rpc_version** — только для RPC-сервисов. Параметр **rpc_number** указывается только для RPC-сервисов, которые не указаны в файле /etc/rpc. Параметр **port** задается только для Не-RPC-сервисов, которые не описаны в /etc/services. Следующие атрибуты поддерживают все операторы присваивания:

1. only_from
2. no_access
3. log_on_success
4. log_on_failure
5. passenv
6. env (не поддерживает оператор «-=»)

Эти атрибуты также могут принимать разные значения в разных секциях описания сервиса.

Файл конфигурации может содержать секцию **default,** в которой описаны атрибуты по умолчанию. Они будут одинаковы для всех сервисов. Возможные атрибуты по умолчанию:

1. log_type
2. log_on_success
3. log_on_failure
4. only_from
5. no_access
6. passenv
7. instances
8. disabled
9. enabled

8.1.6. Параметры запуска xinetd

Я надеюсь, что с настройкой более-менее все понятно. Если же мои надежды не оправдались, то в разделе 8.1.7 вы найдете пример файла /etc/xinetd.conf. Сейчас же займемся запуском только что откомпилированного и настроенного суперсервера. А запускать его можно с параметрами, указанными в табл. 8.3.

В табл. 8.3 я привел описание не всех параметров запуска, выбрав лишь самые нужные. Более подробную информацию вы сможете получить в документации по **xinetd.** Так же **как и inetd, xinetd** можно контролировать с помощью сигналов (см. табл. 8.4).

Параметры запуска xinetd *Таблица 8.3*

Параметр	Описание
-f файл	Устанавливает альтернативный файл конфигурации, который должен использоваться вместо стандартного файла /etc/xinetd.conf
-pidfile pid_файл	Файл с ID-процесса
-stayalive	Даже если ни один сервис не прописан, демон должен выполняться («остаться в живых»)
-loop число	Задает количество коннектов в секунду
-d	Режим отладки (debug mode)
-reuse	Перед тем как связать сокет сервиса с IP-адресом, суперсервер установит опцию сокета SO_REUSEADDR
-limit число	Ограничение на количество одновременно запущенных процессов

Сигналы суперсервера *Таблица 8.4*

Сигнал	Описание
SIGUSR1	Суперсервер перечитает файл конфигурации
SIGQUIT	Остановит xinetd
SIGTERM	Перед остановкой xinetd все процессы будут остановлены

8.1.7. Пример файла конфигурации /etc/xinetd

Теперь, как и обещал, привожу пример файла конфигурации (см. листинг 8.5). В этом листинге перечислены наиболее часто используемые сервисы с оптимальными параметрами (атрибутами). Конечно же, вам предстоит решить: какие сервисы вы будете использовать, а какие нет. Возможно, вы также измените и их атрибуты, например, время работы сервиса.

Листинг 8.5. Фрагмент файла конфигурации /etc/xinetd:

```
# Параметры по умолчанию для всех возможных сервисов
defaults
{
# Число серверов, которые могут быть активны одновременно для
сервиса.
    instances    = 25
# Параметры протоколирования
    log_type     = FILE /var/log/servicelog
    log_on_success    = HOST PID
    log_on_failure    = HOST RECORD
    only_from        = 111.11.111.0 111.111.112.0
    only_from         = localhost 192.168.1.0/32
    disabled          = tftp
}
service login
{
    flags             = REUSE
    socket_type       = stream
    protocol          = tcp
    wait              = no
    user              = root
    server            = /usr/etc/in.rlogind
    log_type     = SYSLOG local4 info
}
```

```
# Сервис telnet — эмуляция  терминала удаленных систем
# (для 127. 0.0.1)
service telnet
{
    flags             = REUSE
    socket_type       = stream
    wait              = no
    user              = root
    server            = /usr/etc/in.telnetd
    bind              = 127.0.0.1
    log_on_failure    += USERID
}
# Сервис telnet — эмуляция терминала удаленных систем
service telnet
{
    flags             = REUSE
    disabled          = yes
    socket_type       = stream
    wait              = no
    user              = root
#   server            = /usr/etc/in.telnetd
    bind              =  192.231.139.175
    redirect          =  128.138.202.20  23
    log_on_failure    += USERID
}
service ftp
{
    socket_type       = stream
    wait              = по
    user              = root
    server            = /usr/etc/in.ftpd
    server_args       = -1
    instances         = 4
    log_on_success    += DURATION USERID
    log_on_failure    += USERID
t  Время работы сервиса
    access_times      = 2:00-8:59  12:00-23:59
f  приоритет
    nice              = 10
}
service name
{
    socket_type       = dgram
    wait              = yes
    user              = root
    server            = /usr/etc/in.tnamed
}

# Поддержка протокола TFTP  (Trivial FTP).  Этот протокол
# используется для обмена информацией между интеллектуальными
# маршрутизаторами и,  скорее всего,  вы не будете его
# использовать.
```

```
service tftp
{
    socket_type       = dgram
    wait              = yes
    user              = root
    server            = /usr/etc/in.tftpd
    server_args       = -s /tftpboot
}
# SMTP-сервис Qmail. Конфигурируется для запуска по требованию
# суперсервера xientd
service smtp
{
    socket_type       = stream
    protocol          = tcp
    wait              = no
    user              = qmaild
    id                = smtp
    server            = /var/qmail/bin/tcp-env
    server_args       = /var/qmail/bin/qmail-smtpd
    log_on_success    -= DURATION USERID PID HOST EXIT
    log_on_failure    -= USERID HOST ATTEMPT RECORD .
}
# Сервис finger, позволяющий узнать полезную общедоступную
# информацию о пользователях системы. Например, для того,
# чтобы узнать информацию о пользователе root системы host.com,
# введите одноименную команду: finger root@host.com
# Для работы этой команды необходим сервис finger.
#

service finger
{
    socket_type       = stream
    disabled          = yes
    wait              = no
    user              = nobody
    server            = /usr/etc/in.fingerd
}
service echo
{
    type              = INTERNAL
    id                = echo-stream
    socket_type       = stream
    protocol          = tcp
    user              = root
    wait              = no
}
service echo
{
    type              = INTERNAL
    id                = echo-dgram
    socket_type       = dgram
    protocol          = udp
```

```
    user            = root
    wait            = yes
}
service rstatd
{
    type            = RPC
    disabled        = no
    flags           = INTERCEPT
    rpc_version     = 2-4
    socket_type     = dgram
    protocol        = udp
    server          = /usr/etc/rpc.rstatd
    wait            = yes
    user            = root
}
```

Как видно из примера, я описал лишь те сервисы, которые больше всего необходимы. Доступ к сервисам в рассматриваемом примере могут получать только из сети 111.111.111.0, 111.111.112.0 и 192.168.1.0. Можно также указывать адрес и маску подсети, например 192.168.1.0/32. Вместо **sendmail я** использовал **qmail**. Можно было бы запускать **qmail** в режиме **standalone,** но так как я не очень часто пользуюсь услугами 25-го порта, то мне удобнее запускать сервис **smtp** через **xinetd**. Из соображений безопасности я отключил некоторые сервисы: **finger, telnet.**

8.2. Удаленный доступ: ssh и telnet

Сервис **Telnet** обеспечивает базовую эмуляцию терминалов удаленных систем, поддерживающих протокол **Telnet** над протоколом TCP/IP. Обеспечивается эмуляция терминалов Digital Equipment Corporation VT 100, Digital Equipment Corporation VT 52, TTY. Протокол Telnet описан в документе RFC 854, который вы найдете на прилагаемом компакт-диске.

Любые команды, выполняемые с помощью **Telnet,** обрабатываются telnet-сервером, а не локальным компьютером. Пользователь лишь видит результат выполнения этих команд.

Для использования **Telnet** на удаленном компьютере должен быть установлен telnet-демон. На компьютере пользователя нужно установить программу-клиент. Практически в каждой операционной системе существует утилита **telnet,** которая является клиентом для протокола telnet (см. рис. 8.2).

Сервис Telnet был и остается одним из самых популярных способов удаленной регистрации и работы на удаленной машине. Основным его недостатком является то, что любая информация, в том числе и пароли, передается в открытом виде без какого-либо кодирования.

SSH (Secure Shell) — программа, позволяющая вам зарегистрироваться на удаленных компьютерах и установить зашифрованное соединение. Существует также «безопасная» версия telnet — stelnet.

SSH использует криптографию открытого ключа для шифрования соединения между двумя машинами, а также для опознавания пользователей.

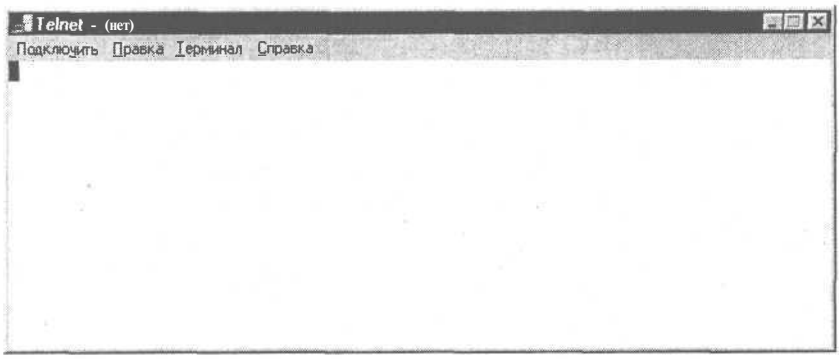


Рис. 8.2. *Telnet-клиент для Windows*

Оболочку ssh можно использовать для безопасной регистрации на удаленном сервере или копировании данных между двумя машинами, в то же время предотвращая атаки путем присоединения посередине (session hijacking) и обманом сервера имен (DNS spoffing).

Оболочка **Secure Shell** поддерживает следующие алгоритмы шифрования:

BlowFish — это 64-разрядная схема шифрования. Этот алгоритм часто используется для высокоскоростного шифрования данных больших объемов.

Тройной DES (Data Encryption Standard) — стандарт для шифрования данных. Данный алгоритм довольно старый, поэтому не рекомендуется его использовать. Обычно DES используется для шифрования несекретных данных.

IDEA (International Data Encryption Algorithm) — международный алгоритм шифрования информации. Этот алгоритм работает со 128-разрядным ключом и поэтому он более защищен, чем BlowFish и DES.

RSA (Rivest-Shamir-Adelman algorithm) — алгоритм Ривеста-Шамира-Адельмана. Представляет собой схему шифрования с открытым и секретным ключами.

При выборе алгоритма шифрования нужно исходить из конфиденциальности информации, которую вам нужно передать. Если информация секретна, лучше использовать алгоритмы **IDEA** или **RSA**. Если же вы просто не хотите передавать данные в открытом виде, используйте алгоритм **BlowFish,** поскольку он работает значительно быстрее, чем **DES.**

Оболочка ssh очень эффективна против анализаторов протоколов, так как она не только шифрует, но и сжимает трафик перед его передачей на удаленный компьютер. Программу ssh можно скачать по адресу **http://www.cs.hut.fi/ssh/.** Версия ssh для UNIX распространяется бесплатно, а за Windows-версию (имеется в виду клиент для Windows) нужно заплатить.

Оболочка ssh незаменима в тех случаях, когда удаленно нужно администрировать сервер или когда сервер не имеет собственного монитора. При использовании **telnet** все данные, которые передаются через telnet-соединение, доступны в открытом виде. А значит, имена пользователей и пароли будут доступны всем, кто прослушивает трафик с помощью анализатора. Шифрование ssh выполняет, используя несколько различных алгоритмов, включая **DES** и **3DES.**

Программа состоит из демона **sshd,** который запускается на Linux/UNIX-машине, и клиента **ssh,** который распространяется как для Linux, так и для Windows. Чтобы установить ssh, возьмите исходные тексты и поместите их по традиции в каталог /usr/src/. Затем распакуйте архив и установите программу, выполнив следующую последовательность действий:

```
cd /usr/src/
tar xzf ssh-2.4.0.tar.gz
cd ssh-2.4.0
./configure
make
make install
```

Чтобы **ssh** начал работать, необходимо запустить демон **sshd** на той машине, к которой предполагается подключение. Желательно добавить команду запуска в сценарий загрузки системы для автоматического запуска. Демон **sshd** работает по 22 порту (см. листинг 8.6). Если не ошибаюсь, **ssh** невозможно использовать вместе с **xinetd/inetd** — его нужно запускать подобно httpd-серверу в режиме **standalone.**

***Листинг 8.6. Фрагмент файла** /etc/services*

```
ssh        22/tcp      # SSH Remote Login Protocol
ssh        22/udp      # SSH Remote Login Protocol
```

Обычно с настройкой **sshd** не возникает никаких неприятных моментов. Подробно настройка демона будет рассмотрена чуть ниже в этой главе. Теперь попробуйте зарегистрироваться на этой машине через **ssh.** Для этого нужно установить этот же пакет на другую машину под управлением Linux/UNIX (или установить Windows-клиент **ssh)** и ввести команду:

```
$ ssh hostname.domain
```

ssh запросит вас ввести пароль пользователя. В качестве имени пользователя для установки соединения будет использовано имя текущего пользователя, то есть имя, под которым вы сейчас зарегистрированы в системе. В случае, если аутентификация пройдет успешно, начнется сеанс связи. Прекратить сеанс можно комбинацией клавиш Ctrl+D.

Если вам нужно указать другое имя пользователя, используйте параметр -l программы **ssh:**

```
ssh -l user hostname.ru
```

Так можно указать программе **ssh,** от имени какого пользователя нужно регистрироваться на удаленной машине (см. рис. 8.3).

При использовании Windows-клиента имя компьютера, имя пользователя и пароль нужно ввести в диалоговом окне программы. Если соединение не устанавливается, попробуйте выбрать метод кодирования **blowfish.** Если и это не поможет, выберите **3DES.**

Работа в ssh аналогична работе в **telnet.** Вы можете администрировать удаленную машину также легко, как и локальную. Опции программы ssh указаны в табл. 8.5.

Рис. 8.3. *Регистрация на удаленной машине*

Опции программы ssh *Таблица 8.5*

Опция	Описание
-a	Отключает перенаправление аутентификации агента соединения
-A	Включает перенаправление аутентификации агента соединения
-c blowfish|3des	Позволяет выбрать алгоритм шифрования при использовании первой версии протокола SSH. Можно указать или blowfish, или 3des
-c шифр	Задает список шифров, разделенных запятыми в порядке предпочтения. Только для второй версии протокола SSH. Допускаются значения blowfish, twofish, arcfour, cast, des и 3des
-f	Данная опция переводит ssh в фоновый режим после аутентификации пользователя. Рекомендуется использовать для запуска программы X1 1 . Например, ssh -f host xterm
-i идент_файл	Задает нестандартный идентификационный файл (для нестандартной RSA/DSA-аутентификации)
-l имя_пользоват	Указывает от имени какого пользователя будет осуществляться регистрация на удаленной машине
-p порт	Определяет порт, к которому подключится программа ssh (по умолчанию используется порт 22)
-q	«Тихий режим». Будут отображаться только сообщения о фатальных ошибках. Все прочие предупреждающие сообщения в стандартный выходной поток выводиться не будут
-x	Отключить перенаправление X1 1
-X	Включить перенаправление X1 1
-1	Использовать только первую версию протокола SSH
-2	Использовать только вторую версию протокола SSH
-4	Разрешается использовать IP-адреса только в формате IPv4
-6	Разрешается использовать IP-адреса только в формате IPv6

Оболочка **ssh** использует два файла конфигурации ssh_conf и sshd_conf. Думаю, что нет смысла говорить о том, что они находятся в директории /etc/ssh. Рекомендую в файле sshd_conf прописать следующую строчку:

```
allowedadress 10.1.1.1 10.1.2.1 10.1.3.1
```

Это означает, что доступ по **ssh** может быть выполнен только с машин с адресами 10.1.1.1, 10.1.2.1, 10.1.3.1. Это оградит ваш компьютер от нежелательных вторжений извне.

Программа **stelnet** во всем полностью аналогична программе **telnet,** но она выполняет шифрование трафика, который передается во время telnet-соединения.

Демон **sshd** -- **это** программа-демон для оболочки ssh. Обычно **sshd** запускается на машине, к которой подключаются клиенты SSH. Последние версии демона **sshd** поддерживают две версии протокола SSH — SSH версия 1, и SSH версия 2.

Протокол SSH версия 1

У каждого узла есть свой **RSA-ключ** (обычно 1024 бит), который используется для идентификации узла. Этот ключ еще называется открытым. Дополнительно, при запуске демона, генерируется еще один RSA-ключ — ключ сервера (обычно 768 бит). Этот ключ создается заново каждый час и никогда не сохраняется на диске.

Каждый раз при установке соединения с клиентом демон отправляет ему в ответ свой открытый ключ и ключ сервера. Клиент сравнивает полученный открытый ключ со своей базой данных, чтобы проверить, не изменился ли он. Затем клиент случайным образом генерирует 256-разрядное число и кодирует его, используя одновременно два ключа — открытый ключ и ключ сервера. Обе стороны используют этот случайный номер как ключ сессии, который используется для кодирования всех передаваемых во время сессии данных.

Затем клиент пытается аутентифицировать себя, используя **.rhosts-аутентификацию, аутентификацию RSA** или же **аутентификацию с использованием пароля.**

Обычно .rhosts-аутентификация небезопасна и поэтому она отключена.

Протокол SSH версия 2

Версия 2 работает аналогично: каждый узел имеет определенный **DSA-ключ**, который используется для идентификации узла. Однако, при запуске демона ключ сервера не генерируется. Безопасность соединения обеспечивается благодаря соглашению Диффи-Хелмана (Diffie-Hellman key agreement).

Сессия может кодироваться следующими методами: 128-разрядный AES, Blowfish, 3DES, CAST128, Arcfour, 192-разрядный AES или 256-разрядный AES.

Опции демона **sshd** указаны в табл. 8.6.

Опции демона sshd *Таблица 8.6*

Опция	Описание
-Ь биты	Определяет число битов для ключа сервера (по умолчанию 768). Данную опцию можно использовать, только если вы используете протокол SSH версии 1
-d	Режим отладки (DEBUG). В этом режиме сервер не переходит в фоновый режим и подробно протоколирует свои действия в системном журнале. Использование данной опции особенно полезно при изучении работы сервера
-e	Если указана это опция, демон sshd отправляет отладочные сообщения не в системный журнал, а на стандартный поток ошибок
-f конфиг_файл	Задает альтернативный файл конфигурации. По умолчанию используется /etc/ssh/sshd_config
-д время	Предоставляет клиенту, не прошедшему аутентификацию, дополнительное время, чтобы аутентифицировать себя. По умолчанию время равно 600 секундам. Если за это время клиент не смог аутентифицировать себя, соединение будет прекращено. Значение 0 интерпретируется как бесконечное ожидание
-h файл_ключа	Задает альтернативный файл открытого ключа (ключ узла). По умолчанию используется файл /etc/ssh/ssh_host_key. Эта опция может понадобиться, чтобы sshd мог выполняться не только от имени суперпользователя root. Кроме этого, частым использованием этой опции является запуск sshd из сценариев, задающих различные настройки в зависимости от времени суток. Например, в дневное (рабочее) время устанавливаются одни опции, а в вечернее(иерабочее) время — другие

Продолжение табл. 8.6

Опция	Описание
-i	Используется, если нужно запускать sshd через суперсервер xinetd (inetd). Обычно демон sshd не запускается суперсервером xinetd (inetd), а запускается при загрузке системы, потому что демону sshd требуется некоторое время (10 секунд) для генерирования ключа сервера, прежде чем он сможет ответить на запросы клиентов
-k время	Задает время, спустя которое ключ сервера будет создан заново. По умолчанию время составляет 3600 секунд (1 час). Данную опцию можно использовать, только если вы используете протокол SSH версии 1
-p порт	Указывает альтернативный порт, который демон sshd будет прослушивать. По умолчанию используется порт 22
-q	«Тихий режим». В данном режиме протоколирование сессии производиться не будет. Обычно протоколируется начало аутентификации, результат аутентификации и время окончания сессии
-t	Тестовый режим. Данный режим применяется для проверки корректности файла конфигурации
-D	При использовании этой опции демон не будет переходить в фоновый режим
-4	Разрешается использовать IP-адреса только в формате IPv4
-6	Разрешается использовать IP-адреса только в формате IPv6

Файл конфигурации демона /etc/ssh/sshd_config выглядит примерно так, как это покзано в листинге 8.7

Листинг 8.7 Файл конфигурации /etc/ssh/sshd_config

```
# $OpenBSD: sshd_config,v 1.38 2001/04/15 21:41:29 deraadt Exp $
# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin
f This is the sshd server system-wide configuration file.  See sshd(8)
# for more information.

Port 22
#Protocol 2,1
#ListenAddress 0.0.0.0
#ListenAddress ::
HostKey /etc/ssh/ssh_host_key
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
#
# Don't read ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# Uncomment if you don't trust ~/.ssh/known_hosts for
RhostsRSAAuthentication
#IgnoreUserKnownHosts yes
StrictModes yes
XllForwarding yes
XllDisplayOffset 10
```

```
PrintMotd yes
#PrintLastLog no
KeepAlive yes

# Logging
SyslogFacility AUTHPRIV
LogLevel  INFO
#obsoletes QuietMode and FascistLogging

RhostsAuthentication  no
#
# For this to work you will also need host keys in /etc/ssh/
ssh_known_hosts
RhostsRSAAuthentication no
I similar for protocol version 2
HostbasedAuthentication no
i
RSAAuthentication yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
PermitEmptyPasswords no
# Uncomment to disable s/key passwords
#ChallengeResponseAuthentication no
# Uncomment to enable PAM keyboard-interactive authentication
# Warning: enabling this may bypass the setting of 'PasswordAuthentication'
#PAMAuthenticationViaKbdInt yes

# To change Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#AFSTokenPassing no
#KerberosTicketCleanup no

# Kerberos TGT Passing does only work with the AFS kaserver
#KerberosTgtPassing yes

#CheckMail yes
#UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net
#ReverseMappingCheck yes

Subsystem  sftp/usr/libexec/openssh/sftp-server
```

В большинстве случаев вас должны устроить опции по умолчанию, однако сейчас мы все же рассмотрим некоторые из них.

Директива **Port** предназначена для указания порта, которые демон будет прослушивать (данная директива аналогична опции -p.):

```
Port 22
```

Следующая директива — это директива **Protocol.** С помощью этой директивы можно указать в порядке предпочтения номера поддерживаемых протоколов SSH:

```
Protocol 2,1
```

Такое определение директивы означает, что сначала сервер будет пытаться установить соединение с клиентом по протоколу SSH версии 2, а потом — по протоколу SSH версии 1. Можно указать использование только одной версии протокола, например, **Protocol 1.**

Директива **ListenAddress** указывает локальный адрес, который должен прослушивать демон.

Директива определяет файлы ключей. Файлами по умолчанию являются:

```
/etc/ssh/ssh_host_key
/etc/ssh/ssh_host_rsa_key
/etc/ssh/ssh_host_dsa_key
```

Директива **ServerKeyBits** определяет разрядность ключа сервера для протокола SSH первой версии. По умолчанию используется 768-разрядный ключ (768 бит).

Директива **LoginGraceTime** аналогична опции -g: предоставляет клиенту дополнительное время, чтобы аутентифицировать себя. По умолчанию время равно 600 секундам. Если за это время клиент не смог аутентифицировать себя, соединение будет прекращено.

Директива **KeyRegenerationInterval** аналогична опции -k. Она определяет время, спустя которое ключ сервера будет создан заново. По умолчанию время составляет 3600 секунд (1 час).

Директива **PermitRootLogin** определяет, разрешено ли пользователю root регистрироваться по ssh. Значение по умолчанию:

```
PermitRootLogin yes
```

Еще две директивы, имеющие непосредственное отношение к аутентификации — это **PasswordAuthentication и PermitEmptyPasswords.** Первая разрешает (при значении yes) аутентификацию с помощью пароля, а вторая — разрешает (при значении yes) использовать пустые пароли. Значения по умолчанию:

```
PasswordAuthentication  yes
PermitEmptyPasswords no
```

Описание остальных опций вы найдете в справочной системе, введя команду **man sshd.**

8.3. Маршрутизация

Маршрутизацию между сетями можно организовать с помощью команды **route** или с помощью **IpChains.** Сейчас рассмотрим более или менее подробно первый случай, а о втором поговорим в гл. 14.

Примечание.

IPChains *— это средство фильтрации пакетов. Фильтр просматривает заголовок пакета и решает, что делать со всем пакетом. Например, можно указать фильтру, что определенные пакеты должны быть удалены, а некоторые перенаправлены, то есть обеспечить маршрутизацию.*

Пусть, у вас есть две сетевые платы eth0 и eth1:

```
ifconfig ethO 192.168.1.1 up
ifconfig ethO 192.168.2.1 up
```

и вам нужно обеспечить маршрутизацию между подсетями 192.168.1.0 и 192.168.2.0. С этой целью объявляем, что машины, которые находятся в вашем локальном сегменте 192.168.1.*, «сидят» на первом интерфейсе и общаться с ними нужно напрямую:

```
route add net 192.168.1.0 192.168.1.1 netmask 255.255.255.0 O
```

А с машинами с адресами 192.168.2.* будем разговаривать через ethl:

```
route add net 192.168.2.0 192.168.2.1 netmask 255.255.255.0 0
```

Последний параметр — это метрика. Ее можно понимать как «расстояние до шлюза-назначения» или «сколько пересадок между шлюзами придется сделать пакету по пути и обратно». Т.к. адреса 192.168.1.1 и 192.168.2.1 являются нашими собственными адресами, то метрика равна 0.

Сетевые пакеты для IP-адресов, которые не лежат в нашей локальной сети, будем отправлять на машину 192.168.1.11, а она сама будет разбираться, что с ними делать:

```
route add default 192.168.1.11 1
```

Другими словами, сейчас мы объявили маршрут по умолчанию. Обратите внимание на значение метрики = 1. Как видите, мы все сделали без всяких конфигураторов — все просто и логично.

Сейчас постараемся, как говорится, рассмотреть второй вариант в трех строчках. Допустим, у вас есть те же две подсети — 192.169.1.0 и 192.168.2.0. Постараемся организовать маршрутизацию средствами **IpChains:**

```
ipchains -P forward DENY
ipchains -A forward -s 192.168.1.0/24 -d 192.168.2.0/24 -j ACCEPT
ipchains -A forward -s 192.168.2.0/24 -d 192.168.1.0/24 -j
ACCEPT
```

О том, что означают данные три строчки, вы узнаете в гл. 14.

8.4. Настройка DHCP (Dynamic Host Configuration Protocol)

Для чего нужен протокол DHCP? DHCP — это протокол настройки узла, который автоматически назначает IP-адреса компьютерам. По сути, протокол DHCP — это дальнейшее развитие протокола **BOOTP**. Последний разрешает бездисковым клиентам запускать и автоматически конфигурировать протокол TCP/IP. Протокол DHCP централизовано назначает IP-адреса в вашей сети и автоматически конфигурирует рабочие станции. Возможно, вы подумали, что в одной сети должен быть только один сервер DHCP,

потому что в противном случае между серверами возникнет конфликт, а пострадавшим опять окажется клиент, который зависнет при загрузке. А вот и не так — в одной сети может быть несколько серверов DHCP. И это не только не отразится на производительности сети, но даже повысит надежность сети, если, например, один из серверов выйдет из строя.

Итак, установите пакет **dhcp** и включите поддержку динамических IP-адресов командой **echo "1" > /proc/sys/net/ipv4/ip_dynaddr.** DHCP в Linux реализован в виде демона сервера (dhcpd) и демона клиента **(dhcpcd).** Демон сервера непосредственно отвечает за назначение IP-адресов клиентам, при входе и выходе их из сети. Клиентский демон, как явствует из названия, запускается на стороне клиента.

Конфигурационным файлом для **dhcpd** является `/etc/dhcp.conf`. При запуске DHCP-сервера происходит выделение IP-адресов согласно содержащимся в файле `/etc/dhcp.conf` установкам. Выделенные адреса **dhcpd** регистрирует в файле `dhcpd.leases`, который обычно находится в каталоге `/var/dhcpd`.

Сейчас давайте рассмотрим простейшую конфигурацию, которую будем постепенно наращивать (см. листинг 8.8). Обратите внимание на то что, чтобы внесенные вами в файл `/etc/dhcp.conf` изменения вступили в силу, демон **dhcpd** необходимо остановить и запустить снова. При этом используйте команду **/etc/rc.d/init.d/dhcpd stop** для останова демона, и команду **/etc/rc.d/init.d/dhcpd start** для его запуска.

Листинг 8,8. Файл /etc/dhcpd.conf

```
I описание сети, указывающее, какая из подсетей будет
# обслуживаться. Указывается сетевой адрес и маска сети
subnet 192.168.1.0 netmask 255.255.255.0 {
# маршрутизатор по умолчанию
option routers 192.168.1.1;
# маска подсети 255.255.255.0
option subnet-mask 255.255.255.0;
# установка домена по умолчанию и сервера NIS, если таковой используется
option nis-domain "domain.ua";
option domain-name "domain.ua";
f адрес DNS-сервера, который будут использовать клиенты
option domain-name-servers 192.168.1.1;
# диапазон адресов для клиентов
# 192.168.1.50-192.168.1.250
range 192.168.1.10 192.168.1.254;
# сказать клиентам, чтобы отдали адрес через 21600 секунд (6 часов)
# после получения адреса
default-lease-time 21600;
# забрать адрес самому через 28800 секунд (8 часов)
max-lease-time 28800;
}
```

Теперь будем постепенно усложнять конфигурацию. Каждая сетевая карточка имеет уникальный собственный MAC-адрес. Допустим, вам нужно связать какой-то MAC-адрес с определенным IP-адресом. Для этого воспользуйтесь конструкцией **host:**

```
host myhost {
hardware ethernet xx:xx:xx:xx:xx:xx;
fixed-address 192.168.1.9;
}
```

Ее нужно вставить в ту конструкцию подсети subnet, которой принадлежит назначаемый IP-адрес. Данная конструкция означает, что компьютеру с аппаратным адресом xx:xx:xx:xx:xx:xx будет назначен IP-адрес 192.168.1.9. Например:

```
subnet 192.168.1.0 netmask 255.255.255.0 {
    # прочие опции
    # ...
    #
    host myhost {
        hardware ethernet 00:40:C7:34:90:1E;
        fixed-address 192.168.1.9;
            }
}
```

Данный пример показывает, что аппаратному адресу 00:40:C7:34:90:1E будет сопоставлен IP-адрес 192.168.1.9. Обратите внимание, что IP-адрес хоста myhost 192.168.1.9 относится к подсети 192.168.1.0 и включен в инструкцию subnet подсети 192.168.1.0, а не какой-либо другой сети!

Существует довольно удобная утилита для просмотра всех MAC-адресов сетевых адаптеров в вашей сети — программа **TCPNetView**. Эта программа разработана Александром Горлачем и загрузить ее вы можете по адресу **http://www.enet.ru/~gorlach/netview/.** Правда, есть одно «но»: эта программа работает под Windows. В любом случае, если вы будете использовать эту программу, при настройке сервера вам не придется подходить к каждому компьютеру, чтобы узнать его MAC-адрес. Существуют также и утилиты под Linux, способные показать сразу все MAC-адреса. Примером может послужить программа **Trafshow**, которую вы найдете на прилагаемом компакт-диске. Но по сравнению с Trafshow, TCPNetView несколько удобнее (см. рис. 8.4).

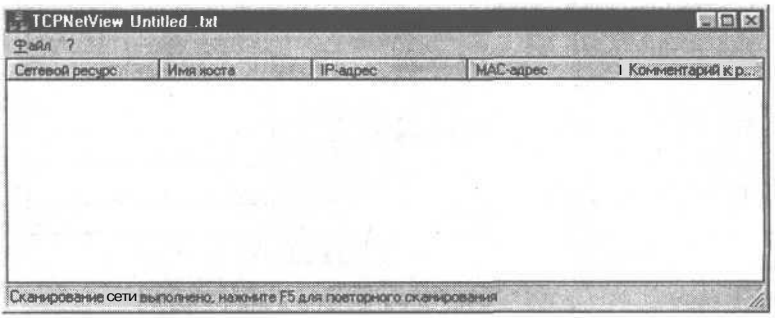


Рис. 8.4. *Программа TCPNetView*

Теперь, предположим, что вам необходимо обеспечить поддержку **WINS,** а на вашей машине установлен сервер **Samba.** В этом случае в конструкцию **subnet** нужно включить следующие директивы:

```
option netbios-name-servers 192.168.1.1;
option netbios-dd-server 192.168.1.1;
option netbios-node-type 8;
```

Примечание.

*Служба **WINS** (Windows Internet Name Service) используется для разрешения (перевода) имен **NetBIOS** в IP-адреса. Сервер **WINS** — это усовершенствованный сервер имен **NetBIOS,** разработан Microsoft для снижения широковещательного трафика. Пакет **Samba** предназначен для использования протокола **SMB** (Server Message Block), который также еще называется протоколом **NetBIOS.** С помощью пакета **Samba** ваша машина, работающая под управлением Linux, ничем не будет отличаться от рабочей станции или сервера сети Microsoft.*

Вот практически и все. Правда, еще можно добавить пару незначительных опций:

```
# определяем широковещательный адрес
option broadcast-address 192.168.2.255;
# включаем IP-Forwarding
option ip-forwarding on;
t можно добавить глобальную опцию:
server-identifier server.domain.ua;
```

Как обычно, дополнительную информацию можно получить, введя команду **man dhcpd.conf.** При настройке клиентов Windows следует активизировать режим «Получить IP-адрес автоматически» в свойствах TCP/IP (рис. 8.5). А при настройке Linux с помощью конфигуратора **netconf** — включить режим DHCP (см. рис. 8.6).

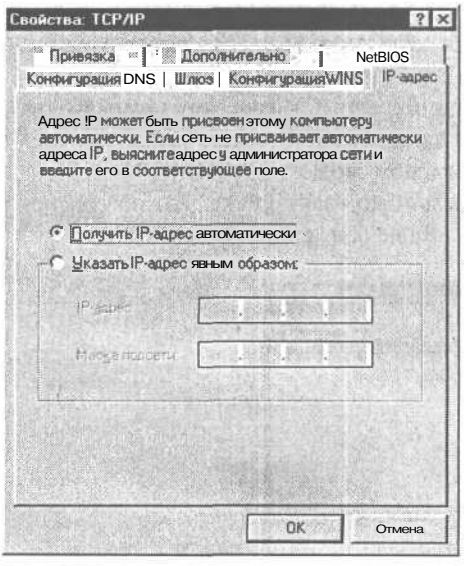


Рис. ***8.5.*** *Настройка Windows-клиента*

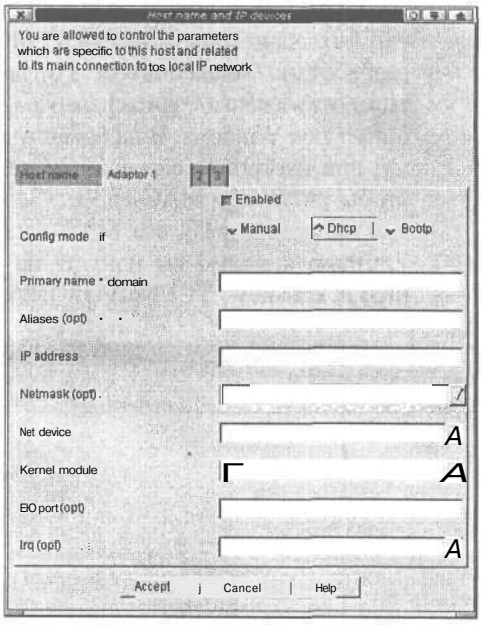


Рис. 8.6. *Настройка Linux-клиента*

Протокол DHCP подробно описан в RFC 1533, 1534, 1541, 1542, а протокол BOOTP описан в RFC 1532. Окончательный вариант конфигурационного файла приведен в листинге 8.9.

Листинг 8.9. Конфигурационный файл /etc/dhcpd.conf (окончательный вариант)

```
I Подсеть 192.168.1.0, маска сети 255.255.255.0
subnet 192.168.1.0 netmask 255.255.255.0 {

# маршрутизатор по умолчанию
option routers 192.168.1.1;

# маска подсети 255.255.255.0
option subnet-mask 255.255.255.0;

# устанаввка домена по умолчанию и сервера NIS, если таковой используется
option nis-domain "domain.ua";
option domain-name "domain.ua";

# задание широковещательного адреса
option broadcast-address 192.168.2.255;

# включение IP-Forwarding
option ip-forwarding on;

# глобальная опция server-identifier:
server-identifier server.domain.ua;

# адрес DNS-сервера, который будут использовать клиенты
option domain-name-servers 192.168.1.1;

f диапазон адресов для клиентов
t 192.168.1.50-192.168.1.250
range 192.168.1.10 192.168.1.254;

# сказать клиентам, чтобы отдали адрес через 21600 секунд (6 часов)
# после получения адреса
default-lease-time 21600; '

t забрать адрес самому через 28800 секунд (8 часов)
max-lease-time 28800;
option netbios-name-servers 192.168.1.1;
option netbios-dd-server 192.168.1.1;
option netbios-node-type 8;

# описание трех клиентов (dhcp50, dhcp51, dhcp52)
# и их аппаратных адресов
      host dhcp50 {
          hardware ethernet 00:40:C7:34:90:1E;
# обратите внимание на то, что вы должны использовать IP-адрес
# из указанного ранее диапазона адресов 192.168.1.50-250.
          fixed-address 192.168.1.50;
                    }
      host dhcp51 {
          hardware ethernet 00:40:C7:34:90:1F;
          fixed-address 192.168.1.51;
                    }
```

```
    host dhcp52 {
        hardware ethernet 00:40:C7:34:90:2A;
        fixed-address 192.168.1.52;
                }
}
```

8.5. Подсчет трафика. Программа MRTG

Эта тема является одной из наиболее интересных. Хочу сразу заметить, что существует такое множество способов считать трафик, что можно было бы написать не одну книгу, рассматривая все из них.

Самым простым способом является подсчет с помощью программы **ifconfig**. Чтобы понять как все работает, введите команду:

```
cat    /proc/net/dev
```

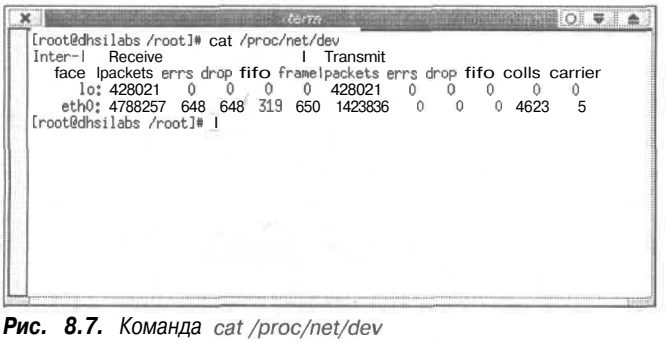


Рис. 8.7. *Команда cat /proc/net/dev*

В результате вы увидите строки, изображенные на рис. 8.7. Для наглядности я ввел эту команду в ОС Linux со старым ядром, так как в новом ядре появились новые опции учета и они не умещаются в окне терминала, а при переносе строк теряется наглядность примера.

Файл /proc/net/dev содержит информацию о работе сетевых устройств. На этом и основывается данный метод подсчета трафика. Для самого же подсчета удобнее использовать нижеприведенный сценарий **stat:**

```
#!/bin/sh
/bin/grep "$1" /proc/net/dev | /bin/awk -F ":" '{ print $2 }' | \
/bin/awk '{ print "In:  " $1 "\nOut: " $9 ;}'
```

Символ \ используется как перенос в обычном тексте. Его обычно используют для повышения удобочитаемости сценариев. Интерпретатором он будет воспринят как пустой символ.

Скопируйте данный сценарий в каталог /usr/bin и сделайте его выполнимым:

```
cp ./stat /usr/bin
chmod 755 /bin/stat
```

Использовать данный сценарий можно так:

```
/bin/stat  ethO
```

где ethO.....это нужный вам интерфейс.

Можно было бы добавить проверку на правильность указания сценариев, но это не касается самого подсчета трафика. Попробуйте пропинговать интерфейс ethO по его IP-адресу и снова выполните сценарий.

Честно говоря, это самый простой способ и, скорее всего, для вас он окажется совершенно бесполезным. Я привел его лишь в демонстрационных целях — это как своеобразная программка "Hello, world!".

Теперь перейдем к более традиционному методу. В данном методе для подсчета трафика используется **IpChains.** Чтобы глубже понимать, о чем пойдет речь, я рекомендую прочитать сначала гл. 14, а потом вернуться к этой главе. Думаю, читатель меня простит за это неудобство -- так уж получилось. Впрочем, я постараюсь объяснить все как можно подробнее и в этом разделе, во всяком случае, все должно быть понятно, а за разъяснениями опций **ipchains** читатель может обратиться к гл. 14.

Предлагаемый мною способ является не самым лучшим, но с его помощью я надеюсь подтолкнуть читателя на создание своего фундаментального продукта для подсчета трафика. Для того чтобы данный способ работал, необходимо включить опцию **IP: accounting** в конфигураторе ядра и перекомпилировать его. В большинстве случаев эта опция уже включена. Затем нужно установить **IpChains.** Когда все будет готово, установите следующее правило **IpChains:**

```
ipchains -A output -d AAA.AAA.AAA.AAA -j ACCEPT
```

Данное правило нужно установить для каждого адреса, учет которого вы хотите вести. А просмотреть статистику можно с помощью команды:

```
ipchains -L -v
Chain input (policy ACCEPT: 4195746 packets, 1765818402 bytes):
Chain forward (policy ACCEPT: 142999 packets, 29941516 bytes):
Chain output (policy ACCEPT: 4182597 packets, 1309541595 bytes):
pkts bytes target  prot opt  tosa tosx ifname  source      destination
4    308   ACCEPT  all        0xFF 0x00 any     anywhere    AAA.AAA.AAA.AAA
```

Как вы заметили, это не полный листинг. Я выбрал основное, пропустив поля **mark, outsize и ports,** которые не представляют для нас интереса.

Здесь видно, что клиент AAA.AAA.AAA.AAA получил 308 байт или 4 пакета. В данном случае в качестве пакетов вступали пакеты протокола ICMP. Правило установлено таким образом, что учет ведется по всем интерфейсам (ifname=any), из любого источника, то есть адреса (source=anywhere) и учитываются все протоколы (prot=all). Аналогично можно установить правило для учета данных, полученных от клиента. Только в этом случае необходимо будет использовать цепочку **input.** Вам нужно установить это правило, потому что обычно считается не только исходящий, но и входящий трафик клиента, если ваш сервер, например, выступает в роли шлюза.

Можно также использовать данные, взятые из аппаратного маршрутизатора **Cisco.** Кстати, через **Cisco** работает популярная программа **tacacs+.** Эта программа используется для учета времени работы пользователей в системе. И именно эта программа используется рядом провайдеров при организации **dialin-доступа.** Программа доступна по адресу **ftp.vsu.ru/pub/hardware/cisco /tacacs/tac+ia-0.96pre9.3.tar.gz**

Очень полезна также программа **useripacct**. Она позволяет узнать о трафике каждого пользователя.

Считать трафик можно также программой **trafshow**, которую вы найдете на прилагаемом компакт-диске. Эта программа считает только локальный трафик (сколько байтов принял и передал данный компьютер и от кого он принял и кому передал). Поэтому, установив ее на шлюзе, можно вполне контролировать трафик всей сети (см. рис. 8.8). Кроме того, немного изменив исходный код **trafshow,** можно заставить ее отображать не только IP-адреса или доменные имена компьютеров вашей сети, но и MAC-адреса. Исходный код этой программки вы также найдете на компакт-диске.

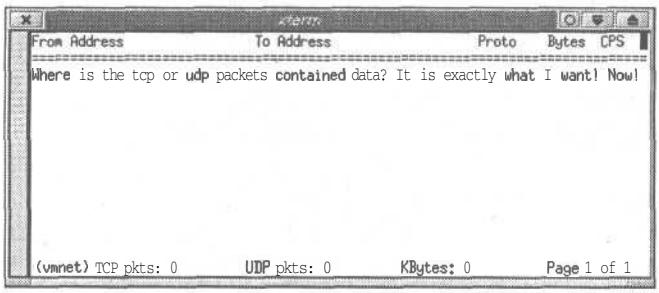


Рис. 8.8. *Программа trafshow*

Как я уже писал, можно привести сотни способов учета трафика, а пользоваться вы все равно будете одним. Причем, в большинстве случаев, это будет способ, который вы «изобретете» сами, написав сценарий для обработки данных статистики.

К другим способам учета трафика можно отнести учет с использованием протокола **SNMP.** Именно по этому протоколу работает программа **MRTG (http://www.switch.ch/misc/leinen/snmp/perl/).** Программа **MRTG** предоставляет очень удобные средства для подсчета трафика: подсчет для всей сети и для отдельного узла, генерирование отчетов и диаграмм в формате HTML и многое другое.

Программа **MRTG** (The Multi Router Traffic Grapher) предназначена для мониторинга загрузки канала за сутки, неделю, месяц и год. Программа MRTG умеет рисовать красивые картинки в формате PNG, которые отображают состояние канала за определенный период времени.

Пример использования вы можете увидеть на сайте:

http://www.stat.ee.ethz.ch/mrtg/

Для работы **mrtg** нам потребуется маршрутизатор, поддерживающий протокол SNMP. В этой же главе будет рассмотрен пример, позволяющий обойтись без маршрутизатора и вообще не использовать протокол SNMP. Общая конфигурация сети должна выглядеть примерно как на рис 8.9.

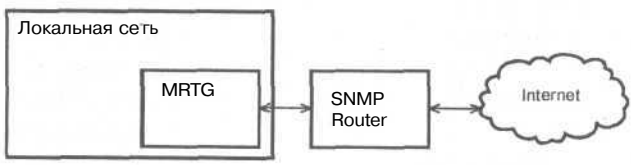


Рис. 8.9. *Конфигурация сети*

Из рисунка видно, что наша сеть получает доступ к Internet через SNMP-маршрутизатор. Компьютер MRTG — это узел локальной сети, на котором установлена программа MRTG. Программа MRTG будет периодически запускаться на узле MRTG, обновляя информацию о трафике. Пользователи локальной сети могут ознакомиться с этой информацией по протоколу HTTP. Естественно, на узле MRTG должен быть установлен Web-сервер.

Перед установкой программы убедитесь в наличии следующих библиотек:

1. gd (http://www.boutell.com/gd/);
2. **libpng** (http://www.libpng.org/pub/png/);
3. **zlib** (http://www.info-zip.org/pub/infozip/zlib/).

Загрузить последнюю версию MRTG можно по адресу:
http://ee-staff.ethz.ch/~oetiker/webtools/mrtg/pub

Если вы используете операционную систему RedHat версии 7 или выше, программа MRTG, скорее всего, будет уже у вас установлена. Мы не будем скачивать исходные тексты программы, а сразу воспользуемся уже собранным пакетом **rpm**. Установим **mrtg** командой:

```
rpm -ih mrtg*
```

После установки нужно подготовить программу к первому запуску, то есть указать, откуда получать сведения о трафике.

Программа MRTG состоит из трех частей:

1. **cfgmaker** ..... утилита для создания конфигурационного файла.
2. **indexmaker** ... утилита для создания файла `index.html` — страницы краткого обзора, дающую вам общее представление о всех целях, которые вы контролируете. О целях мы поговорим немного позже.
3. **mrtg** ..... **сам mrtg.**

Первый конфигурационный файл удобно создать с помощью программы **cfgmaker,** а потом добавить в него дополнительные параметры.

Введите команду:

```
cfgmaker –global 'WorkDir: /var/www/html/mrtg'  \
    –global 'Options[_]: bits,growright' \
    –output /var/www/html/mrtg/mrtg.cfg     \
    community@router
```

Теперь разберемся, что все это означает. Прежде всего, напомню, что означает наклонная черта: это просто перенос строки. Когда вы будете вводить команду, вместо наклонной черты используйте пробел как разделитель параметров. Наклонная черта используется, когда команда не помещается в командной строке.

Параметр **WorkDir** задает рабочий каталог. В этот каталог будут помещены html-файлы и рисунки — отчеты о трафике. Каталог `/var/www/html/`, как вы уже успели заметить, является корневым каталогом нашего Web-сервера, поэтому для просмотра статистики нужно ввести следующий URL в окне браузера: **http://host/mrtg/**

Кроме параметра **WorkDir** имеются также параметры **HtmlDir** и **ImageDir**. В эти каталоги будут помещены html-файлы и картинки. При определении этих параметров нужно учитывать, что параметр **WorkDir** имеет приоритет над параметрами **HtmlDir и ImageDir** и поэтому, если указан параметр **WorkDir, mrtg** поместит отчеты и картинки именно в этот каталог, а значения параметров **HtmlDir и ImageDir** будут проигнорированы.

Группа глобальных параметров **Options** управляет построением изображения. Параметр **bits** означает, что мы измеряем трафик в битах, поэтому все числа нужно умножить на 8. Второй параметр, **growright**, указывает направление оси времени. Позже мы рассмотрим все параметры подробнее.

Параметр **output** программы **cfgmaker** задает имя конфигурационного файла, который будет создан.

Примечание.

*Параметры **WorkDir и Options** являются параметрами программы mrtg, а параметры **global и output** — параметрами программы **cfgmaker.***

Параметр **community@router** указывает имя сообщества SNMP-устройства. В нашем случае — это наш SNMP-маршрутизатор. Обычно используется имя сообщества **public.** За более точной информацией обратитесь к администратору SNMP-маршрутизатора. Вместо имени узла в этом параметре можно указать IP-адрес, например, public@192.168.1.1.

Параметр **community@router** как раз и является целью, которую мы будем контролировать.

После выполнения этой команды будет сгенерирован такой файл **mrtg.cfg**:

```
WorkDir: /var/www/html/mrtg
Options[_]: bits,growright
Target[r1]: community@router
```

Имя цели (rl) пишется в квадратных скобках. Для разных целей можно задавать разные параметры, например:

```
Target[r1]: 1:community@router
Target[r2]: 2:community@router
MaxBytes[r1]: 1250000
MaxBytes[r2]: 2500000
Title[r1]: Traffic Analysis for first interface
PageTop[r1]: <H1>Stats for our interface #1</H1>
Title[r2]: Traffic Analysis for second interface
PageTop[r2]: <H1>Stats for our interface #2</H1>
```

Параметр **MaxBytes** определяет максимальное число байт для цели. Значения, превышающие MaxBytes, будут игнорироваться.

Параметр **Title** определяет заголовок страницы, которая будет содержать информацию о цели, а параметр **PageTop** — текст, который будет помещен в верхней части этой страницы.

Числа 1 и 2 перед именем сообщества — это номера интерфейсов во внутренней таблице SNMP-устройства. После имени (или IP-адреса)

SNMP-устройства можно указать порт SNMP. По умолчанию используется стандартный порт 161:

```
community@router:161
```

В качестве цели может быть использована программа, которая выводит на стандартный вывод четыре строки:

1. Количество принятых байтов.
2. Количество отправленных байтов.
3. Время работы объекта после включения.
4. Имя объекта.

Программу нужно записать в обратных кавычках, например:

```
Target[r3]: `/usr/bin/program`
```

Очень полезными параметрами являются **Refresh и Interval.** Первый определяет частоту обновления страниц с отчетами в браузере, а второй – предполагаемый интервал запуска программы **MRTG.** По умолчанию значения обоих параметров равно 300 секундам.

Опции **perminute и perhour** позволяют измерять трафик в единицах за минуту и час соответственно. Опция **noinfo** подавляет вывод информации об устройстве и времени его работы. Пример:

```
Options[_]: bits, perminute, noinfo
```

Я думаю, что теории вполне достаточно, тем более, что вместе с **MRTG** поставляется отличная документация, которая доступна по адресу **http://localhost/mrtg/**. Теперь перейдем к практической настройке. Скорее всего, у вас не будет SNMP-маршрутизатора, поскольку в небольших сетях маршрутизатором является сама Linux-машина, а выделение средств на приобретение аппаратного маршрутизатора в ближайшие несколько лет не предвидится. Да и намного интереснее считать трафик своего компьютера, а не какого-то маршрутизатора, которого вы даже и не видели и который установлен где-то на третьем этаже.

Для работы **MRTG** необходимо установить и настроить сервер **snmpd.** Однако в большинстве случаев этого делать не нужно: корректная настройка данного сервера — это довольно нетривиальная задача, а лишняя «дыра» в системе безопасности нам не нужна. К тому же настройка сервера **snmpd** оправдывает себя, если вы хотите считать трафик этого компьютера не локальной программой **mrtg**, а удаленной, которая запущена на другом компьютере и получает данные от нашего сервера по протоколу SNMP.

Я предлагаю довольно простое решение, настройка которого не займет у вас много времени. Основывается оно вот на чем: как я уже отмечал, вместо цели можно указать программу, которая бы выводила информацию на стандартный вывод в таком формате:

```
Строка 1
Строка 2
Строка 3
Строка 4
```

- **Строка 1** — это входящие байты (принятые).
- **Строка** 2 — исходящие байты (отправленные).

- **Строка 3** — время, на протяжении которого работает устройство.
- **Строка 4** — имя цели.

Где же взять эту программу? Написать самому! Сейчас я подробно опишу, как это сделать. Настоятельно не рекомендую вам сразу взять и использовать готовый листинг: вы не поймете самого главного — как именно происходит подсчет трафика. В результате ваша система подсчета трафика будет работать в таком режиме: программа будет считать трафик, а MRTG — строить графики.

Определим, откуда будем брать информацию о трафике. Операционная система Linux сама выполняет подсчет трафика. Вся информация, которая вам необходима, содержится в файле /proc/net/dev. Выполните команду:

```
cat /proc/net/dev
```

Результат выполнения этой команды вы уже видели в этой главе (рис. 8.7). Более новые ядра предоставляют больше информации о работе сетевых устройств, поэтому выполните данную команду для того, чтобы увидеть, какую информацию о сетевых устройствах предоставляет ваша система. Обычно первое информационное поле файла /proc/net/dev — это количество принятых байтов, а девятое — количество отправленных байтов.

Разрабатываемая программа должна найти нужный интерфейс и возвратить количество принятых и переданных байтов. Затем программа возвращает время, на протяжении которого работает устройство. Это время достаточно легко вычисляется с помощью программы **uptime.**

```
1:51pm up 2:10, 4 users, load average: 0.02, 0.04, 0.00
```

Программа **uptime,** кроме всякой другой информации, возвращает время, на протяжении которого система работает, то есть с момента загрузки операционной системы. В вышеприведенном примере видно, что машина непрерывно работала 2 часа и 10 минут. 2 часа и 10 минут — это значение, которое разрабатываемая программа должна вывести в третьей строке. Вы можете смело использовать это время, потому что в основном интерфейсы сервера «подымаются» при загрузке системы и разница между **uptime** системы и **uptime** интерфейса составит всего несколько секунд.

Четвертая строка — это имя цели, то есть имя интерфейса, трафик которого нам нужно подсчитать. Надеюсь, что алгоритм программы ясен, осталось реализовать все это программно.

Создайте файл count (см. листинг 8.10) и поместите его в каталог /usr/bin (не забудьте сделать его исполнимым!).

Листинг 8.10. Программа count

```
#!/bin/bash
/bin/grep "$1" /proc/net/dev | /bin/awk -F ":" '{ print $2 }' |
/bin/awk '{ print $1 "\n" $9 }'
UPTIME=`/usr/bin/uptime | /bin/awk -F " " '{ print $3 }'`
echo $UPTIME
echo $1
```

Использовать программу нужно так:

```
count интерфейс
```

Например, **count ethO.**

Запустив программу, вы должны увидеть примерно следующие строки:

```
2738410
1235960
2:57,
ethO
```

Во второй строке программы происходит следующее: находится нужная нам запись с именем интерфейса, который мы указали в первом параметре при вызове программы ($1). Затем интерпретатор awk выводит значения первого и девятого полей, содержашие количество принятых и переданных байтов.

В третьей строке программы вычисляется время **uptime.**

Последние две строки выводят время **uptime** и название интерфейса.

Предположим, что у вас имеется два интерфейса: локальный ethO и выделенная линия (pppO), идущая к провайдеру. При этом конфигурация сети несколько упростилась (см. рис. 8.10).

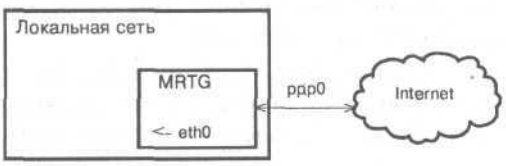


Рис. 8.10. *Конфигурация сети (2)*

Теперь узел MRTG сам является маршрутизатором и самостоятельно считает свой трафик. Файл конфигурации **mrtg** будет выглядеть так, как это показано в листинге 8.11.

Листинг 8.11. Файл /var/www/html/mrtg/mrtg.cfg

```
WorkDir: /var/www/html/mrtg/ipc
Options[_]: bits,growright
Target[eth0]: '/usr/bin/count eth0'
Title[eth0]: Local Ethernet
MaxBytes[eth0]: 99999999
PageTop[eth0]: Status of /dev/ethO
Target[ppp0]: Vusr/bin/count ppp0'
Title[ppp0]: Leased line
MaxBytes[ppp0]: 99999999
PageTop[ppp0]: Status of /dev/pppO
```

Из листинга 8.11 видно, что у вас имеются две цели, для каждой из них заданы свои параметры. Нужно учитывать, что имя интерфейса, которое вы передаете программе **count,** должно совпадать с названием цели (ethO и pppO).

В качестве рабочего каталога я использовал /var/www/html/mrtg/ipc. От использования каталога /var/www/html/mrtg/ я отказался, поскольку в нем находится документация по mrtg.

Параметры **MaxBytes, Title и PageTop** являются обязательными. При их отсутствии mrtg попросит вас исправить ошибки в конфигурационном файле.

Теперь можете запустить программу **mrtg** командой:

```
mrtg /var/www/html/mrtg/mrtg.cfg
```

В каталоге /var/www/html/mrtg/ipc должны появиться первые файлы-отчеты. Имя файла-отчета будет совпадать с именем цели. Первые два запуска mrtg будет «ругаться» на отсутствие предыдущих данных, но потом все будет работать как надо.

Если третий запуск прошел гладко, то есть без сообщений об ошибках, можно добавить **mrtg** в расписание демона **crond.** Для этого добавьте в файл /etc/crontab одну из следующих строк (какая кому нравится):

```
5,10,15,20,25,30,35,40,45,50,55,59 * * * * root /usr/bin/mrtg /
var/www/html/mrtg/mrtg.cfg
```

или

```
0-59/5 * * * * root /usr/bin/mrtg /var/www/html/mrtg/mrtg.cfg
```

После этого желательно перезапустить демон **crond:**

```
/etc/init.d/crond restart
```

Программу **mrtg** можно запускать в режиме демона (не через **crond**). Для этого установите значение параметра **RunAsDaemon** равное yes. За более подробной информацией обратитесь к документации по **mrtg.**

Теперь самое время проверить, как работает **mrtg.** Запустите браузер и введите адрес **http://localhost/mrtg/ipc/eth0.html**. В результате вы должны увидеть информацию о загрузке канала. Первые графики вы увидите примерно через час после первого запуска **mrtg,** в зависимости от настроек периода запуска **mrtg.**

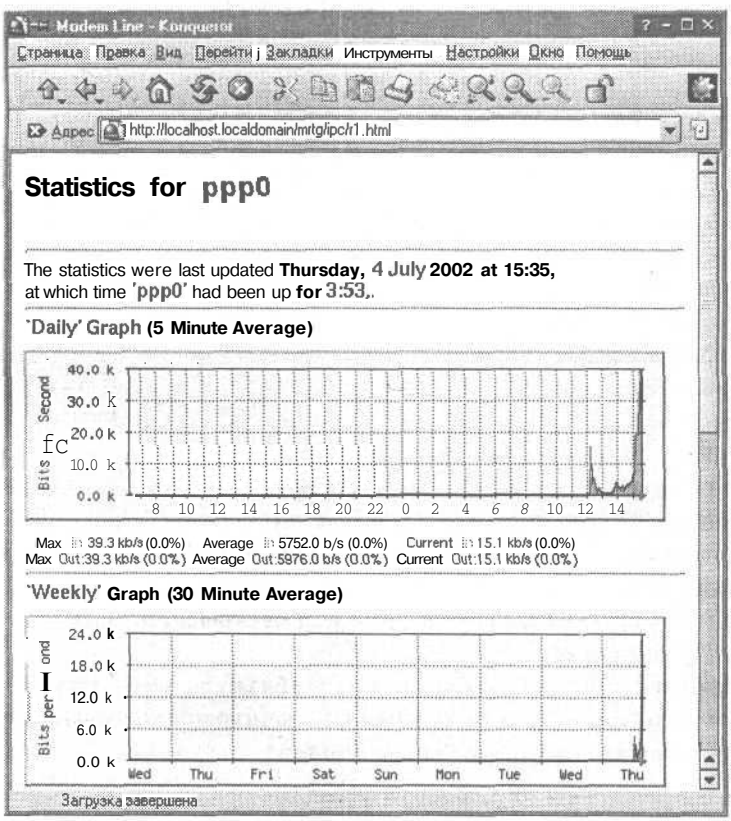


Рис. *8.11. Статистика для устройства pppO*

8.6. Сетевая файловая система (NFS)

Сетевая файловая система позволяет монтировать файловые системы на удаленных компьютерах. При этом создается ощущение, что эти файловые системы являются локальными, если не считать, конечно, скорости соединения.

После монтирования вы сможете непосредственно обращаться к файлам этой файловой системы. Сетевая файловая система чем-то напоминает службу «Доступ к файлам и принтерам» сети Microsoft. Для того, чтобы компьютер мог предоставлять свои ресурсы для сетевой файловой системы NFS, на нем должен быть установлен и настроен NFS-сервер. Для того, чтобы компьютер имел доступ к ресурсам сетевой файловой системы, на нем должен быть установлен и настроен NFS-клиент. И тот и другой можно установить на одном компьютере, если этот компьютер и предоставляет свои ресурсы системе NFS, и использует ресурсы NFS.

Для использования NFS нужно убедиться, что у вас запущены сервисы **netfs и nfslock,** а в некоторых системах **nfsd и mountd**. Это можно сделать с помощью конфигуратора или просмотреть наличие соответствующих ссылок в каталоге `/etc/init.d/rc.d`.

Возможно, у вас не установлена поддержка NFS. В этом случае, установите пакет `nfs-utils-0.2.1-2mdk.i586.rpm` на сервере, а пакет `nfs-utils-0.2.1-2mdk.i586.rpm` — на клиенте.

8.6.1. Настройка сервера NFS

Напомню, что если у вас не установлена поддержка NFS, то для сервера необходимо установить пакет `nfs-utils-0.2.1-2mdk.i586.rpm`.

Для настройки сервера сетевой файловой системы NFS используется файл конфигурации /etc/exports. В нем указываются файловые системы, которые будут экспортированы для совместного использования (см. листинг 8.12). Обычно в качестве таковых используются файловые системы **/home**, **/pub** и некоторые другие.

Листинг 8.12. Файл /etc/exports

```
/pub(ro,insecure,all_squash)
/home/den denhome.domain.com(rw)
/mnt/cdrom (ro)
/mnt/cdrom comp11.domain.com(noaccess)
```

Давайте разберемся в данной конфигурации. К каталогу /pub имеют доступ в режиме «только чтение» все компьютеры сети. К каталогу /home/den доступ в режиме чтения и записи имеется только с компьютера denhome.domain.com. К каталогу /mnt/cdrom доступ в режиме «только чтение» имеют все компьютеры, кроме comp11.domain.com.

При конфигурировании сетевой файловой системы могут использоваться опции, указанные в табл. 8.7.

Опции, задаваемые в файле /etc/exports *Таблица 8.7*

Опция	Описание
secure	Требует, чтобы запросы исходили из портов, принадлежащих только безопасному диапазону (с номерами < 1024). Данная опция включена по умолчанию
insecure	Отключает опцию secure
ro	Доступ в режиме "только чтение»
rw	Доступ в режиме чтения и записи
noaccess	Запрещает доступ к конкретной ветви экспортируемого дерева каталогов
link_absolute	Оставляет все символические ссылки без изменений. Включена по умолчанию
link_relative	Конвертирует абсолютные ссылки в относительные
squash_uidssquash_gids	Указанные идентификаторы групп и пользователей будут конвертированы в анонимные
all_squash	Все идентификаторы групп и пользователей будут конвертированы в анонимные. По умолчанию так не делается
no_all_squash	Обратна опции all_squash. Активизирована по умолчанию
root_squash	Преобразует запросы от пользователя root (uid=0) в запросы от анонимного пользователя. Благодаря этой опции, пользователь root не сможет пользоваться своими правами (правами пользователя root) при доступе к файловой системе. Данная опция установлена по умолчанию
no_root_squash	Отменяет опцию root_squash и позволяет пользователю root пользоваться своими правами (правами пользователя root) при доступе к сетевой файловой системе из клиентской системы
anonuid=UID anonguid=GID	Задают идентификаторы анонимных пользователей

8.6.2. Настройка клиента NFS

В предыдущем разделе было рассмотрено, как настроить сетевую файловую систему NFS. Теперь давайте рассмотрим как можно подмонтировать имеющуюся сетевую файловую систему к какому-либо клиенту. Напоминаю, что если на предполагаемом NFS-клиенте не установлена поддержка NFS, то на нем необходимо установить пакет nfs-utils-0.2.1-2mdk.i586.rpm.

Итак, вернемся к настройке NFS-клиента. Ниже приведен пример того, как можно подмонтировать к нему сетевую файловую систему. Монтирование осуществляется с помощью команды **mount:**

```
mount -t nfs -o timeo=30 nfsserver.domain.com:/home/den /home/den/remote/
```

Прежде всего, нужно указать тип файловой системы **-t** nfs. Параметр **timeo** задает время ожидания, равное 3 секундам. Интересующая нас файловая система находится на компьютере nfsserver.domain.com и смонтирована там как /home/den. Мы же подмонтируем ее к своему домашнему каталогу /home/den/remote/.

При монтировании сетевых файловых систем доступны опции, указанные в табл. 8.8.

Если на вашем компьютере запущен сервер DNS и сервер NFS, проследите за тем, чтобы сервер DNS запускался после запуска сервиса NFS. При соблюдении данного условия гарантируется корректная работа сервера NFS. Точно такое же замечание я сделал и в главе 10, посвященной DNS. Чтобы вы не подумали, что я повторяюсь, объясню: лучше дважды упомянуть, чем забыть или не обратить внимание.

Опции команды mount для сетевых файловых систем *Таблица 8.8*

Опция	Описание
bg	В том случае, если первая попытка монтирования файловой системы NFS окажется неудачной, она будет автоматически повторяться в фоновом режиме
fQ	В том случае, если первая попытка монтирования файловой системы NFS окажется неудачной, она будет автоматически повторяться в приоритетном режиме. Данная опция установлена по умолчанию
soft	Задержка при выполнении операции, связанной с файлом, расположенным в сетевой файловой системе NFS (возникает при сбое сервера или отключении сети), будет приводить к отправке приложению сообщения об ошибке ввода/вывода. И хотя некоторые приложения могут корректно обрабатывать такую ошибку, но большинство из них все-таки такой возможностью не обладают. Не рекомендуется использовать данную опцию, так как она может привести к появлению испорченных файлов и потере данных
hard	Задержка при выполнении операции, связанной с файлом, расположенном в сетевой файловой системе NFS, будет приводить к приостановке, а затем возобновлению процесса с прерванного места. Таким образом, будут предприниматься повторные попытки выполнения операции
tcp	Монтирует сетевую файловую систему с помощью протокола TCP, а не UDP
rsize=1024	Задает размер информации, пересылаемый при чтении файлов за один раз. По умолчанию этот размер равен 1024 байт
wsize=1024	Аналогично rsize, но для операции записи
noexec	Запрещает выполнение программ или сценариев в монтируемой файловой системе

Для того чтобы сетевая файловая система монтировалась автоматически при загрузке системы, нужно внести определенные записи в файл /etc/fstab. Например, такая запись для рассмотренного выше примера может иметь примерно следующий вид:

```
nfsserver.domain.com:/home/den   /home/den/remote/  nfs  bg,hard,rw 1  0
```

8.7. Поисковый сервер ht:/Dig

Сервер **Dig** предназначен для поиска и индексирования содержимого Web-страниц в небольших сетях. Сервер **Dig** прекрасно справляется с поиском информации на серверах вашей сети, однако заменить полноценную поисковую машину, такую, как Rambler, Yandex или Google, он не может. Этот поисковый сервер не очень масштабируемый и сможет охватить лишь несколько серверов вашей сети.

Сервер **Dig** предоставляет простые и сложные методы поиска информации. К сложным методам относятся логический *(boolean method)* и нечетко определенный метод поиска (*fuzzysearching method).* Нечетко определенный поиск включает в себя несколько алгоритмов: простой, зондирующий и поиск с использованием синонимов.

Поиск производится по HTML-документам и по простым текстовым документам. Документы HTML могут содержать ключевые слова, что упрощает поиск. Поиск ограничивается глубиной и локализацией. Можно идентифицировать пользователя при попытке поиска в определенных каталогах или вообще запретить поиск в указанных каталогах (ограничение локализацией).

Файл конфигурации `htdig.conf` сервера Dig находится в каталоге /etc/htdig. Директива **database_dir** определяет расположение базы данных

сервера **ht:/Dig.** Базы данных могут быть довольно большими, поэтому нужно позаботиться о том, чтобы хватило дискового пространства.

Директива **start_url** указывает начальные URL-адреса поиска. Сервер Dig будет производить индексирование, начиная с этих адресов. Вы можете указать несколько адресов.

Директива **limit_urls_to** определяет, какие адреса будут ограничены во время создания индекса. Обычно здесь нужно указать те URL-адреса, которые вы указали в директиве **start_url**.

Директива **exclude_urls** определяет, какие адреса не будут индексированы. Обычно не требует индексирования каталог `/cgi-bin/`, содержащий сценарии.

Директива **bad_extensions** запрещает индексирование файлов с указанным расширением.

Другие директивы позволяют установить максимальный размер заголовка документа HTML (**max_head_length**), максимальный размер файла (**max_doc_size**) и установить алгоритм поиска (**search_algoritm**), а с помощью директивы **allow_virtual_hosts** можно указать серверу индексировать виртуальные хосты как отдельные компьютеры.

В состав системы **Dig** входят пять программ: **htdig, htmerge, htfuzzy, htnotify и htsearch.** Поиск выполняет программа **htsearch,** программы **htdig, htmerge, htfuzzy** выполняют индексирование. Сначала программа **htdig** собирает информацию в локальной базе данных, а затем сопоставляет найденные Web-страницы с установленными вами критериями поиска. Программа **htmerge** использует информацию, предоставленную ей программой htdig, для создания поисковой базы данных. Программа **htfuzzy** создает индексы в базе данных, что позволяет использовать методы нечетко определенного поиска.

Довольно часто пользователи используют Web-страницы, которые вызывают программу **htsearch** для организации поиска. При этом программе **htsearch** передаются некоторые параметры: параметр поиска, конфигурация программы (**config**), метод поиска **(method)** и вид критерия (sort). При работе с этой программой можно использовать методы передачи данных GET и POST.

Для создания базы данных предназначен сценарий **rundig**.

8.8. Прокси-сервер Socks5

8.8.1. Установка и настройка сервера

Сервер **Socks5** — это универсальный прокси-сервер. Сервер **Socks5** требует поддержки протокола socks5 со стороны программного обеспечения клиента. При этом могут применяться специальные программные пакеты, позволяющие использовать стандартное программное обеспечение. Под специальным программным обеспечением подразумевается программа **runsocks**, входящая в состав сервера **socks5** или аналогичные ей программы.

В большинстве случаев прокси-сервер **Socks5** нужно использовать для того, чтобы обеспечить работу Socks-клиентов (обычно в этой роли выступает программа **ICQ)** через бастион. Довольно часто встречающаяся ситуа-

ция: программу **ICQ** запускает пользователь локальной сети, у которого нет реального Интернет-адреса и он подключается к Интернет через *firewall.* Для решения этой проблемы можно воспользоваться двумя методами: или использовать IP-маскарадинг, или же установить Socks5-сервер на шлюзе. Первый способ описан в гл. 14, а настройка второго рассматривается в этой главе. Существует также еще один метод, который заключается в использовании прокси-сервера **SQUID.** Для этого нужно добавить в список разрешенных портов порт 5190. Этот порт используется современными ICQ-клиентами:

```
acl SSL_ports port 5190
```

Однако, по сравнению со **SQUID,** прокси-сервер **Socks5** представляет собой наиболее гибкое решение. Да и зачем устанавливать на шлюзе полнофункциональный прокси-сервер, если нужно только обеспечить работу ICQ для пользователей внутренней сети?

Нужно отметить, что оба прокси-сервера (Socks5 и SQUID) могут быть установлены на одном сервере и одновременно функционировать, не мешая друг другу.

Клиентами сервера **socks5** являются популярные клиенты **ICQ** и **licq,** клиентская версия оболочки ssh, а также другие программы. Подробную информацию о сервере **socks5** вы можете найти по адресу: **http://www.socks.nec.com.** Некоторые вопросы по настройке **socks5** довольно хорошо рассмотрены в справочной системе. Прочитать стандартную документацию вы можете, введя команды:

```
man socks5.conf
man libsocks5.conf
```

В этой главе будет рассмотрена только базовая настройка сервера **socksS.**

Сначала нужно загрузить последнюю версию прокси-сервера **(http://www.socks.nec.com/cgi-bin/download.pl)** — на данный момент это v1.0 release 11. То есть вам нужно выкачать файл `socks5-v1.0r11`. Желательно также скачать **socks5tools** - - **в** нем находится сценарий для обработки протоколов сервера. После распаковки выполните привычную последовательность команд:

```
./configure
make
make install
```

При корректной сборке в каталоге /etc будет создан файл `socks5.conf`, в котором и содержатся все настройки сервера.

В большинстве случаев параметры по умолчанию являются вполне приемлемыми. Сейчас рассмотрим пример конфигурационного файла (см. листинг 8.13), а потом разберемся, что все это означает.

Листинг 8.13. Файл /etc/socks5.conf

```
set SOCKS5_NOREVERSEMAP
set SOCKS5_NOSERVICENAME
set SOCKS5_NOIDENT
```

```
set  SOCKS5_MAXCHILD  128
set  SOCKS5_TIMEOUT  10
auth — - u
permit u -- - - - - - -
interface 192.168.0. — eth0
```

В первой строке мы отменяем обратный резолвинг адресов, благодаря чему сервер будет работать заметно быстрее. Вторая строка означает, что мы будем протоколировать номера портов вместо имен сервисов. Теоретически это тоже должно повысить эффективность работы сервера. Параметр SOCKS5_NOIDENT запрещает рассылку клиентам **ident-запросов**. Четвертая строка устанавливает максимально допустимое число потомков сервера — не жадничайте. Пятая строка, как вы уже успели догадаться, устанавливает тайм-аут (10 секунд).

Практически вся настройка сервера выполняется с помощью манипулирования командами **auth и permit.** Первая устанавливает тип аутентификации, а вторая — разрешает доступ определенным хостам/пользователям. Полный формат команды **auth** такой:

```
auth исходный_хост  исходный_порт  метод_аутентификации
```

В данном случае мы будем запрашивать пароль со всех пользователей (точнее, клиентов).

Формат команды **permit:**

```
permit аутентификация команда исх_хост хост_назнач исх_порт
порт_назнач [список_польз]
```

В примере я разрешаю доступ всем и отовсюду с использованием аутентификации. Если вас интересует более расширенный пример использования команды **permit,** который демонстрирует всю гибкость этого прокси-сервера, обратите внимание на этот:

```
permit u cpubt 192.168. -• - [100,1000] user
```

В данном случае мы разрешаем доступ пользователю *user* (с использованием пароля, конечно). Пользователь *user* имеет право использовать Connect, Ping, Udp, Bind и Traceroute (cpubt) с адресов 192.168.*.*. Диапазон входящих (первый «-») и входящих (второй «-») портов — 100..1000.

Директива **interface** (листинг 8.13) разрешает все соединения от компьютеров с адресами 192.168.0.* (наша внутренняя сеть) ко всем портам интерфейса eth0. Кроме команды **permit** существует противоположная ей команда **deny** с аналогичными параметрами.

Создайте файл /etc/socks5 .passwd — в нем содержатся имена пользователей и их пароли. Например:

```
petrov 123456
ivanov paswd
```

За этим все! Вы уже настроили ваш сервер. Осталось его запустить:

```
# /usr/local/bin/socks5 -f -s
```

При этом демон должен перейти в фоновый режим и выводить диагностические сообщения на стандартный вывод (в нашем случае это экран).

Если сервер сконфигурирован правильно, вы должны увидеть примерно следующее:

```
11410: Socks5 starting at Mon Mar 4 19:13:55 2002 in normal mode
```

После удачного запуска остановите сервер и добавьте его в скрипты автозагрузки системы:

```
# killall socks5
```

8.8.2. Альтернативные серверы Socks5

В качестве альтернативы серверу **socks5** вы можете использовать прокси-сервер **dante-socks**, который доступен по адресу http://www.inet.no/dante/. Данный сервер использует файл конфигурации `sockd.conf` (см. листинг 8.14).

Листинг 8.14. Файл /etc/sockd.conf

```
internal: 192.168.0.1 port = 1080
external: 111.111.111.111
client pass {
    from: 192.168.0.0/16 to: 0.0.0.0/0
        }
pass    (
    from: 0.0.0.0/0 to: ,192.168.0.0/16
    command: bindreply udpreply
    log: connect error
        }
```

Параметр internal определяет ваш внутренний интерфейс (точнее, внутренний IP-адрес), а external — ваш настоящий IP (111.111.111.111.111). В блоке client pass указываются возможные клиенты вашего сервера (сеть 192.168.0.0), а в блоке pass указываются имена узлов, которые могут «общаться» с вашими клиентами. В приведенном примере разрешается отвечать клиентам со всех узлов (0.0.0.0). Протоколироваться будут только ошибки соединения.

8.8.3. Настройка клиента Socks5 (licq)

Настройку клиентов будем рассматривать на примере двух, наверное самых популярных, Socks-клиентов. Сначала рассмотрим настройку программы ICQ для Windows, а потом licq — ICQ-клиент для Linux.

Запустите программу ICQ и нажмите на кнопку «ICQ», которая находится ниже кнопки «Services» (рис. 8.12). Затем из появившегося меню выберите команду Preferences и перейдите в раздел Connections на вкладку Server. Включите режим использования прокси-сервера и установите тип прокси-сервера — **Socks5** (см. рис. 8.13).

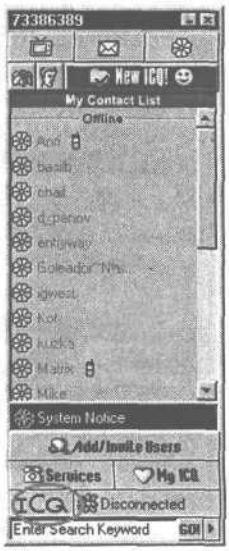


Рис. 8.12. Программа icq

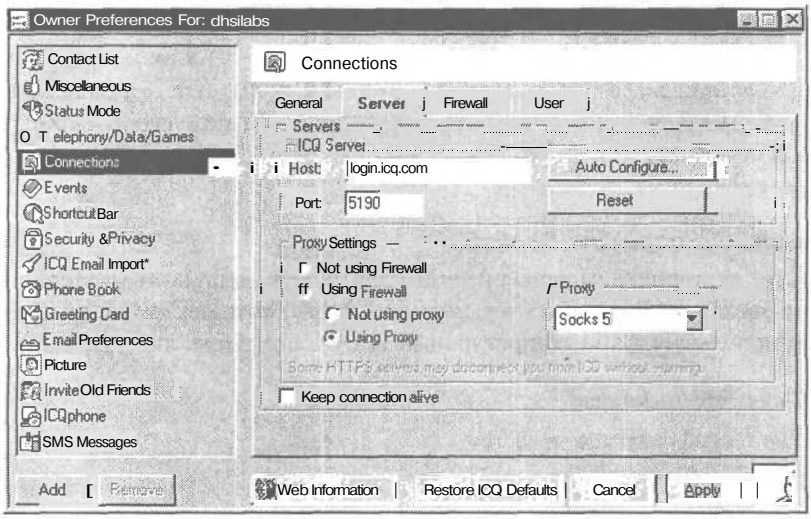


Рис. S. *13. Свойства соединения icq*

Потом перейдите на вкладку **Firewall** и установите параметры прокси-сервера: имя, порт, тип (socks5), имя пользователя и пароль (см. рис. 8.14). Нажмите на кнопку «Apply» и на этом настройка клиента ICQ для Windows завершена.

С программой **licq** будет немножко сложнее. Во-первых, нужно установить программу **runsocks,** которая входит в состав прокси-сервера (эту программу можно также найти в Интернет отдельно), на компьютере пользователя и перекомпилировать **licq,** включив поддержку **socks5**. Для этого перейдите в каталог, содержащий исходные тексты **licq** и запустите скрипт **configure** с параметром **—enable-socks5**:

```
./configure —enable-socks5
```

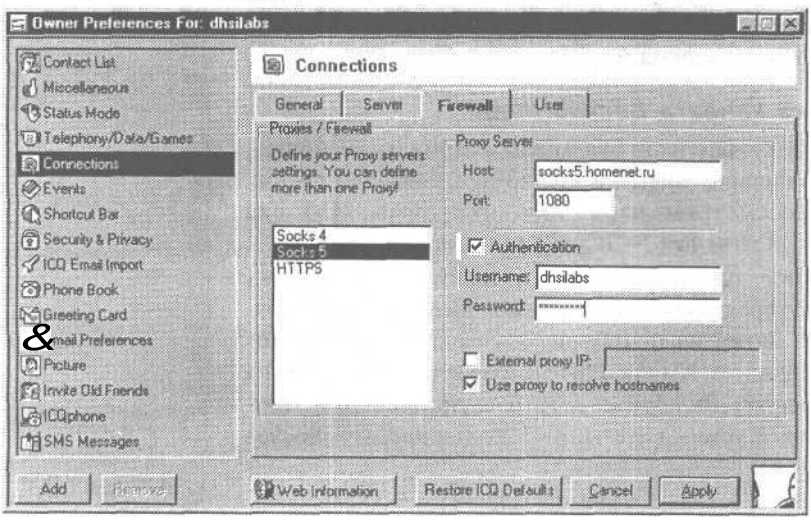


Рис. S. *14. Параметры прокси-сервера*

После этого выполните привычные команды:

```
make; make install
```

Теперь нужно создать файл /etc/libsocks5.conf и добавить в него строку:

```
socks5 -    - - 192.168.0.1:port
```

192.168.0.1 — это адрес вашего socks5-сервера, port — порт, необходимый клиенту (обычно 1080).

8.9. Система обнаружения и защиты от вторжения

8.9.1. Что такое LIDS?

LIDS (Linux Intrusion Detection System) — система обнаружения вторжения. Данная система представляет собой патч к ядру, который позволяет обнаружить и защитить вашу систему от вторжения.

Система **LIDS** была создана китайским и французским разработчиками Xie Huagang и Philippe Biondi и доступна на сайте **http://www.lids.org**. На этом сайте доступна как сама система **LIDS,** так и документация к ней. Однако разработчики системы не спешат обновлять документацию к новым версиям системы. А документация по версии 0.8.x ничем вам не поможет, если вы используете версию 0.9.x. Поэтому в этой главе я не буду рассматривать одну определенную версию, так как при выходе следующей версии этот материал безнадежно устареет, а рассмотрю общую настройку **LIDS** — все написанное в этой главе должно работать у вас вне зависимости от версии LIDS.

Система LIDS позволяет запретить или просто ограничить доступ пользователя root к файлам, сетевым интерфейсам, памяти, блочным устройствам. Почему именно пользователя root, я думаю, вы уже догадались: ограничить доступ любого пользователя можно с помощью стандартных средств самой Linux. В первую очередь система защищает систему от злоумышленника, который, воспользовавшись багом («дырой» или функцией на языке Microsoft) какой-нибудь программы, получил права root.

Основным преимуществом системы **LIDS** является то, что ее нельзя отключить обычными способами.

Во-первых, для отключения **LIDS** необходимо находиться непосредственно за компьютером жертвы, а такое могут себе позволить не все хакеры.

Примечание.

Значение слова «хакер» значительно объемнее, чем просто «вандал», пытающийся разрушить вашу систему. Подробнее вы можете прочитать об этом в документе Hacker-HOWTO, которые вы найдете на прилагаемом к книге CD.

Во-вторых, чтобы снять защиту с определенного объекта, например, файла /etc/passwd, нужно знать пароль администратора **LIDS,** а не пользователя root. Этот пароль хранится в зашифрованном виде в специальном файле. А этот файл, в свою очередь, «виден» только программе **LIDS,** вы его не увидите, даже если введете команду **ls-** 1. Таким образом можно защитить не только файл паролей **LIDS,** но и файл /etc/passwd, предварительно разрешив доступ к нему только программам **login, su и passwd,** то есть тем

программам, которым действительно этот файл нужен. Ограничить доступ можно по-разному, например, разрешив одной программе только читать файл, а другой — записывать в него.

LIDS может запретить загрузку или выгрузку модулей ядра, защитив вас таким образом от модулей-троянов. С помощью **LIDS** также можно запретить перезагрузку системы.

Встроенный детектор сканирования портов позволяет обнаружить большинство известных методов сканирования.

О любых «незаконных» действиях по отношению к защищенным с помощью **LIDS** объектам будет сообщено по электронной почте администратору, что позволит ему незамедлительно отреагировать на попытку взлома.

8.9.2, Установка LIDS

Для работы LIDS необходимо ядро версии 2.2.13 или выше. Загрузить версию **LIDS** для вашего ядра можно по адресу **http://www.lids.org/download.**

Сразу следует оговориться, что вам нужно будет настроить систему **LIDS** так, чтобы она была незаметной для остальных пользователей вашей системы и, в то же время, препятствовала бы вторжению в вашу систему. Для этого нужно учесть работу всех программ, поскольку после установки **LIDS** некоторые программы могут не функционировать. В связи с этим не устанавливайте **LIDS** на рабочей станции или на своем домашнем компьютере: работу на системе, защищенной с помощью **LIDS,** комфортной не назовешь. Идеальным узлом сети для установки **LIDS** является сервер или *firewall,* ограничивающий внутреннюю локальную сеть от Интернет. На сервере установлено сравнительно малое количество программ (в основном — это серверы служб), поэтому вам не нужно учитывать многочисленные пользовательские программы. В самом крайнем случае **LIDS** можно установить на рабочей станции, напрямую (то есть не через бастион) подключенной к Интернет. При большом количестве программ, установленных на рабочей станции, настройка **LIDS** может оказаться намного сложнее, чем в случае с сервером.

И еще одна рекомендация: не устанавливайте **LIDS** на компьютерах локальной сети, если сеть защищена бастионом — извне доступ к локальным компьютерам все равно никто не получит, так как сеть защищена бастионом, а от сотрудников вашей организации, у которых есть загрузочная дискета Linux и которые умеют нажимать на RESET, **LIDS** не спасет.

Итак, если вы решили установить **LIDS,** приступим к ее установке. Для начала убедитесь, что у вас есть копия исходных текстов ядра (например, в виде rpm-пакетов). Если ее нет, скопируйте исходные тексты ядра в другой каталог — на всякий случай. Это правило относится не только к **LIDS,** а и ко всем программам, которые добавляют патчи или сами являются патчами к ядру Linux. Распакуйте **LIDS:**

```
tar xfz LIDS-x.x.x-k.k.k.tar.gz
```

В номере версии **LIDS** я заменил номера версии буквами x, а номера версии ядра — буквами k. Теперь перейдите в каталог `/usr/src` и введите команду:

```
patch -p1 /usr/src/LIDS-x.x.x/LIDS-x.x.x-k.k.k.patch
```

Этой командой вы добавите патч LIDS к вашему ядру (**LIDS** я распаковал в каталог `/usr/src`). Затем нужно установить программу администрирования системы LIDS, а потом пересобрать ядро с поддержкой LIDS. С этой целью перейдите в каталог `/usr/src/LIDS-x.x.x/lidsadm-x.x.x` и введите команды:

```
make
make install
```

Если в процессе выполнения команды **make install** вы увидите сообщение, что не найден файл `lidsadm.1.gz` (такое наблюдалось в версии LIDS 0.9.8), просто введите команду:

```
gzip lidsadm.1
```

А затем снова — make install.

Теперь займемся конфигурированием ядра. Перейдите в каталог с исходными текстами ядра и запустите программу **menuconfig** (команда **make menuconfig**). Вам нужно включить следующие опции:

1. Prompt for development and/or incomplete code/drivers (Code maturity level options).
2. Sysctl support (General Setup).
3. Linux Intrusion Detection System support (EXPERIMENTAL) (General Setup).

При включении поддержки LIDS вы увидите список опций LIDS (табл. 8.9).

Опции LIDS *Таблица 8.9*

Опция	Описание
Maximum protected objects to manage	Максимальное количество защищаемых объектов. По умолчанию 1024. Пока не следует изменять данное значение
Maximum ACL subjects to manage	Максимальное количество субъектов правил. О субъектах и о правилах мы поговорим в следующем пункте, а пока оставьте данное значение без изменения (1024)
Maximum ACL objects to manage	Максимальное количество объектов правил
Maximum protected proceeds	Максимальное количество защищаемых процессов. По умолчанию 1024. Устанавливая максимальные значения, следует учитывать, что чем больше максимальное значение, тем больше код ядра
Hang up console when raising security alert	Отключает консоль злоумышленника при попытке доступа к защищенному объекту. Рекомендую включить данную опцию
Security alert when excepting unprotected programs before sealing LIDS	Включает вывод сообщения о нарушении безопасности при запуске незащищенных программ. Включите данную опцию — это позволит вам обнаружить незащищенные программы
Do not execute unprotected programs before sealing LIDS	Включает или отключает запуск незащищенных программ. Не включайте эту опцию до тех пор, пока не убедитесь, что все программы, которые запускаются автоматически при запуске системы, не защищены. На данном этапе пока лучше оставить эту опцию выключенной
Try not to flood logs	Система LIDS не будет записывать в протоколах повторяющиеся сообщения об одном и том же нарушении. Не включайте эту опцию
Allow switching LIDS protections	Включает или выключает защиту LIDS, то есть это возможность включения или выключения LIDS в процессе работы системы после ввода пароля администратора LIDS. Включите эту опцию. При включении этой опции вы увидите список подобных опций, например, опция Allow remote users to switch LIDS protections, разрешающая включать или выключать защиту LIDS удаленным пользователям. Не включайте эти опции! (особенно опцию Allow any program to switch LIDS protections)

Продолжение табл. 8.9

Опция	Описание
Port Scanner Detector in kernel	Эта опция добавляет мощный сканер портов в ядро, определяющий практически все известные способы сканирования портов. Несмотря на увеличение размера ядра, включите эту опцию
Send security alerts through network	Если эта опция включена, то при обнаружении нарушения на указанный адрес электронной почты будет отправлено сообщение об этом. Можно использовать как локальный, так и удаленный сервер SMTP. Включите данную опцию
LIDS debug (не во всех версиях LIDS)	Включает вывод отладочных сообщений. Включите эту опцию, если вы хотите самостоятельно переписать исходные тексты системы LIDS

Теперь убедитесь, что вы находитесь в каталоге `/usr/src/linux` и введите команды:

```
make clean
make dep
make install
make modules
make modules_install
```

Примечание.

Вместо пяти вышеуказанных команд можно ввести одну команду:

```
make clean dep install modules modules_install
```

Параметры (clean, dep ...) — это просто цели для сборки. Каждая цель определяет какие-нибудь действия, например, ***modules*** *собирает модули, а* ***modules_install*** *- устанавливает модули ядра.*

Теперь перейдем к настройке самой системы **LIDS.** Ни в коем случае не перезагружайте систему, чтобы убедиться, работает ли ядро или нет! Сначала нужно настроить **LIDS,** а только затем перезагрузить компьютер, чтобы убедиться, что все работает правильно. При настройке LIDS будьте предельно внимательны, иначе последствия могут быть катастрофическими, особенно, если у вас нет загрузочной дискеты.

8.9.3. Базовая настройка

После установки системы LIDS в каталоге /etc у вас появится подкаталог /lids. В нем вы обнаружите четыре файла: `lids.cap`, `lids.net`, `lids.pw`, `lids.conf`.

В первом из них хранятся текущие установки способностей **(cap** — от *capabilities* — способности). О том, что такое способности, мы поговорим немного позже.

Во втором файле (`lids.net`) находятся установки для отправки сообщений электронной почты. В третьем (lids.pw) — пароль в зашифрованном виде. Система **LIDS** использует метод шифрования **RipeMD-160**. Пароль можно изменить только с помощью программы **lidsadm**. **В** последнем файле определяются правила доступа. Этот файл можно изменять только с помощью программы **lidsadm.**

Способность — это возможность программы совершать какое-либо действие. В табл. 8.10 каждая из способностей рассмотрена более подробно. Формат способностей, в котором они содержатся в файле `lids.cap`, выглядит так:

```
[+/-]номер:название
```

Если в первом поле установлен знак «+», значит эта способность включена. Номер — это просто порядковый номер способности. Название определяет действие, разрешенное или запрещенное программам. Выключение способности распространяется на все программы, кроме тех, которые непосредственно указаны в правилах доступа с помощью программы **lidsadm**. Если способность включена, ее ограничение распространяется на все без исключения программы. Нельзя установить ограничение на все программы, кроме нескольких. Пример файла `lids.cop` приведен в листинге 8.15.

Листинг 8.15. Пример файла lids.cap

```
+0:CAP_CHOWN
+1:CAP_DAC_OVERRIDE
+2:CAP_DAC_READ_SEARCH
+3:CAP_FOWNER
+4:CAP_FSETID
+5:CAP_KILL
+6:CAP_SETGID
+7:CAP_SETUID
+8:CAP_SETPCAP
-9:CAP_LINUX_IMMUTABLE
-10:CAP_NET_BIND_SERVICE
+11:CAP_NET_BROADCAST
-12:CAP_NET_ADMIN
-13:CAP_NET_RAW
+14:CAP_IPC_LOCK
+15:CAP_IPC_OWNER
-16:CAP_SYS_MODULE
-17:CAP_SYS_RAWIO
-18:CAP_SYS_CHROOT
+19:CAP_SYS_PTRACE
+20:CAP_SYS_PACCT
-21:CAP_SYS_ADMIN
+22:CAP_SYS_BOOT
+23:CAP_SYS_NICE
+24:CAP_SYS_RESOURCE
+25:CAP_SYS_TIME
+26:CAP_SYS_TTY_CONFIG
+27:CAP_HIDDEN
+28:CAP_INIT_KILL
```

Способности *Таблица 8.10*

Способность	Описание
CAP_CHOWN	Разрешает (или запрещает, если способность выключена) программам изменять группу и владельца файла. Далее подразумевается, что рассматриваемая способность включена и если ее отключить, то данное действие будет недоступно программам
CAP_DAC_OVERRIDE	Программы, запускаемые пользователем root, не будут принимать во внимание права доступа к файлам. Например, если режим доступа к файлу пользователя равен 0600, то даже root не сможет открыть его (получить доступ к файлу)
CAP_DAC_READ_SEARCH	То же самое, но для каталогов (режимы доступа: чтение и поиск)
CAP_FOWNER	Запрещает операции с файлами, если идентификатор владельца файла не совпадает с идентификатором пользователя, который выполняет операцию
CAP_FSETID	Разрешает установку битов SUID и SGID для файлов, не принадлежащих пользователю root
CAP_KILL	Разрешает процессам пользователя root завершать («убивать») процессы других пользователей
CAP_SETGID	Разрешает программам изменять группу, под которой они работают. Программа должна быть запущена пользователем root. Эту возможность используют программы: httpd, sendmail, safe_mysql, safe_finger, postfix, ftpd
CAP_SETUID	Разрешает программам изменять пользователя, под которым они работают. Программа должна быть запущена пользователем root
CAP_SETPCAP	Включает способность программ редактировать способности
CAP_LINUX_IMMUTABLE	Отключите данную способность. Эта способность относится к таким атрибутам файлов, как S_IMMUTABLE (команда chattr -i) и S_APPEND (chattr -a)
CAP_NET_BIND_SERVICE	Разрешает программам прослушивать порты с номерами, меньшими 1024
CAP_NET_BROADCAST	Разрешает программам отправлять широковещательные пакеты
CAP_NET_ADMIN	Эта способность относится к сетевому администрированию: конфигурирование сетевых интерфейсов, изменение таблиц маршрутизации ядра, правил firewall и т.п.
CAP_NET_RAW	Разрешает программам использовать сокет-соединения (Raw Unix Socket)
CAP_IPC_LOCK	Разрешает процессам пользователя root блокировать сегменты разделяемой памяти
CAP_IPC_LOCK	Разрешает процессам пользователя root вмешиваться в межпроцессорное взаимодействие процессов других пользователей
CAP_SYS_MODULE	Управляет способностью загружать (выгружать) модули ядра. Отключите данную способность
CAP_SYS_RAWIO	Управление доступом к файлам устройств, например, /dev/mem, /dev/hd*, /dev/sd*. Другими словами, разрешает прямой ввод/вывод
CAP_SYS_CHROOT	Разрешает изменять корневой каталог в процессе работы пользователя. Отключите данную способность
CAP_SYS_PTRACE	Разрешает программа использовать функцию ptrace(). Включите данную способность
CAP_SYS_PACCT	Управляет способностью конфигурировать учет процессов. Отключите данную способность
CAP_SYS_ADMIN	Управляет способностью изменения многих системных параметров: от установления имени компьютера до монтирования дисков. Отключите данную способность, иначе ничего не сможете сделать в системе
CAP_SYS_BOOT	Управляет способностью перезагружать машину
CAP_SYS_NICE	Управляет способностью изменять приоритет процессов других пользователей
CAP_SYS_RESOURCE	Данная способность относится ко всевозможным ограничениям системных ресурсов, например, дисковые квоты, количество консолей. Выключите данную способность
CAP_SYS_TIME	Разрешает изменять системное время
CAP_SYS_TTY_CONFIG	Разрешает изменять настройки консолей
CAP_HIDDEN	Разрешает программам становиться невидимыми в списке процессов
CAP_INIT_KILL	Разрешает «убивать» потомков процесса init. К потомкам относятся практически все демоны, запущенные при запуске системы

Для инициализации способностей используются команды **lidsadm -I.** Эту команду обычно помещают в сценарии автозагрузки системы, например, rc.local, и, как правило, эта команда должна быть последней, чтобы могли беспрепятственно загрузиться демоны и инициализироваться сетевые интерфейсы.

Теперь перейдем к настройке параметров отправления сообщений электронной почты. Эти параметры, как уже было отмечено, находятся в файле lids.net (см. листинг 8.16).

Листинг 8.16. Файл lids.net

```
MAIL_SWITCH=1
MAIL_RELAY=127.0.0.1:25
MAIL_SOURCE=localhost
MAIL_FROM=LIDS@domain.ru
MAIL_TO=admin@adminhome.ru
MAIL_SUBJECT=The intrusion is revealed
```

Первый параметр включает (1) или отключает (0) функцию отправки сообщения. Параметр MAIL_RELAY определяет IP-адрес сервера SMTP и порт сервиса SMTP. MAIL_SOURCE — это источник почты, то есть узел, отправивший сообщение. Параметр MAIL_FROM устанавливает адрес отправителя, а MAIL_TO — адрес получателя. MAIL_SUBJECT — это тема сообщения.

В качестве адреса получателя рекомендую установить номер мобильного телефона, точнее e-mail-адрес, который сопоставлен с вашим номером телефона. Этот адрес можно узнать у вашего оператора мобильной связи. В этом случае сообщение о вторжении будет отправлено по SMS прямо на мобильный телефон, что очень удобно — не будете же вы всегда находиться возле компьютера, ожидая сообщения от **LIDS?**

Следующий этап настройки системы **LIDS** — это изменение пароля администратора системы LIDS. Для изменения пароля (точнее, установки нового пароля), введите команду:

```
lidsadm -P
```

8.9.4. Правила доступа

Правила доступа системы **LIDS** чем-то напоминают правила бастиона, но данные правила распространяются не на пакеты, а на программы. Правила доступа хранятся в файле lids.conf и редактировать этот файл можно только с помощью программы **lidsadm.** Первоначально у вас уже установлены определенные правила, просмотреть которые вы можете с помощью команды:

```
lidsadm -L
```

Обновить правила вы можете с помощью команды:

```
lidsadm -U
```

Я рекомендую вам очистить все правила и создать собственные. К тому же, первоначально установленные правила у вас не будут работать, так как кроме имени файла система **LIDS** также использует и номер **mode,** а в вашей системе номера информационных узлов (inodes) будут отличаться от номеров узлов на машине разработчиков LIDS. Очистить правила можно с помощью команды:

```
lidsadm -Z
```

Правила состоят из трех частей:

1. Объекта.
2. Субъекта.
3. Доступа (цели).

Объект — это любой объект (файл, каталог), который будет защищен с помощью системы LIDS или на который будет действовать ограничение доступа. Если защитить каталог, то будут защищены все файлы в этом каталоге, все подкаталоги, все файлы в подкаталогах и т.д.

Субъектом является защищенная программа, которой предоставляется доступ к защищенному объекту. Субъект также должен быть защищен с помощью LIDS. -

Параметр доступ (цель) устанавливает тип доступа к объекту:

READ.....чтение.
WRITE.....запись.
DENY.....запрещает любой доступ.
APPEND.....разрешает запись в конец файла.
IGNORE.....игнорирование защиты.

Устанавливается правило так:

```
lidsadm -A -o объект -s субъект -j доступ
```

Если параметр «субъект» не указан, то правило будет действовать для всех программ.

Для начала защитим нашу систему от самого известного «троянского коня» — **rootkit**. О том, что такое **rootkit** и какой вред он может причинить вашей системе вы можете прочитать в статье Инги Захаровой «Сканирование на предмет обнаружения rootkit'oB» **http://www.softerra.ru/review/security/16999/page1.html**.

Пакет **rootkit** заменяет стандартные утилиты администрирования их «поддельными» версиями, что позволяет скрыть следы атаки и несанкционированного доступа.

Для защиты от такого рода «троянов» создайте такие правила:

```
lidsadm -A -o /bin -j READ
lidsadm -A -o /sbin -j READ
lidsadm -A -o /etc -j READ
lidsadm -A -o /usr/bin -j READ
lidsadm -A -o /usr/sbin -j READ
lidsadm -A -o /lib -j READ
lidsadm -A -o /boot -j READ
```

Как видите, мы не определили субъект в наших правилах, поэтому установленные ограничения будут распространяться на все программы: мы разрешаем доступ «только чтение» всем программам, но запрещаем запись в указанные каталоги.

Как я уже отмечал, при установке правил нужно учитывать особенности установленных в вашей системе программ. Если оставить все как есть, некоторые программы не смогут работать. Например, программа **mount** пи-

шет в файл /etc/mtab при монтировании новой файловой системы. Установите такие дополнительные правила, разрешив некоторым субъектам доступ WRITE к некоторым файлам:

```
lidsadm -A -o /etc -s /bin/mount -j WRITE
lidsadm -A -s -o /etc /bin/umount -j WRITE
lidsadm -A -s -o /lib/modules/2.2.17-21mdk /sbin/depmod -j WRITE
lidsadm -A -s -o /etc/mtab /sbin/fsck.ext2 -j WRITE
lidsadm -A -s -o /etc /etc/rc.d/rc.local -j WRITE
lidsadm -A -s -o /etc/HOSTNAME /etc/rc.d/rc.sysinit -j WRITE
lidsadm -A -s -o /etc/mtab /etc/rc.d/rc.sysinit -j WRITE
```

Однако в этих правилах перечислены далеко не все субъекты, которым необходим доступ к указанным объектам, но этих правил достаточно для запуска системы и монтирования файловых систем. В этой главе будут еще рассмотрены субъекты регистрации пользователей в системе, которым необходим доступ к файлам /etc/passwd и /etc/shadow, а правила для всех остальных программ, которые используются в вашей системе вам предстоит добавить самостоятельно. Вот эти правила:

```
lidsadm -A -o /etc/shadow -j DENY
lidsadm -A -o /etc/shadow -s /bin/login -j READ
lidsadm -A /etc/shadow -s /bin/su -o -j READ
lidsadm -A -o /etc/shadow -s /usr/sbin/in.ftpd -j READ
```

Если не добавлять в систему **LIDS** последние три правила, никто (даже root) не сможет зарегистрироваться в системе. Иногда такая возможность бывает полезной, например, при создании стационарных серверов, не нуждающихся в администрировании. Такими серверами могут быть маршрутизаторы между подсетями большого предприятия. Если увеличение количества подсетей не предвидится, можно вообще отключить регистрацию пользователей (в том числе и пользователя root). Изменить конфигурационные файлы в такой системе можно, загрузившись с системной дискеты. Естественно, что такую дискету нужно создать до установки **LIDS.**

Полезно также защитить системные журналы от изменений. Для этого нужно защитить каталог /var/log, установив доступ READ для всех программ, а затем отдельно разрешив каждой программе писать только в свой протокол. Это довольно утомительный процесс, но один раз проделав его, вы будете уверены, что никто уже не сможет изменить протоколы, чтобы «замести следы». При использовании программы **logrotate** предоставьте ей доступ ко всему каталогу /var/log.

В качестве объекта могут выступать не только программы, но и способность. Например, вы можете выключить способность для всех программ и предоставить ее только какой-нибудь одной определенной.

Доступом в этом случае будут являться такие режимы:

INHERIT.....предоставлять потомкам процесса данную возможность.
NO_INHERIT.....не предоставлять.

Для определения таких правил используется команда:

```
lidsadm -A -s СУБЪЕКТ -t -o СПОСОБНОСТЬ -j ДОСТУП (ЦЕЛЬ)
```

Вся разница между указанием обыкновенного объекта и способности заключается в наличии параметра -t.

После настройки **LIDS** перезагрузите систему для проверки ее работоспособности. Если что-то пошло не так, загрузите Linux, передав ядру параметр **security=0**. Указание данного параметра отключит систему **LIDS** и вы сможете настроить ее заново:

```
LILO boot: linux security=0
```

8.9.5. Администрирование LIDS

Иногда уже в процессе работы системы нужно отключить или включить некоторые способности или произвести некоторые действия, которые запрещены с помощью системы **LIDS,** например, добавить какой-нибудь модуль в ядро при включенной способности CAP_SYS_MODULE.

Администрирование системы **LIDS** выполняется с помощью все той же программы — **lidsadm**. Каждый раз при выполнении программы **lidsadm** у вас будет запрошен пароль. Каждая попытка запуска **lidsadm** будет фиксироваться в протоколах, так же как и каждая попытка ввода неправильного пароля. В зависимости от настроек вашей системы на указанный вами e-mail будут посылаться сообщения при каждой попытке подобрать пароль администратора **LIDS.** Формат использования **lidsadm** в данном контексте таков:

```
lidsadm -S — +/-флаг
```

При администрировании доступны флаги, указанные в табл. 8.11.

Флаги администрирования LIDS *Таблица 8.11*

Флаг	Описание
-LIDS	Прекращение работы системы LIDS в текущей оболочке и всех его потомках. При этом вам можно будет запускать программы, запуск которых был запрещен при работе LIDS. Однако в глобальных масштабах (в масштабах всей системы) работа LIDS не будет прекращена, а это значит, что действия других пользователей будут ограничены системой LIDS. Обычно этот режим наиболее оптимален при администрировании LIDS
+LIDS	Возобновляет работу LIDS после ее останова
+RELOAD_CONF	При указании этого флага LIDS перечитывает файлы конфигурации
LIDS_GLOBAL	Прекращение или возобновление работы LIDS в масштабах всей системы
Способность	Включает или выключает указанную способность

Приведу несколько примеров по администрированию **LIDS.** Если вам нужно сделать изменения в. файлах конфигурации систем **LIDS,** выполните команды:

```
lidsadm -S — -LIDS
vi /etc/lids/lids.cap
lidsadm -S -• +RELOAD_CONF
lidsadm -S - +LIDS
```

Теперь разберемся подробнее с этим примером. Сначала вы останавливаете работу **LIDS** для текущей оболочки. Затем вы редактируете файлы конфигурации, например, файл `lids.cap`. Потом вы заставляете **LIDS** перечитать файлы конфигурации и возобновляете ее работу. При этом на

время правки файлов конфигурации система **LIDS** будет функционировать для других пользователей.

Приведу еще один полезный пример. Например, вы включили способность CAP_SYS_MODULE, а вам нужно добавить модуль «на лету», то есть в процессе работы системы. Для этого не нужно останавливать всю систему, достаточно просто отключить данную способность, добавить модуль и включить способность снова.

```
lidsadm -S - -CAP_SYS_MODULE
insmod module.o
lidsadm -S - +CAP_SYS_MODULE
```

9

ПРОТОКОЛ Server Message Block (SMB)

9.1. Установка Samba

В этой главе вам предстоит настроить пакет **Samba,** предназначенный для использования протокола SMB (Server Message Block), который также еще называется протоколом **NetBIOS.** С помощью пакета Samba ваш компьютер, работающий под управлением Linux, ничем не будет отличаться от рабочей станции или сервера сети Microsoft. Дополнительную документацию по этому поводу можно найти по адресу **http://www.samba.org,** а также на всевозможных форумах.

С помощью Samba вы сможете следующее:

1. Предоставлять доступ к разделам Linux для рабочих станций Windows.
2. Получать доступ к ресурсам сети Microsoft.
3. Распечатывать документы на сетевых принтерах сети Microsoft, а также позволить использовать свой принтер в качестве сетевого.

При установке, на первом этапе нужно установить пакеты **samba, samba-common и samba-client:**

```
# rpm -ih /mnt/cdrom/Mandrake/RPMS/samba*
```

При этом, если вы используете другой дистрибутив (не Mandrake, как указано в примере), перейдите в нужный каталог.

Пакет **samba** состоит из двух основных файлов — `smbd` и `nmbd`. Первый из них является носителем протокола SMB, а второй обеспечивает поддержку имен **NetBIOS.** Сразу же после их настройки ваш компьютер будет отображаться в сети.

После установки сервисы **smbd** и **nmbd** конфигурируются как автозапускаемые, то есть вам не придется самостоятельно производить никаких действий по их запуску. Возможно, вас не устраивает такой вариант (например, в тех случаях, если обращение к ним будет производиться редко, и вы хотите освободить память). В этой ситуации никто не мешает добавить их в файл конфигурации суперсервера `/etc/inetd.conf` и запускать «по востребованию». При этом не забудьте только отключить их автозагрузку с помощью конфигуратора системы.

В этой главе будет рассмотрена настройка пакета Samba «вручную», то есть не прибегая к помощи конфигуратора. Вы же можете использовать

конфигуратор netconf (см. рис. 9.1), но в этом случае есть одно «но»: если вы будете настраивать Samba или любую другую службу сервера в другом дистрибутиве (не Red Hat или Mandrake), привычного вам конфигуратора может и не быть, поэтому вы должны знать хотя бы назначение и расположение системных файлов той или иной службы сервера.

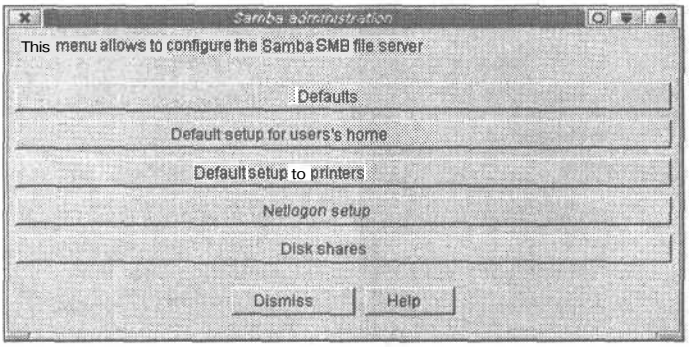


Рис. *9.1. Конфигурирование Samba*

Если же вы все-таки решили использовать конфигуратор, запустите **netconf** и, перейдя на вкладку **Server Tasks,** выберите конфигурирование Samba (см. рис. 9.1).

С помощью конфигуратора **netconf** вы можете полностью настроить пакет Samba — от указания общих параметров (см. рис. 9.2) до определения общих ресурсов (см. рис. 9.3).

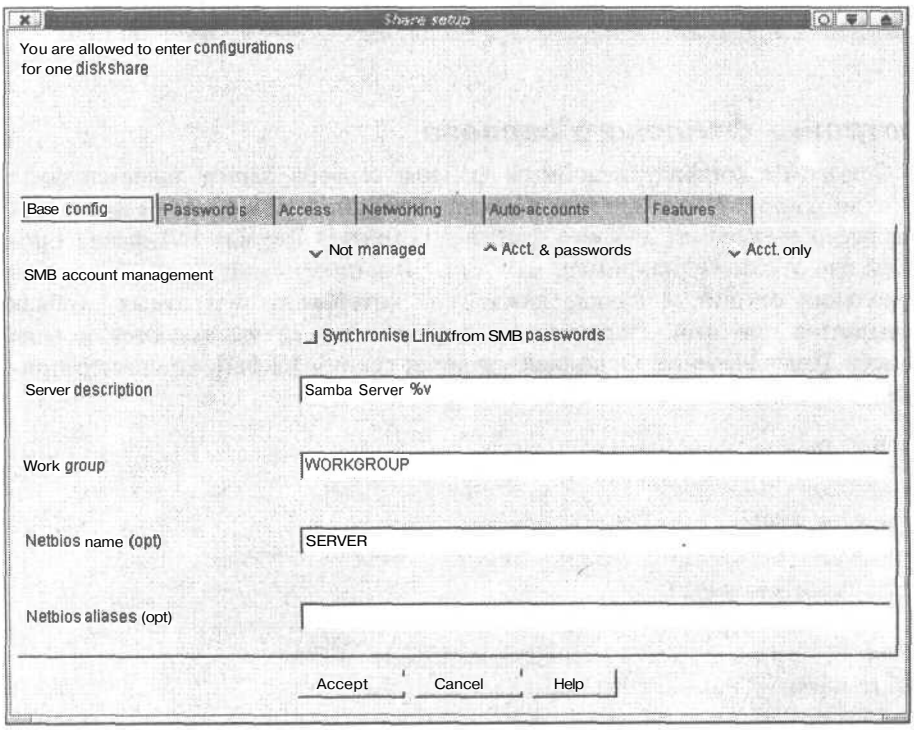


Рис. *9.2. Общие параметры*

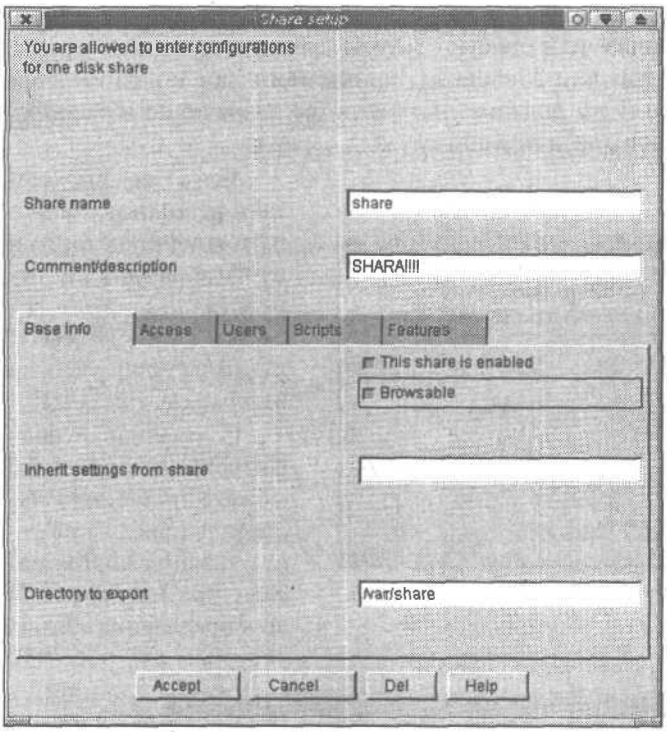


Рис. 9.3. *Определение общих ресурсов*

9.2. Настройка файлового сервера

Основным конфигурационным файлом сервера Samba является файл `/etc/smb.conf`. Именно в нем задаются все используемые и предоставляемые ресурсы. Формат данного файла напоминает формат INI-файла программ для Windows, например, `win.ini`. Файл `/etc/smb.conf` состоит из нескольких секций, в начале каждой из которых в квадратных скобках указывается ее имя. Параметры в каждой секции указываются в виде записей **Имя=Значение**. Основной является секция **[global]**, ее пример приведен в листинге 9.1.

Листинг 9.1. Пример секции global

```
[global]
workgroup = WORK
comment = Linux Server
guest account = guest
security = share
printing = bsd
printcap name = /etc/printcap
load printers = yes
client code page = 866
```

```
character set = koi8-r
encrypt passwords = Yes
log file = /var/log/samba/log.%m
max log size = 50
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
dns proxy = no
wins support = yes
domain master = yes
interfaces = 192.168.1.1/24 192.168.2.1/24
```

Параметр **workgroup** определяет рабочую группу или имя домена NT. Параметр **comment** аналогичен параметру NT **Description** для ОС Windows NT или **Description** (Описание компьютера) для ОС Windows 9x. Параметр **guest account** задает имя пользователя. Указание **guest** в качестве пользователя означает, что пользователи получают доступ без регистрации. Вернее, регистрация все же происходит, но используется гостевая учетная запись.

Следующий параметр — параметр **security** может принимать три значения:

share..... при каждом доступе будет запрашиваться имя пользовательского ресурса.

user..... для аутентификации будет использоваться имя пользователя и пароль, которые используются для входа в сеть Windows. Это значение используется по умолчанию.

server..... для проверки пароля будет использоваться сервер NT.

Записи **printing и printcap name** относятся к подсистеме печати. Первая из них задает систему печати типа BSD, а вторая — указывает, где расположен файл, содержащий информацию о принтерах. О настройке принтеров мы поговорим немного позже.

Параметры **client code page и character set** необходимы для корректного отображения русскоязычных имен файлов файловой системы Windows.

В ОС Windows NT, начиная с Service Pack 3, передача паролей по сети происходит в закодированном виде. Последние версии Samba позволяют поддерживать эту возможность. Для этого нужно установить значение параметра **encrypt password** равным yes. Если ваша версия Samba не поддерживает данную возможность, то вы можете отключить использование закодированных паролей в Windows. Учитывая, что вам придется вручную изменять параметр реестра всех рабочих станций Windows, мне кажется, что проще обновить Samba. Но если вас все же интересует, какой именно параметр реестра Windows нужно изменить, я укажу его. В разделе реестра ОС Windows NT:

[NT `HKEY_LOCAL_MACHINE\System\Current ControlSet\Services\Rdr\Parameters]`

нужно создать ключ **EnablePlainTextPassword** типа DWORD и установить его значение, равное 1. В ОС Windows 9x вам нужно создать тот же ключ, но в разделе `[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\Parameters]`.

В ОС Windows 2000 нужно внести изменения в раздел реестра

`[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanWorkStation\Parameters]`.

Параметры **log file** и **max log size** определяют имя файла протокола (журнала) и его максимальный размер. Опции сокетов задаются с помощью параметра **socket option.** Прежде, чем задавать опции сокета, рекомендую прочитать справочное руководство по файлу `smb.conf`.

Если в вашем компьютере установлено несколько сетевых интерфейсов, вы можете сконфигурировать пакет Samba так, чтобы он использовал все интерфейсы. Как это сделал я, показано в листинге 9.1.

Теперь перейдем к следующей секции, которая определяет параметры совместно используемых ресурсов. Данная секция называется **[homes].** Пример содержимого этой секции приведен в листинге 9.2.

Листинг 9.2. Секция [homes]

```
[homes]
comment = Home
browseable = yes
writable = yes
```

Параметр **browseable=yes** (см. листинг 9.2) разрешает отображение совместно используемых ресурсов в сети Microsoft. Параметр **writable =yes** разрешает запись в каталоги (вместо этого параметра может использоваться параметр **read only=no**).

Теперь создадим общий каталог (см. листинг 9.3). Делается это в секции **[public]**.

Листинг 9.3. Секция [public]

```
[public]
comment = Public Directory
path = /home/samba
read only = no
```

Ваших знаний уже достаточно, чтобы самому произвести дальнейшую настройку. В качестве завершения этого раздела я приведу пару практических примеров (см. листинг 9.4). Обратите внимание, что в файле `smb.conf` комментарии могут обозначаться либо решеткой (#), либо точкой с запятой (;).

Листинг 9.4.

```
; Каталог NETLOGON для входа в домен
[netlogon]
comment = Samba Netlogon Service
path = /var/netlogon
; Не устанавливайте значение yes
case sensitive = no
guest ok = yes
locking = no
writable = yes
browseable = yes
; Профиль для совместно используемых ресурсов
[Profiles]
```

```
path = /usr/local/samba/profiles
browseable = no
printable = no
guest ok = yes
; Каталог, используемый пользователем admin
; Пользователь admin должен существовать на сервере Samba
[admin]
comment = admin's directory
path = /home/admin
valid users = admin root
read only = no
```

9.3. Доступ к SMB-ресурсам из Linux

Будем смотреть правде в глаза, ОС Windows разрабатывалась для домохозяек. Каждая домохозяйка, чтобы просмотреть ресурсы сети Microsoft, использует пиктограмму «Сетевое окружение» на рабочем столе Windows. А теперь представьте, что эта домохозяйка работает в ОС Linux и хочет просмотреть ресурсы сети. Скорее всего, у нее возникнут определенные трудности. Попробуем их сейчас решить.

Для просмотра ресурсов сети Microsoft используется программа **smbclient**. Допустим, вы хотите подключиться к общему каталогу share компьютера nt_ws1. При этом допустим, что ваше имя пользователя *user* и пароль *123456*. В этом случае использование команды **smbclient** выглядит следующим образом:

```
$ smbclient //nt_ws1/share -U user%123456
```

Если пароль не нужен, то указывается только имя пользователя без знака процента.

После подключения к общему ресурсу, если точнее, к каталогу, вы можете использовать те же команды, что и при работе с клиентом **ftp** (см. табл. 9.1).

Команды программы smbclient *Таблица 9.1*

Команда	Описание
Dir	Выводит список файлов в каталоге
cd [каталог]	Выполняет переход в заданный каталог на сервере (учтите, что именно на сервере, а не на клиентском компьютере). В том случае, если каталог не указан, то smbclient просто выдаст имя текущего каталога
get [файл] [лок. имя]	Получает указанный файл из общего ресурса и сохраняет его на локальном компьютере. Если указано локальное имя, то полученный с сервера файл будет сохранен на клиентском компьютере под этим именем
put [файл] [удал. имя]	Копирует файл на удаленный компьютер (сервер) и сохраняет его там под именем, указанном в поле удаленное имя. Соответственно, если это имя не указано, то файл при сохранении переименовываться не будет
rm [каталог] rmdir [каталог]	Удаляет указанный каталог на удаленном компьютере (сервере)
md [каталог] mkdir [каталог]	Создает указанный каталог на удаленном компьютере (сервере)
mput [файлы]	Копирует все указанные файлы на удаленный компьютер (сервер)
del [файлы]	Удаляет на сервере указанные файлы, если конечно пользователь обладает на это правами
help	Помощь
exit или quit	Завершение сеанса работы программы smbclient

Использовать программу **smbclient** не очень удобно. Гораздо удобнее использовать программу **smbmount**, которая умеет монтировать удаленный общий ресурс как обычную файловую систему. При этом впоследствии использовать общий ресурс становится гораздо приятнее и удобнее. Ниже приведен пример команды, которая монтирует общий ресурс customers компьютера nt, используя имя пользователя *user*. Точка монтирования — каталог /mnt/customers, идентификатор пользователя (UID) равен 500, а группы (GID) — 100:

```
smbmount //nt//customers" -U user -c 'mount /mnt/customers -u 500 -g 100'
```

Для этих же целей можно воспользоваться командой:

```
smbmount //nt/customers/ /mnt/customers -U user
```

Примечание.

Использовать команду ***smbmount*** *имеет право только пользователь* root. *Для того, чтобы обычный пользователь мог использовать эту программу, следует установить для нее атрибут* ***Setuid root,*** *однако такое решение является небезопасным. Выходом из этого положения может послужить запуск программы* ***smbmount*** *при загрузке системы. Добавьте в сценарии автозагрузки вызов программы* ***smbmount*** *для монтирования файловых систем совместного использования, с которыми вы работаете чаще всего. После этого обычные пользователи смогут работать с удаленными ресурсами как с обычной локальной файловой системой.*

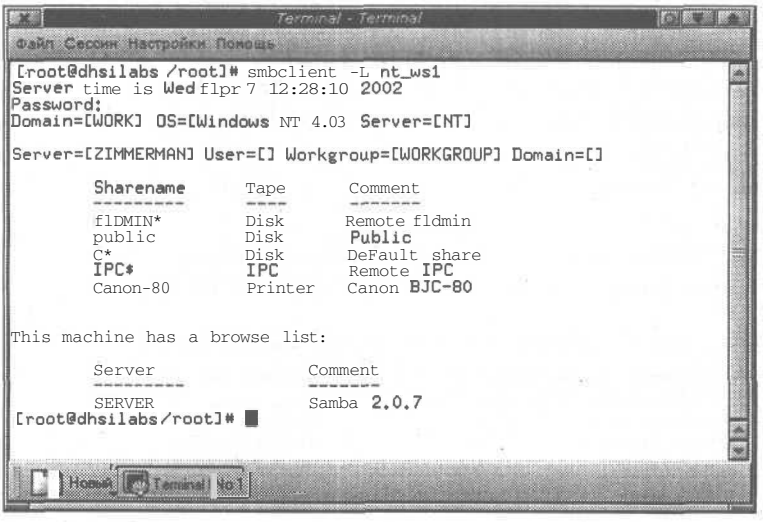


Рис. 9.4. *Общие ресурсы*

Просмотреть общие ресурсы компьютера можно с помощью опции -L (см. рис. 9.4).

Список **Browse list** (см. рис. 9.4) показывает другие SMB-сервера в сети с доступными ресурсами. Опция **-I** разрешает программе **smbclient** работать с именами DNS. Например, если домен называется domain.ru, то просмотреть общие ресурсы компьютера nt_wsl можно командой:

```
$ smbclient -L -I nt_ws1.domain.ru
```

9.4. Доступ к принтеру Linux для Windows-машин

Для обеспечения поддержки печати мы уже сделали почти все возможное. Параметр секции **[global] load printers** загружает принтеры из файла `/etc/printcap` (см. листинг 9.1). Используется система печати BSD. Теперь осталось определить секцию **[printers]** файла `smb.conf` (см. листинг 9.5). В этой секции задаются глобальные параметры для всех принтеров, поэтому нет необходимости указывать их отдельно для каждого принтера.

Листинг 9.5. Секция [printers]

```
[printers]
    comment = All Printers
    security=server
    path = /var/spool/lpd/lp
    browseable = no
    printable = yes
    public = yes
    writable = no
    create mode = 0700
```

Некоторые параметры, используемые в этой секции **(browseable, writable, comment),** имеют те же значения, что и в секции **[homes].** Параметр **path** задает буферный каталог, в который файлы будут копироваться перед печатью (так называемый *спул* принтера). Параметр **public** в значении yes разрешает печать из-под гостевой учетной записи, то есть всем желающим. Чтобы запретить печать из-под гостевой учетной записи укажите **public=no.** В этом случае доступ к принтеру будут иметь только зарегистрированные на сервере пользователи. Вместо параметра **public** иногда используется его синоним — параметр **guest ok.** Параметр **writable** установлен в значении **no** для того, чтобы в буферный каталог принтера (спул) могли записываться только печатаемые файлы.

Возможно, вам потребуется разрешить печать только одному или нескольким определенным пользователям на каком-то определенном принтере. Сделать это можно так, как это показано в листинге 9.6.

Листинг 9.6. Разрешение печати определенному пользователю

```
[admprn]
    valid user = root admin administator
    path = /home/admin
    printer = canon
    public = no
    writeable = no
    printable = yes
```

Подключение к Windows-компьютеру сетевого принтера, подключенного к Linux-серверу или Linux-станции, осуществляется аналогично подключению обыкновенного сетевого принтера, подключенного к рабочей станции

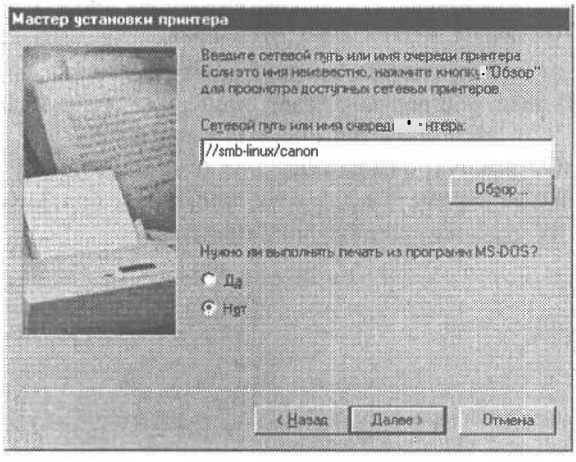


Рис. 9.5. *Подключение сетевого принтера*

Windows. В операционной системе Windows 98 для этого проделайте следующие действия:

1. Выполните команду меню **Пуск-> Настройка-> Принтеры.**
2. Активизируйте мастера **Установки принтера.**
3. Выберите тип принтера: **сетевой.**
4. Укажите путь к принтеру или нажмите на кнопку «Обзор» для автоматического выбора ресурса (см. рис. 9.5).
5. Далее установка сетевого принтера аналогична установке локального.

9.5. *Доступ к Windows-принтеру с компьютеров, работающих под Linux*

Прежде всего, вам нужно создать правильные записи в файле /etc/printcap. С форматом этого файла вы можете ознакомиться в листинге 9.7.

Листинг 9.7. Файл printcap

```
# /etc/printcap
#
# //nt_ws1/hp5m via smbprint
#
lp:\
# просто комментарий
    :cm=HP 5MP Postscript hp5m on nt_ws1:\

# имя устройства, открываемого для вывода
    :lp=/dev/lp0:\

# директория спула принтера (на локальной машине)
    :sd=/var/spool/lpd/lp:\

# файл учета использования принтера
    :af=/var/spool/lpd/lp/acct:\

# максимальный размер файла.
# Если указано значение «ноль», то ограничение снимается
    :mx#0:\

# имя фильтра
    :if=/usr/bin/smbprint:
```

В листинге 9.7 приведено (в комментариях) описание полей файла printcap, но, я думаю, не помешает отдельно привести их перечень:

cm..... задает комментарий;

lp..... имя устройства, открываемого для вывода;

sd..... директория спула принтера на локальной машине;

af..... файл учета использования принтера;

mx..... максимальный размер файла. Если указано значение «ноль», то ограничение снимается;

if..... имя входного фильтра.

Возвращаясь к настройке доступа, убедитесь, что каталог для спула принтера существует и разрешен для записи. Также нужно указать существующее устройство для вывода. В листинге 9.7 этим устройством является /dev/lp0.

В пакет Samba входит сценарий **smbprint**. С помощью этого сценария можно распечатывать документы на сетевом принтере, используя сервисы SMB. Возможно, в состав вашего пакета он не входит, поэтому я воссоздал его в листинге 9.8. Этот листинг частично позаимствован из руководства по пакету Samba.

Листинг 9.8. Сценарий smbprint

```
#!/bin/sh -x
# (c) Andrew Tridgell
# Этот скрипт является фильтром для системы печати, использующей
# файл /etc/printcap
# Он использует программу smbclient для печати файла на сетевом
# принтере,который подключен к рабочей станции Windows.
#
# smb:lp=/dev/null:sd=/usr/spool/smb:sh:if=/usr/local/samba/smbprint
#
# Запись создает unix-принтер, названный "smb", который будет
# печатать с помощью этого скрипта. Вам необходимо создать директорию
# спула /usr/spool/smb с соответствующими правами и владельцем

# Установите здесь имя сервера и принтер, на который вы хотите печатать.

I Далее скрипт был изменен Майклом Гамильтоном (Michael Hamilton)
# так что сервер, сервис и пароль могут быть считаны из файла
# /usr/var/spool/lpd/PRINTNAME/.config
f
# Для того, чтобы это работало, запись в /etc/printcap должна
# включать файл учета использования (af=...):
#
#cdcolour:\
#       :cm=CD IBM Colorjet on 6th:\
#      :sd=/var/spool/lpd/cdcolour: \
#       :af=/var/spool/lpd/cdcolour/acct:\
I       :if=/usr/local/etc/smbprint:\
#       :mx=0:\
#       :lp=/dev/null:
```

```
#
t Файл /usr/var/spool/lpd/PRINTNAME/.config должен содержать
#       server=PC_SERVER
f       service=PR_SHARENAME
#password="password"
f
#Например,
#       server=PAULS_PC
#       service=CJET_371
#      password=""
#
# Файл для отладочной информации, можно изменить на /dev/null
f
logfile=/tmp/smb-print.log
# logfile=/dev/null

spool_dir=/var/spool/lpd/lp
config_file=$spool_dir/.config

eval `cat $config_file `
echo "server $server, service $service" >> $logfile

(
        echo translate
        echo "print -"
        cat
) | /usr/bin/smbclient "\\\\$server\\$service" $password -U $user
-N -P >> $logfile
```

Теперь вы можете печатать на сетевом принтере. Но, тем не менее, я все же рекомендую прочитать руководство по пакету Samba для получения более подробной информации о печати на сетевых принтерах.

9.6. Пример файла smb.conf

В листинге 9.9 приведен пример моего файла `/etc/smb.conf`. Скорее всего он и у вас тоже будет корректно работать.

Листинг 9.9. Файл smb.conf

```
[global]
workgroup = WORK
comment = Linux Server
guest account = guest
security = share
printing = bsd
printcap name = /etc/printcap
load printers = yes
client code page = 866
character set = koi8-r
encrypt passwords = Yes
```

```
log file = /var/log/samba/log.%m
max log size = 50
# Следующие строки я закомментировал, потому что они характерны
f только для моей конфигурации
# socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
# dns proxy = no
t wins support = yes
# domain master = yes
# interfaces = 192.168.1.1/24 192.168.2.1/24
[admin]
comment = admin's directory
path = /home/admin
valid users = admin root
read only = no
[homes]
comment = Home
browseable = yes
writable = no
[public]
comment = Public Directory
path = /home/samba
read only = no
[printers]
comment = All Printers
path = /var/spool/lpd/lp
browseable = no
printable = yes
writable = no
create mode = 0700
guest ok = yes
```

9.7. Конфигуратор SWAT

Конфигуратор SWAT (Samba Web-based Administrative Tool) предназначен для настройки пакета Samba через Web-интерфейс. Как и другие конфигураторы, например, **netconf** или **linuxconf**, SWAT предоставляет удобный графический интерфейс для администрирования сервера Samba. Основным преимуществом данного конфигуратора является то, что вам не нужно находиться за компьютером, который вы администрируете. Администрировать сервера Samba вы можете из любого компьютера вашей сети. Как и при работе с другими конфигураторами, при работе со SWAT вам не нужно знать ни формат конфигурационных файлов, ни их название, ни расположение.

Для установки SWAT нужно установить пакет samba-swat. Обычно данный пакет находится на втором компакт-диске вашего инсталляционного набора Linux. Установите пакет командой:

```
rpm -ihv samba-swat-2.2.1a-4.i386.rpm
```

После установки пакета проследите за тем, чтобы в вашем файле /etc/services была следующая запись:

```
swat 901/tcp
```

Конфигуратор SWAT для своей работы использует протокол TCP и порт 901. Тем не менее, вы можете назначить любой другой порт. При изменении номера порта не забудьте изменить номер порта в файле /etc/inetd.conf или /etc/xinetd.conf.

Если вы используете суперсервер **inetd**, добавьте в файл /etc/inetd.conf следующую строку (если ее там нет):

```
swatstream  tcp  nowait.400     root/usr/sbin/swat swat
```

При использовании суперсервера **xinetd** в каталог /etc/xinetd.conf будет добавлен файл **swat** следующего содержания (листинг 9.11):

Листинг 9.11. Файл /etc/xinetd.conf/swat

```
# default: off
I description: SWAT is the Samba Web Admin Tool. Use swat \
tt to configure your Samba server. To use SWAT, \
# connect to port 901 with your favorite web browser.
service swat
{
    disable = no
    port    = 901
    socket_type = stream
    wait    = no
    only_from = 127.0.0.1
    user    = root
    server = /usr/sbin/swat
    log_on_failure += USERID
}
```

Если вы хотите конфигурировать сервер Samba с любого компьютера вашей сети, закомментируйте запись only_from= 127.0.0.1 или установите любые другие параметры доступа к SWAT.

Теперь нужно перезапустить суперсервер. Для этого введите команду:

```
/etc/init.d/xinetd restart
```

Можно также воспользоваться командой:

```
killall-HUPxinetd
```

При использовании **inetd** перезапустить суперсервер можно командой:

```
killall-HUPinetd
```

Все! Настройка SWAT завершена и теперь можно приступить к конфигурированию Samba с помощью SWAT. Для этого запустите свой любимый браузер и введите URL:

```
http://host:901
```

После установления соединения вы увидите окно, запрашивающее имя пользователя и пароль (см. рис. 9.6).

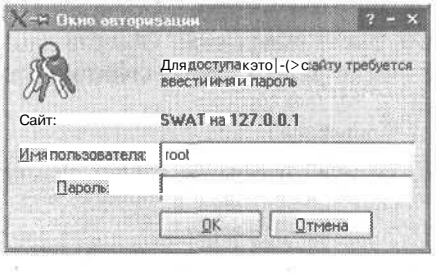


Рис. 9.6. *Окно авторизации*

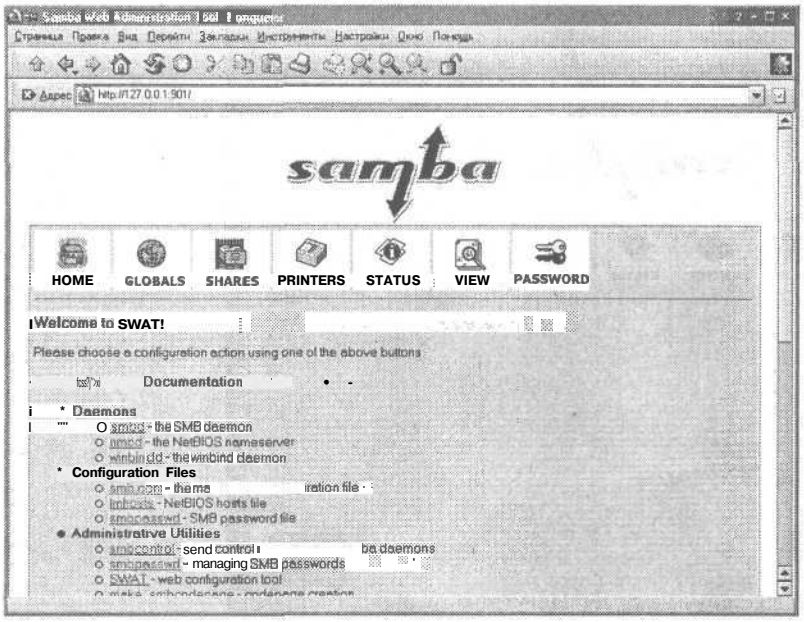


Рис. 9.7. *Samba Web Administration Tool*

Введите имя пользователя и пароль. Потом вы увидите основное окно конфигуратора (см. рис. 9.7).

Из рис. 9.7 видно, что в конфигураторе SWAT все самое нужное находится «под рукой» администратора: от документации до паролей пользователей.

В разделе **Globals** определяются значения глобальных переменных (рис. 9.8). Вы можете получить подсказку по тому или иному параметру,

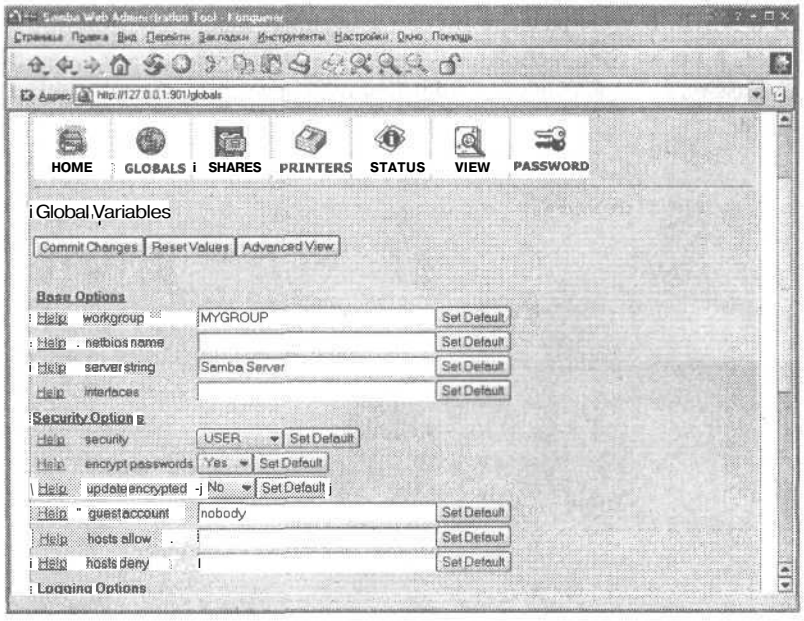


Рис. 9.8. *Глобальные переменны Samba*

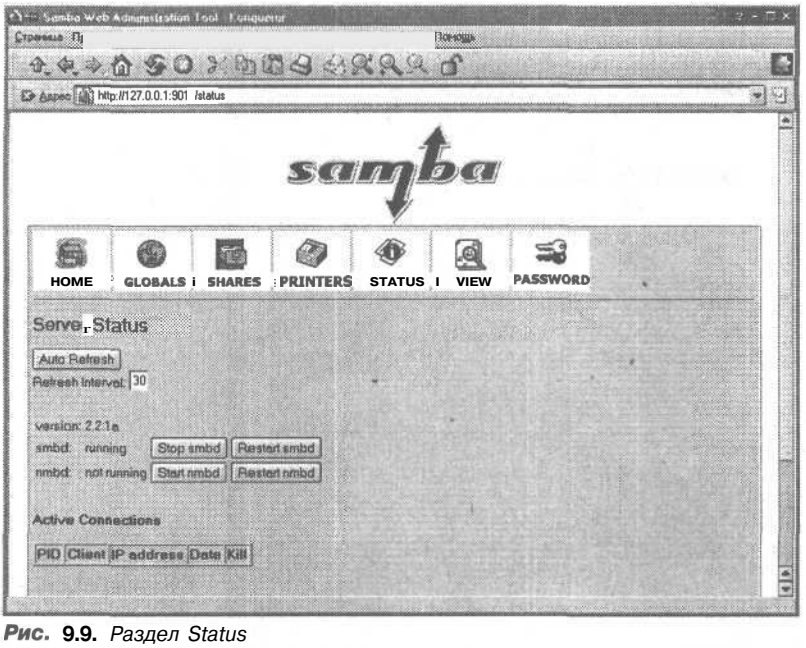


***Рис.* 9.9.** *Раздел Status*

перейдя по ссылке **Help.** Установить значение по умолчанию можно, нажав на кнопку **Set Default.**

В разделе **Shares** определяются общие ресурсы, а в разделе **Printers** — общие принтеры. Состояние сервера Samba можно посмотреть в разделе **Status** (см. рис. 9.9). Здесь же можно запустить, остановить и перезапустить сервисы SMB и NMB. В этом разделе можно завершить любое соединение с сервером Samba, а также просмотреть состояние соединения.

В разделе **Passwords** определяются пользователи, которые имеют доступ к серверу Samba.

10

Служба имен — DNS

Думаю, что не нужно в очередной раз рассказывать о преобразовании IP-адреса в имена компьютеров и о том, что выполняет служба DNS. Данному вопросу посвящены целые тома. Мы же давайте займемся непосредственно настройкой сервера DNS, который будет работать под управлением ОС Linux.

Перед началом настройки сервера DNS разберемся, как он работает. Система имен DNS — это иерархическая древообразная система. В этом дереве существует корень — он обозначается «.» (root). Список корневых серверов должен быть у каждого сервера: он содержится в файле `named.ca`, созданием которого мы займемся в п. 10.1. Этот файл может называться и по-другому — в зависимости от настроек сервера. Существует определенное количество доменов верхнего уровня. Наиболее известные вы знаете: com, gov, net, org и другие.

Допустим, что пользователь вводит в окне браузера адрес http://www.yahoo.com. Однако адресация в сети Интернет построена на основе IP-протокола. Поэтому для того, чтобы установить соединение с компьютером www.yahoo.com компьютеру пользователя необходимо знать его IP-адрес, поэтому операционная система пользователя пытается разрешить (перевести) имя компьютера в IP-адрес. С этой целью она обращается к серверу DNS. Сервер DNS сначала пытается разрешить имя данного компьютера, используя свой собственный кэш имен. Если требуемое имя компьютера в нем отсутствует, то сервер DNS обращается к одному из корневых серверов DNS, о которых мы поговорим позже. Запрос обрабатывается рекурсивно: корневой сервер обращается к серверу, который отвечает за домен com, а тот, в свою очередь, к серверу DNS домена yahoo.com. Сервер DNS домена yahoo.com возвращает IP-адрес компьютера www — 64.58.76.222 или все адреса, которые сопоставлены этому имени (многие сетевые операционные системы, в том числе и Linux, позволяют одному имени сопоставлять несколько IP-адресов).

Примечание.

На самом деле, если выполнить разрешение имени www.yahoo.com, сервер DNS возвратит следующие адреса:

64.58.76.222
64.58.76.228
64.58.76.223
64.58.76.176
64.58.76.224
64.58.76.177
64.58.76.227
664.58.76.179

А официальное имя компьютера www.yahoo.com (это его каноническое имя — о канонических именах и как их использовать будет сказано ниже) -- www.yahoo.akadns.net

10.1. Настройка сервера DNS

Учитывая, что на обращение к серверу DNS провайдера требуется 10...15, а иногда и все 30 секунд (это зависит от загрузки сети и от скорости соединения), установка сервера DNS в локальной сети с выходом в Интернет является просто необходимой. Обычно сервер DNS устанавливается на шлюзе, который используется для выхода в Интернет. Прежде чем приступить к настройке сервера, нужно определить, запущен ли он:

```
ps -ax I grep named
```

Если сервер DNS запущен, то его нужно остановить (командой **kill** или **ndc),** а если он вообще не установлен, то вам придется установить пакет **bind.** Обратите внимание, что исполнимый файл называется named, а сам пакет — **bind.** BIND (Berkley Internet Nameserver Deamon) — это наиболее известный и используемый DNS-сервер, настраиваемый в Linux. Для работы сервера должен быть активизирован сервис **network.** Я надеюсь, вы не забыли, как это сделать?

Теперь приступим к непосредственной настройке сервера и рассмотрим ее на примере. Для этого обратимся к файлу `/etc/named.conf`, в котором содержится основная информация о параметрах сервера (см. листинг 10.1).

Листинг 10.1. Файл named.conf

```
logging {
    category cname (null; };
    };
options {
    directory "/var/named";
    };
zone "." {
    type hint;
    file "named.ca";
    };
zone "dhsilabs.com" {
    type master;
```

```
    file "dhsilabs.com";
    notify no;
    };
  zone "0.0.127.in-addr.arpa" {
    type master;
    file "named.local";
    };
 zone "1.168.192.in-addr.arpa" {
    type master;
    file  "192.168.1";
    notify yes;
    };
```

Основной рабочий каталог сервера — `/var/named`. Указанные без начального обратного слэша имена файлов будут искаться относительно этого каталога. То есть именно в нем сервер будет искать файлы *dhsilabs.com, named.local, 192.168.1, named.ca* (см. листинги 10.1, 10.3, 10.4). Обслуживаемая сервером зона (домен) — **dhsilabs.com.**

Давайте рассмотрим поподробнее листинг 10.1. Сначала в нем были определены опции протоколирования — блок **logging.** Затем идет задание параметров самого сервера — блок **options.** Параметр **directory** определяет корневой каталог сервера -- /var/named. Помимо параметра **directory** в блоке **options** могут задаваться и другие параметры (такие, как **forwarders, forward** и др.), о которых сказано будет несколько позже (см. п. 10.2). Для функционирования сервера достаточно и одного параметра **directory.**

После блока параметров должны быть перечислены зоны, обслуживаемые сервером. Мы будем обслуживать зону (домен) **dhsilabs.com.** Информация об этой зоне хранится в файле `/var/named/dhsilabs.com`. Позже мы займемся созданием этого файла. С помощью него наш сервер будет преобразовывать имена компьютеров в IP-адреса. Для обратного преобразования служит файл `/var/named/192.168.1`.

Зоны "." и "0.0.127.in-addr.arpa" — особые. Я не буду их подробно описывать: их назначение вы поймете из дальнейшего текста книги. Файл `named.local` — это файл обратного соответствия, предназначенный для преобразования IP-адресов в имена, то есть, в частности, он используется для преобразования адреса 127.0.0.1 в имя **localhost.**

Файл `named.ca` -- это файл, в котором перечислен начальный набор корневых DNS-серверов. Он содержит информацию о корневых серверах DNS. При разрешении имени в IP-адрес или наоборот, полученная информация кэшируется и остается в памяти сервера определенное время. В свой работе, если нужно разрешить имя в IP-адрес (или наоборот), ваш DNS-сервер сначала будет искать необходимую ему информацию в кэше. Если ее там не окажется, то сервер обратится к одному из корневых серверов DNS, IP-адреса которых находятся в файле `named.ca`. Файл `named.ca` необходимо регулярно обновлять, чтобы он всегда содержал свежие данные (первый раз его нужно обновить сразу же после установки сервера, несмотря на то, что этот файл будет только-что создан). Немного позже я отдельно опишу его обновление.

Файл dhsilabs.com непосредственно служит для преобразования имен в IP-адреса (см. листинг 10.2).

Листинг 10.2. Файл ***dhsilabs.com***

```
@       IN      SOA     den.dhsilabs.com. hostmaster.dhsilabs.com. (
                93011120  ; серийный номер
                10800     ; обновление каждые 3 часа
                3600      ; повтор каждый час
                3600000   ; хранить информацию 1000 часов
                86400)    ; TTL записи — 24 часа
        IN      NS      den.dhsilabs.com.
        IN      A       192.168.1.1
        IN      MX      150 den.dhsilabs.com.
den     IN      A            192.168.1.1
        IN      HINFO   INTEL CELERON (LINUX)
        IN      MX      100 den
        IN      MX      150 evg.dhsilabs.com.
ns      IN      CNAME   den.dhsilabs.com.
www     IN      CNAME   den.dhsilabs.com.
ftp     IN      CNAME   den.dhsilabs.com.
mail    IN      CNAME   den.dhsilabs.com.
evg     IN      A            192.168.1.2
        IN      MX      100 den.dhsilabs.com.
localhost       IN      A       127.0.0.1
```

Попробую объяснить все как можно быстрее и проще. Свое объяснение оформлю в виде табл. 10.1.

Записи DNS ***Таблица*** *10.1*

Запись	Описание
NS	Обозначает сервер имен (name server)
A	Задает IP-адрес, соответствующий имени компьютера
PTR	Задает имя компьютера, соответствующее IP-адресу
MX число	Определяет почтовик, который будет обслуживать наш домен. Числовой параметр возле записи MX является приоритетом данного почтового сервера. Чем меньше число, тем выше приоритет
CNAME	Определяет каноническое имя узла, то есть, если вы в окне браузера введете http://www.dhsilabs.com, то обращение будет произведено к den.dhsilabs.com
HINFO	Сведения об аппаратном обеспечении. Рекомендую не заполнять эту запись или использовать заведомо неправильные данные. Чем меньше информации имеет о вашей сети злоумышленник, тем сложнее ему будет атаковать ее
TXT	Прочие сведения. Содержит произвольный текст

Обратите внимание на точку в конце

```
@       IN      SOA     den.dhsilabs.com. hostmaster.dhsilabs.com. (
```

Если точка не указана, то к имени будет добавлено имя домена (то есть dhsilabs.com).

Листинг 10.3. Файл ***named.local***

```
@   IN  SOA dhsilabs.com. root. dhsilabs.com. (
        199609203     ;серийный номер
```

```
        28800          ;обновление каждые 8 часов
        7200           ;повтор каждые 2 часа
        604800         ;хранить информацию 168 часов (1 неделю)
          86400)       ;TTL записи — 24 часа
        NS dhsilabs.com.
1       PTR localhost.
```

Файл 192.168.1 или файл обратного соответствия представлен в листинге 10.4.

Листинг 10.4. Файл обратного соответствия

```
@ IN  SOA den.dhsilabs.com. hostmaster.dhsilabs.com. (
        93011120      ; серийный номер
        10800         ; обновление каждые 3 часа
        3600          ; повтор каждый час
        3600000       ; хранить информацию 1000 часов
        86400 )       ; TTL записи - 24 часа
@  IN  NS  den.dhsilabs.com
1  IN  PTR den.dhsilabs.com
2.1.168.192    IN   PTR  evg.dhsilabs.com
```

Запись PTR используется для преобразования IP-адреса в имя.

Если указан не весь IP, например:

```
1  IN  PTR den.dhsilabs.com
```

то к нему будет добавлен адрес подсети 1.168.192.

IP-адреса указываются в обратном порядке!

Для установки файла корневого кэша следует установить пакет **caching-nameserver**, но я рекомендую получить и установить самую новую версию. Для этого подключитесь к Интернет, запустите сервер DNS, а затем выполните команду:

```
# nslookup I tee ns
```

В ответ на приглашение программы **nslookup** введите две команды

```
> set q=ns  (или set type=ns)
> .
```

На экране вы увидите список корневых серверов DNS, который будет помещен в файл ns. Для преобразования файла ns в формат named.ca воспользуйтесь следующей программкой на **awk** (см. листинг 10.5).

Листинт10.5. Сценарий reformat

```
#!/bin/awk
awk ' BEGIN {
/root/ { print ". IN NS " $4"." }
/internet/ { print $1"." " 999999 IN A " $5 }
END '
```

Использовать ее нужно как **reformat <source ffle> <output file>,** то есть:

```
reformat ns named.ca
```

Теперь осталось скопировать named.ca в каталог /var/named и на этом — все.

А теперь покажу, как то же самое можно было сделать проще. Для этого следует воспользоваться программой **dig,** выполнив команду:

```
dig @e.root-servers.net.ns > root.hints.new
```

После этого остается просто заменить старый файл named.ca **новым файлом** named.ca.new. **Как видите, второй способ намного проще, но и первый знать не помешает.**

Обычно файл named.ca **содержит примерно такую информацию:**

```
.                       6D    IN    NS        G.ROOT-SERVERS.NET.
.                       6D    IN    NS        J.ROOT-SERVERS.NET.
.                       6D    IN    NS        K.ROOT-SERVERS.NET.
.                       6D    IN    NS        L.ROOT-SERVERS.NET.
.                       6D    IN    NS        M.ROOT-SERVERS.NET.
.                       6D    IN    NS        A.ROOT-SERVERS.NET.
.                       6D    IN    NS        H.ROOT-SERVERS.NET.
.                       6D    IN    NS        B.ROOT-SERVERS.NET.
.                       6D    IN    NS        C.ROOT-SERVERS.NET.
.                       6D    IN    NS        D.ROOT-SERVERS.NET.
.                       6D    IN    NS        E.ROOT-SERVERS.NET.
.                       6D    IN    NS        I.ROOT-SERVERS.NET.
.                       6D    IN    NS        F.ROOT-SERVERS.NET.
;; ADDITIONAL SECTION:
G.ROOT-SERVERS.NET.     5w6d16h  IN A         192.112.36.4
J.ROOT-SERVERS.NET.     5w6d16h  IN A         198.41.0.10
K.ROOT-SERVERS.NET.     5w6d16h  IN A         193.0.14.129
L.ROOT-SERVERS.NET.     5w6d16h  IN A         198.32.64.12
M.ROOT-SERVERS.NET.     5w6d16h  IN A         202.12.27.33
A.ROOT-SERVERS.NET.     5w6d16h  IN A         198.41.0.4
H.ROOT-SERVERS.NET.     5w6d16h  IN A         128.63.2.53
B.ROOT-SERVERS.NET.     5w6d16h  IN A         128.9.0.107
C.ROOT-SERVERS.NET.     5w6d16h  IN A         192.33.4.12
D.ROOT-SERVERS.NET.     5w6d16h  IN A         128.8.10.90
E.ROOT-SERVERS.NET.     5w6d16h  IN A         192.203.230.10
I.ROOT-SERVERS.NET.     5w6d16h  IN A         192.36.148.17
F.ROOT-SERVERS.NET.     5w6d16h  IN A         192.5.5.241
```

Если вы настраиваете сервер DNS только для своей внутренней сети (intranet), которая не имеет выхода в Интернет, не спешите обновлять файл кэша! Он вам вообще не нужен. Вы также должны удалить зону, описывающую корневой кэш в файле named.conf.

Теперь остается сделать пару завершающих штрихов. Отредактируйте файл /etc/resolv.conf таким образом: с помощью директивы **search** укажите домены для поиска, а в качестве сервера по умолчанию — 127.0.0.1. Можно также указать и адрес реального интерфейса:

```
search subdomain.domain.com domain.com
nameserver 127.0.0.1
```

Как вы уже догадались, сервером DNS по умолчанию является первый сервер из списка **nameserver.** Напомню, что в списке может быть не более четырех серверов. Список доменов используется для поиска компьютера в

том случае, если указано только имя узла без домена. Например, если вы введете в окне браузера **http://host**, сначала будет выполнена попытка обращения к узлу host.subdomain.domain.com, а потом, если узел не будет найден, к узлу host.domain.com. Если и этот узел не будет найден, вы получите соответствующее сообщение. И еще: проверьте порядок разрешения имен в файле `/etc/hosts.conf`. Порядок должен быть задан так: order hosts,bind. Несмотря на то, что сейчас мы используем DNS, лучше сначала все же искать в файле `hosts`.

10.2. Кэширующий сервер DNS

,- Кэширующий сервер, как правило, не обслуживает домен, а используется для повышения скорости работы соединения. Для настройки кэширующего сервера используется параметр forwarders, задаваемый в файле `named.conf` (в блоке options). Рассмотрим пример: допустим, ваш сервер для разрешения какого-нибудь имени пытается добраться до одного из корневых серверов. А если у вас коммутируемое соединение да и модем на 14400? Сейчас выглядит смешно, но иногда бывают и такие ситуации, например, в моей системе спокойно уживаются два модема — один 56K V.90, а второй именно на 14K. В любом случае, если у вас нет собственного домена, а сервер DNS запущен на вашей машине, которую вы используете в гордом одиночестве, то с помощью вышеупомянутой директивы можно существенно повысить скорость соединения. Способ очень прост: можно заставить провайдера проделать за вас всю «грязную» работу. В обычной ситуации в процессе разрешения какого-нибудь имени ваш сервер будет последовательно запрашивать несколько удаленных корневых DNS-серверов, с каждым из которых надо установить соединение, отправить запрос и получить ответ. Создание у себя кэширующего DNS-сервера позволит возложить всю эту работу на DNS-сервер провайдера. При этом ваш DNS-сервер будет отсылать в сеть только один запрос на разрешение имени (DNS-серверу провайдера) и получать только один окончательный ответ. Это особенно полезно, если у вас плохое соединение с Интернет.

Для того, чтобы насладиться такой возможностью, следует в файл `named.conf` добавить следующие параметры (в блоке options):

```
forward first;
forwarders {
   192.168.99.1;
   192.168.99.2;
      };
```

Здесь я рассматриваю конкретный пример, вы же у себя замените адреса 192.168.99.1 и 192.168.99.2 на адреса DNS-серверов вашего провайдера. Параметр forwarders задает заключенный в фигурные скобки список IP-адресов, соответствующих DNS-серверам, которым ваш DNS-сервер будет переадресовывать запросы вместо того, чтобы отвечать на них самому. IP-адреса перечисляются через точку с запятой.

Параметр **forward** может принимать одно из двух следующих значений:

only.....ваш DNS-сервер никогда не должен предпринимать попыток обработать запрос самостоятельно;

first...ваш DNS-сервер должен пытаться сам обработать запрос, если указанные далее параметром forwarders сервера DNS не были найдены.

Использование параметра **forward** бессмысленно без использования параметра **forwarders.**

Таким образом, вернемся к настройке сервера, весь файл `named.conf` примет следующий вид, приведенный в листинге 10.6:

Листинг 10.6. Файл named.confкэширующего сервера DNS

```
options {
    directory "/var/named";
    forward first;
    forwarders {
        192.168.99.1;
        192.168.99.2;
    };
    // Раскомментируйте следующую строку, если вы
    // работаете через firewall и система не работает
    // query-source port 53;
};
zone "." {
    type hint;
    file "named.ca";
    };
zone "0.0.127.in-addr.arpa" {
    type slave;
    file " named.local ";
    };
```

Обратите внимание, что в примере уже не поддерживается зона *dhsilabs.com.*

Как правило, кэширующий сервер используется на отдельной машине, которая подключается к Интернет по коммутируемому соединению. Нужно учитывать, что сервер DNS сразу требует обращения к какому-нибудь сетевому ресурсу. В нашем же случае, если соединение не установлено, то устройство pppO существовать не будет, а named будет страшно ругаться на то, что сеть недоступна. При этом недоступным окажется даже интерфейс lo, а программа **nslookup**, если она нам понадобится без существования сети, просто «подвиснет», ожидая ответа от сервера DNS.

Есть два способа решить данную проблему. Какой использовать — это решать вам. Первый заключается в том, что при установлении соединения сценарий **ppp-on**, который обсуждался в гл. 7, будет запускать программу ndc с параметром **start** (см. ниже), а сценарий **ppp-off** будет останавливать сервер DNS командой **ndc stop.**

Второй способ основывается тоже на использовании сценариев **ppp-on и ppp-off**, но в этом случае сервер DNS всегда будет запущен. Принцип работы заключается в подмене файла корневого кэша named.ca. Сервер DNS содержит пустой файл корневого кэша, и при установке соединения сценарий **ppp-on** скопирует вместо пустого файла нормальный файл кэша. Сценарий **ppp-off при** разрыве соединения перезапишет нормальный файл named.ca пустым файлом с таким же именем. При использований этого способа в ваших протоколах (журналах) будут регулярно появляться сообщения примерно такого содержания:

```
Jan 5 16:10:11 den named[10147]: No root nameserver for class IN
```

Для полноты картины хочу отметить, что если вы используете NFS, и у вас возникают проблемы с монтированием удаленных файловых систем, запускайте сервер **named** после запуска **nfsd и mountd**.

10.3. Настройка дополнительного сервера DNS

Вы когда-нибудь обращали внимание, что у любого уважающего себя провайдера есть два сервера DNS -- первичный (primary или master) и вторичный (secondary или slave)? Так вот сейчас и мы займемся настройкой вторичного сервера **DNS.**

Примечание.

Практически у каждого провайдера работают два сервера DNS. Наличие двух серверов обеспечивает, если можно так выразиться, избыточность разрешения имени. Например, с первичным сервером что-нибудь случилось, произошел сбой, что бывает очень редко, или же просто первичный сервер не в состоянии обработать большое количество запросов клиентов. Тогда система разрешения имен операционной системы, получив отказ от первичного сервера DNS, обращается к вторичному.

Например, вам нужно создать вторичный сервер, который будет обслуживать домен domain.com. С этой целью внесите следующие изменения в файл named.conf дополнительного сервера:

```
zone "domain.com" {
    type slave;
    file "domain.com";
    masters { 192.168.1.1; 192.168.1.2; };
    };
```

IP-адреса основных серверов DNS вашей сети указываются внутри подсекции **master** через точку с запятой. Вторичный сервер, в отличие от кэширующего, всегда должен иметь тип **slave.**

10.4. Команды управления сервером DNS

Для управления сервером DNS используется программа **ndc**. Ее можно использовать с параметрами **start, stop, reload, restart.**

Параметр **start** запускет сервер, а **stop** — останавливает. Параметр **reload** перезагружает файлы зоны, если в них произошли изменения, а параметр **restart** перезапускает сервер DNS.

10.5. Использование nslookup

Программа **nslookup** используется для просмотра зоны DNS и входит в состав Linux (и всех вариантов UNIX), а также Windows NT.

Примечание.

*В данном случае зону следует понимать как домен и читать «для просмотра домена». В зоне содержится различная информация о компьютерах в домене. Зоны бывают разные: одни содержат информацию о компьютерах в домене и служат для преобразования имени компьютера в IP-адрес и наоборот (см. листинг 10.1), другие содержат информацию о корневых серверах — зона «.». Последняя зона относится к типу hint — подсказка, зоны для разрешения имен обычно имеют тип **master** (главный), а зоны вторичных серверов относятся к типу **slave** (подчиненный).*

Просмотр зоны — это просмотр информации, которую содержит зона. Обычно просмотр зоны разрешается только определенным, доверенным хостам. Итак, запустите **nslookup:**

```
# nslookup
Default Server: myserver.domain.com
Address: 127.0.0.1
>
```

Для того, чтобы получить информацию от сервера, нужно установить тип запроса set **q=<type>** (или **set type=<type>**). Перечень типов запросов представлен в табл. 10.2.

Типы запросов *Таблица 10.2*

Тип	Описание
soa	Начало полномочий
a	Преобразование имени в IP-адрес узла
aaaa	Отображение IPv6-адреса узла
ns	Отображение информации о сервере DNS
ptr	Преобразование IP-адреса в имя узла
wks	Распространенные службы
hinfo	Информация о «железе» узла
mx	Информация о почтовых серверах домена
txt	Отображение записи общего назначения
cname	Отображение канонического имени
any	Отображение всех ресурсных записей

Теперь рассмотрим несколько практических примеров. Например, вы знаете имя узла — www.server.com. Давайте посмотрим, какая информация будет выведена при указании типа **any:**

```
>set q=any
>server.com
Server: myserver.domain.com
Address: 127.0.0.1
Non-authoritative answer:
server.com nameserver = comp1.server.com
server.com nameserver = comp2.server.com
```

```
server.com nameserver = comp3.server.com
Authoritative answers can be found from:
server.com nameserver = comp1.server.com
server.com nameserver = comp2.server.com
server.com nameserver = comp3.server.com
comp1.server.com internet address = 323.111.200.1
comp2.server.com internet address = 323.111.200.2
comp3.server.com internet address = 323.555.200.3
```

Теперь получим сведения о зоне и почтовиках.

```
>server comp1.server.com
Default Server: comp1.server.com
Address: 323.111.200.1
>server.com.
Server: comp1.server.com
Address: 323.111.200.1
server.com internet address = 123.111.200.2
server.com nameserver = comp1.server.com
server.com nameserver = comp2.server.com
server.com nameserver = comp3.server.com
server.com
origin = comp2.server.com
mail addr = root.server.com
serial = 19
refresh = 10800 (3 hours)
retry = 7200 (2 hours)
expire = 86400 (1 day)
minimum ttl = 3600 (1 hour)
server.com preference = 10, mail exchanger =mail.server.com
comp1.server.com internet address = 323.111.200.1
comp2.server.com internet address = 323.111.200.1
comp3.server.com internet address = 323.111.200.3
mail.server.com internet address = 323.111.200.17
```

А сейчас посмотрим информацию о других узлах в этой сети:

```
ls server.com.
[comp2.server.com]
server.com. 323.111.200.2
server.com. server = comp1.server.com
server.com. server = comp2.server.com
server.com. server = comp3.server.com
mail 323.111.200.17
gold 323.111.200.22
www.ie 323.111.200.11
```

```
jersild 323.111.200.25
comp1 323.111.200.1
comp3 323.111.200.3
parasit3 323.111.200.20
www.press 323.111.200.30
compl 323.111.200.1
www 323.111.200.2
```

Теперь вам понятно, почему не нужно вообще использовать запись HINFO? Если при реальной атаке злоумышленник выяснит, какая операционная система используется на компьютерах в вашей сети, ему будет проще нанести удар. Я не отрицаю, существует много способов выяснить тип ОС, но зачем же сообщать это самому?

Примечание.

В записи HINFO обычно указывается информация об аппаратном обеспечении, платформе компьютера и его операционной системе.

Несколько замечаний:

1. IP-адреса использовались учебные.
2. Не всегда все так просто: иногда настройки сервера DNS и *firewall* не разрешат вам просмотреть некоторую информацию о зоне, например ту, которую мы получали с помощью команды **ls server.com.**

   Разрешить передачу зоны (трансфер зоны) определенным узлам, а значит, запретить всем остальным, вы можете с помощью директивы **allow-transfer.** В следующем примере трансфер зоны разрешен узлам 10.1.1.1 и 10.1.2.1. Другими словами, на узлах 10.1.1.1 и 10.1.2.1 можно будет использовать команду **nslookup** **ls** для просмотра зоны.

```
allow-transfer
{
10.1.1.1;
10.1.2.1;
};
```

11

Настройка FTP

Сервер **FTP** (File Transfer Protocol) используется для обмена файлами между системами. Обычно на FTP-сервере размещают большое количество файлов, например, какой-нибудь программный комплекс или набор музыкальных файлов. Примером FTP-сервера может послужить сервер **ftp://ftp.redhat.com.** На этом сервере вы можете найти как саму операционную систему Linux Red Hat, так и обновления ее пакетов, а также дополнительные программы.

Доступ к серверу FTP осуществляется с помощью FTP-клиента. В любой сетевой операционной системе есть простейший FTP-клиент — программа **ftp** (см. п. 19.3). Обычно для того, чтобы начать работу с FTP-сервером, вы должны зарегистрироваться на нем, другими словами, ввести имя пользователя и пароль. После регистрации вы получаете доступ к своему каталогу. Над файлами и каталогами вы можете производить обычные операции: создание, удаление, копирование, перемещение, переименование. Как правило, при выполнении операции копирования вы либо копируете файлы на сервер (команда **put)** -- загружаете на сервер, либо копируете файлы с сервера на свою локальную машину (команда **get)** — скачиваете с сервера.

Существуют также так называемые общедоступные (анонимные) серверы, к которым имеют доступ все пользователи. На таких серверах размещается свободно распространяемое программное обеспечение, обновление программ, драйверы, документация и прочая публичная информация. Для регистрации на таких серверах обычно нужно использовать имя пользователя *anonymous,* а в качестве пароля - - адрес электронной почты. Создание анонимного сервера рассмотрено в пункте 11.3.

В главе 19.3 рассмотрены все команды клиента FTP. Сейчас же просто рассмотрим регистрацию пользователя на сервере и что при этом происходит. Допустим, имеется некий FTP-сервер, к которому могут подключаться не только обычные пользователи, но и анонимные (см. рис. 11.1).

Перед самой регистрацией сначала FTP-клиент сообщит, что соединение с сервером FTP установлено, а затем вас поприветствует сам сервер - **сообщение 220: ProFTPD 1.2.0 Server.** Далее, в ответ на приглашение **Name** введите имя пользователя (пусть будет den). Так как пользователь не являет-

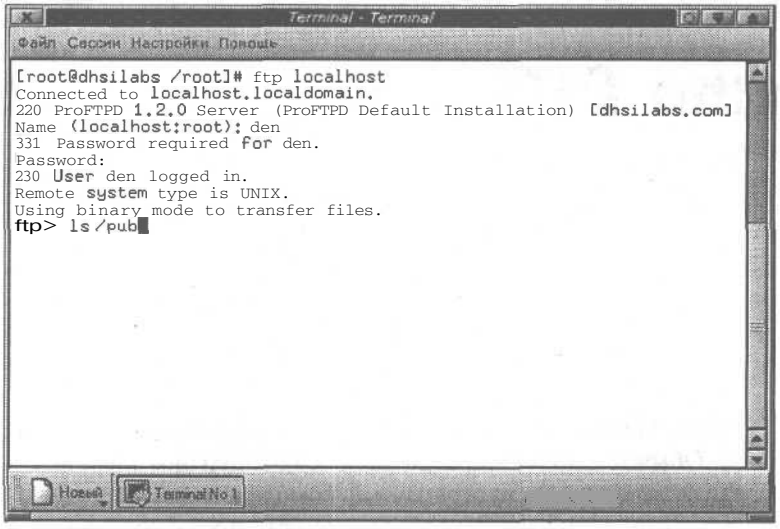


Рис. 11.1. *FTP-клиент в окне терминала*

ся анонимным, сервер сообщит вам, что нужно ввести пароль для этого пользователя. Правильно введя пароль, вы успешно зарегистрируетесь на сервере, о чем вам сообщит сервер — сообщение 230. Также сервер вам сообщит, что удаленной системой является UNIX и что сейчас используется двоичный (binary) режим передачи файлов. Я не рекомендую изменять этот режим на текстовый, потому что при передаче нетекстовой информации, например, пакетов RPM, двоичные файлы будут повреждены.

11.1. Сервер FTP wu-ftpd

Сервер FTP wu-ftpd является разработкой Вашингтонского университета. Этот сервер очень распространен и входит в состав практически каждого дистрибутива Linux. Для его установки нужно установить пакет wu-ftpd. Как и любой другой сервер, wu-fptd может быть постоянно загруженным в память или вызываться суперсервером inetd по мере необходимости. Первый режим называется standalone и применяется, как правило, если FTP-серверу нужно часто обрабатывать запросы клиентов. Второй режим используется в целях экономии памяти и если нагрузка на FTP-сервер не очень велика.

Если вы решите настроить работу сервера FTP во втором режиме, в файле `inetd.conf` должна быть соответствующая запись (см. листинг 11.1)

Листинг 11.1. Фрагмент файла inetd.conf

```
ftp    stream  tcp    nowait    root/usr/sbin/tcpd    in.ftpd -l -a
```

Из листинга 11.1 видно, что FTP-сервер вызывается не напрямую, а через демон tcpd, чем обеспечивается дополнительная безопасность. В том случае, если вы используете новую версию inetd — **xinetd**, формат записи у вас будет другой (см. листинг 11.2).

Листинг 11.2. Фрагмент файла xinetd.conf

```
service ftp
{
    socket_type     = stream
    wait            = no
    user            = root
    server          = /usr/etc/in.ftpd
    server_args     = -1
    instances       = 4
    log_on_success  += DURATION USERID
    log_on_failure  += USERID
    access_times    =  2:00-8:59  12:00-23:59
    nice = 10
}
```

Опция **-l** сервера FTP регистрирует все сеансы FTP в файле `syslog`. Кроме этой опции, сервер ftp имеет ряд других опций, указанных в табл. 11.1.

Опции командной строки сервера wu-ftpd *Таблица 11.1*

Опция	Описание
-d	Записывает отладочную информацию в журнал syslog
-l	Регистрирует все FTP-сеансы в журнале syslog
-L	Регистрирует в журнале syslog все команды, отправленные серверу FTP
-tсекунды	Устанавливает предел времени ожидания для пассивных клиентов (по умолчанию 15 минут). Если за этот промежуток времени от клиента не поступит ни одной команды, то FTP-сеанс с сервером будет разорван
-Tсекунды	Максимально допустимое время сеанса FTP (по умолчанию 2 часа)
-a	Разрешает использование файла конфигурации ftpaccess
-A	Запрещает использование файла конфигурации ftpaccess. Эта опция установлена по умолчанию
-i	Регистрирует в журнале xferlog файлы, полученные сервером FTP
-o	Регистрирует в журнале xferlog файлы, переданные сервером во время сеанса

Сервер **wu-ftp** использует пять файлов конфигурации: ftpaccess, ftphosts, ftpusers, ftpgroups, ftpconversions. В этой главе мы подробно рассмотрим каждый из них.

11.1.1. Файл ftpaccess

Основным файлом конфигурации является ftpaccess. Как и другие файлы конфигурации, он располагается в каталоге /etc. Пример файла /etc/ftpaccess приведен в листинге 11.3.

Листинг 11.3. Пример файла ftpaccess

```
class    all  real,guest,anonymous    *
email root@localhost
loginfails 3
readme   README* login
readme   README* cwd=*
```

```
message /welcome.msg    login
message .message        cwd=*

compress    yes         all
tar         yes         all
chmod       no          guest,anonymous
delete      no          guest,anonymous
overwrite   no          guest,anonymous
rename      no          guest,anonymous

log transfers anonymous,real inbound,outbound

shutdown /etc/shutmsg

passwd-check rfc822 warn
```

Файл ftpaccess определяет возможности доступа к FTP-серверу, которые будут иметь различные группы пользователей. В этом файле задаются параметры доступа, разрешенные операции, виды регистрируемых событий.

Директива **class** определяет класс пользователей, которые будут иметь доступ к серверу FTP. В листинге 11.3 задан класс **all,** который состоит из следующих типов пользователей: настоящие (real), гости (guest), анонимные **(anonymous).** Под настоящими пользователями подразумеваются те, которые зарегистрированы на сервере, то есть их учетные записи хранятся в файле /etc/passwd.

С помощью директивы **email** можно указать адрес администратора сервера.

Директива loginfails задает максимальное количество попыток регистрации. Если это количество превышено, пользователь автоматически будет отключен. Значение по умолчанию для этой директивы равно 5.

Директива **message** определяет файл и событие, когда он должен быть отображен. Например, можно создать несколько файлов, один из которых будет отображаться при регистрации пользователя, а другой — при входе его в определенный каталог.

Директивы **chmod и delete** определяют, могут ли пользователи использовать одноименные команды FTP. А директивы **overwrite** или **delete** разрешают или запрещают определенным пользователям перезаписывать или удалять файлы на сервере. В приведенном примере (см. листинг 11.3) пользователи классов **guest и anonymous** не могут выполнять ни одну из упомянутых операций. Общие директивы сервера **wu-ftpd** перечислены в табл. 11.2.

Директивы сервера wu-ftpd *Таблица 11.2*

Директива	Описание
autogroup имя_группы имя_класса [...]	Разрешает доступ анонимным пользователям определенных классов к файлам, которые принадлежат к указанной группе
alias псевдоним каталог	Создает псевдоним для каталога на FTP-сервере. Псевдоним позволяет быстро (указав только псевдоним) перейти в соответствующий ему каталог из любого другого каталога на сервере
anonymous-root каталог [имя_класса]	Указывает каталог, который будет использоваться в качестве корневого для заданного класса пользователей. После успешной регистрации пользователя на FTP-сервере он автоматически попадет в соответствующий его классу каталог. Если имя класса не указано, то данная директива будет задавать корневой каталог для анонимных пользователей, для которых корневой каталог не определен явно

Продолжение табл. 11.2

Директива	Описание
banner файл	Перед регистрацией клиента ему будет показано сообщение из указанного файла. В качестве параметра файла задается полное (абсолютное) имя файла
bit-limit [raw] in | out | total макс_кол_байт [имя_класса]	Устанавливает ограничение на количество пересылаемой информации в байтах для пользователей указанного класса. Если имя класса не указать, то данное ограничение будет применяться ко всем пользователям, для которых нет явных указаний по этому поводу. Необязательный параметр raw позволяет ограничить весь объем пересылаемой информации (в том числе и служебной), а не только пересылаемых файлов. Значения in, out, total указывают поток данных (на сервер, от сервера или и тот, и другой одновременно), подлежащий учету
class имя_класса типы_пользователей адреса_хостов	Создает класс пользователей с указанным именем. В качестве типа пользователей используются ключевые слова anonymous (анонимные пользователи), guest (гостевые пользователи) и real (зарегистрированные пользователи). Если указывается несколько типов, то они перечисляются через запятую без пробелов. В поле адреса_хостов указываются адреса хостов, только пользователи которых будут принадлежать данному классу. Символ звездочка «*» означает все хосты (см. листинг 1 1.3). Адреса хостов могут указываться в виде одного из следующих форматов: IP-адрес. Отдельный IP-адрес. , IP-адрес: маска_сети. IP-адрес с маской сети. IP-адрес/cidr. IP-адрес с информацией CIDR. !nameserved. Указание этого идентификатора приводит к запрету доступа со всех хостов, имена которых не удается получить от DNS-сервера. /имя_файла. Указывается абсолютное имя текстового файла, в котором содержится список IP-адресов (по одному в каждой строке)
cdpath каталог	Определяет для директивы cdpath выражение, с помощью которого задается путь поиска при переходе в указанный каталог
compress yes | no имя_класса	Разрешает или запрещает сжатие данных перед отправкой (команда compress) для указанного класса пользователей
defaultserver private	Запрещает анонимный доступ к серверу
deny адреса_хостов файл_сообщения	Запрещает доступ к серверу для хостов с указанными адресами. При этом будет отображено сообщение из файла_сообщения. При указании файла необходимо использовать полное (абсолютное) имя. Адреса хостов могут указываться в виде одного из следующих форматов: IP-адрес. Отдельный IP-адрес. IP-адрес: маска_сети. IP-адрес с маской сети. IP-адрес/cidr. IP-адрес с информацией CIDR. !nameserved. Указание этого идентификатора приводит к запрету доступа со всех хостов, имена которых не удается получить от DNS-сервера. /имя_файла. Указывается абсолютное имя текстового файла, в котором содержится список IP-адресов (по одному в каждой строке)
email адрес_почты	Почтовый адрес администратора сервера
file-limit [raw] in | out| total количество_файлов [имя_класса]	Устанавливает ограничение на количество пересылаемых файлов для пользователей указанного класса. Параметр количество_файлов как раз и задает максимально допустимое количество файлов. Значение остальных параметров такое же как и для директивы bit-limit
guestgroup имя группы [имя группы....]	Всем пользователям, входящим в группу с указанным именем, будет разрешен гостевой доступ к серверу FTP
limit имя_класса максимум периоды файл_сообщения	Ограничивает число одновременно работающих пользователей, принадлежащих указанному классу, в определенное время суток. Параметр максимум задает максимально допустимое количество одновременно работающих пользователей. Параметр периоды задает временные интервалы. Клиенту, которому запрещается доступ к FTP-серверу в результате действия данной директивы, будет показано сообщение из файла файл_сообщения
loginfails количество	Определяет максимальное число неудачных попыток регистрации пользователя, после которых он будет отключен. По умолчанию количество попыток равно 5

Продолжение табл. 11.2

Директива	Описание
log commands типы_пользователей	Регистрирует в журнале команды, которые вводились пользователями указанных типов. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)
log transfers тип_пользователей список_направлений	Регистрирует в журнале акты передачи файлов пользователями указанных типов. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class). В поле список_направлений задается направление передачи, подлежащее протоколированию: inbound (входящие файлы), outbound (исходящие). Если указываются оба направления, то они должны быть разделены запятой без пробела (см. листинг 1 1 .3)
message файл_сообщения действие	Отображает файл_сообщения во время регистрации или при переходе в другой каталог. Соответственно значение в поле действие может быть либо LOGIN (регистрация) или CWD=каталог (переход в каталог). Запись cwd=* задает любой каталог (см. листинг 1 1.3)
noretrieve [class=имя_класса] список_файлов	Запрещает получение указанных в списке файлов. Если указан параметр class, то этот запрет распространяется только на пользователей заданного класса
readme файл действие	Во время регистрации или при смене каталога пользователь получит сообщение о существовании и времени модификации указанного файла. Параметр действие определяется так же, как и в директиве message
tar yes | no имя_класса	Разрешает или запрещает использование команды tar для указанного класса пользователей, то есть разрешает или запрещает архивирование файлов архиватором tar перед их пересылкой
virtual адрес	Разрешает использование виртуального FTP-узла

Кроме общих директив, сервер **wu-ftp** имеет директивы, которые управляют правами доступа. Директивы прав доступа определяют, какие операции могут выполнять пользователи того или иного типа. Эти директивы указаны в табл. 11.3.

Директивы прав доступа *Таблица 11.3*

Директива	Описание
chmod yes | no типы_пользователей	Разрешает или запрещает выполнять команду chmod для пользователей указанных типов. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)
delete yes | no типы_пользователей	Разрешает или запрещает выполнять команду delete для пользователей указанных типов. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)
overwrite yes | no типы_пользователей	Разрешает или запрещает пользователям указанных типов перезаписывать файлы на сервере. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)
rename yes | no типы_пользователей	Разрешает или запрещает пользователям указанных типов переименовывать файлы на сервере. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)
password-check rfc822| trivial | none enforce|warn	Задает уровень проверки пароля. При этом в качестве первого параметра указывается метод проверки пароля: none — отключает проверку паролей; trivial — все пароли должны обязательно содержать символ '@'; rfs822 — в качестве паролей должны указываться адреса электронной почты, задаваемые согласно стандарту RFC822 (рекомендую использовать именно это значение). Вторым параметром задается действие, которое должно производиться в тех случаях, когда пользователь введет неправильный пароль. Значение warm говорит о том, что пользователь просто будет проинформирован об ошибке в пароле и далее ему будет позволено заново зарегистрироваться на FTP-сервере. Если указать значение enforce, то пользователю будет выдано сообщение о неправильном пароле и ему в дальнейшем будет запрещен доступ к серверу
upload yes | no типы_пользователей	Разрешает или запрещает выгрузку файлов на сервер пользователям указанных типов. В качестве типов пользователей указываются ключевые слова anonymous, guest, real (см. описание опции class)

11.1.2. Файл ftphosts

Файл ftphosts используется для разрешения или запрещения доступа определенных пользователей с указанных узлов. Например, вы можете разрешить доступ пользователю admin только с компьютера *admm.domain.ru* и запретить со всех остальных. А для других пользователей разрешить доступ со всех компьютеров. Таким образом, в файле могут быть записи двух видов: разрешающие и запрещающие. Формат записей в файле `ftphosts` следующий:

```
allow | deny user hosts [host...]
```

Разрешающая запись **allow** разрешает пользователю регистрироваться с хостов, указанных в списке **hosts,** но запрещает регистрацию со всех остальных. Запись **deny,** наоборот, запрещает доступ с определенных хостов, но разрешает со всех остальных. В листинге 11.4 приведен пример файла ftphosts.

Листинг 11.4. Пример файла ftphosts

```
allow admin 192.168.1.1
deny user 192.168.1.2 192.168.1.3
```

В приведенном примере пользователь admin может регистрироваться на сервере только с компьютера с IP-адресом 192.168.1.1. Если этот пользователь попробует зарегистрироваться с другого компьютера, то ему будет отказано в доступе. Пользователю user запрещено регистрироваться с компьютеров 192.168.1.2 и 192.168.1.3, но он может зарегистрироваться с любого другого компьютера сети.

11.1.3. Файл ftpusers

Файл ftpusers содержит список пользователей, которым запрещено использовать команду **ftp.** Эти пользователи не могут зарегистрироваться на сервере. При попытке регистрации будет выведено сообщение об ошибке **Login Incorrect,** даже если пользователь ввел правильный пароль. Из соображений безопасности этот файл должен содержать хотя бы имена пользователей root, bin, news, uucp. Пустые строки, а также строки, начинающиеся с символа #, игнорируются. Полностью корректный с точки зрения безопасности файл представлен в листинге 11.5.

Листинг 11.5. Файл ftpusers

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

11.1.4. Файл ftpgroups

Файл ftpgroups содержит специальные пароли, с помощью которых пользователи FTP будут рассматриваться как пользователи определенной группы. Такое разрешение получит пользователь, для которого запись в файле `ftpgroups` содержит выражение *yes*. Более подробную информацию об этом файле вы можете получить в справочной системе, введя команду **man ftpgroups.**

11.1.5. Файл ftpconversions

В файле `ftpconversions` заданы операции сжатия, разрешенные пользователям для выполнения во время сеанса FTP. Стандартный файл ftpconversions представлен в листинге 11.6.

Листинг 11.6. Файл ftpconversions

```
:.Z:   :   :/bin/compress -d -c
%s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
:    : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
:.gz:  :   :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
:    : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
:    : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
:    : :.tar.Z:/bin/tar -c -Z -f —
%s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
:    : :.tar.gz:/bin/tar -c -z -f -
%s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP
```

Каждая запись этого файла состоит из восьми полей. Поля разделяются с помощью символа двоеточия. Эти поля содержат префиксы и постфиксы удаления и добавления, внешнюю команду, тип операции и описание. Например, сжатый программой **gzip** файл должен иметь имя с суффиксом **gz.** Чтобы к имени файла был добавлен суффикс gz, запись в файле ftpconversions должна иметь постфикс gz. Я специально не использую здесь термин «расширение», так как он отсутствует в терминологии UNIX, потому что файлы в ОС Linux не имеют расширения.

11.1.6. Файл xferlog

Файл `xferlog` является журналом сервера FTP, в который записываются все транзакции, которые были произведены в результате работы пользователей. С помощью опций **–o** и **–i** сервера FTP можно выбрать тип транзакций, записываемых в журнал (см. табл. 11.1).

Примечание.

Транзакция — это обработка запроса, то есть прием порции данных от пользователя, ее обработка и выдача результата пользователю. В базах данных и файловых системах — это выполнение элементарной целостной операции над данными, например, удаление записи.

Рассмотрим листинг 11.7, в котором представлен фрагмент файла xferlog.

Листинг 11.7. Фрагмент файла xferlog

```
Wed Jan  9 11:49:35 2002 1 localhost.localdomain 1490 /home/
den/vmware.html a _ o r den ftp 0 * c
Wed Jan  9 11:50:08 2002 1 localhost.localdomain 281 /home/den/
w.out a _ o r den ftp 0 * c
Wed Jan  9 11:50:15 2002 1 localhost.localdomain 281 /home/den/
w.out a _ i r den ftp 0 * c
Wed Jan  9 11:52:08 2002 1 localhost.localdomain 888 /home/den/
ftphosts.html b _ i r den ftp 0 * c
```

Теперь проанализируем записи. Из первой записи вы можете узнать, что пользователь *den* был зарегистрирован с удаленного узла localhost.localdomain. Начало передачи файла `/home/den/vmware.html` произошло в среду, 9 января 2002 года в 11:49. Общее время передачи — одна секунда. Общий объем переданной информации составляет 1490 байт. Для передачи файла использовался режим ASCII (a), не было произведено никаких специальных операций (_). Файл vmware.html пользователь загружал с сервера, на что указывает направление передачи (o). Пользователь *den* является реальным (зарегистрированным) пользователем системы (r). Название службы, которая производила операцию — ftp.

Теперь рассмотрим четвертую запись. Тот же пользователь *den* передал на сервер файл `ftphosts.html`. Направление передачи -- на сервер (i). Режим передачи — двоичный (b). Вторая и третья записи сообщают о загрузке с сервера и на сервер файла `w.out` в текстовом (a) режиме.

Тип пользователя обозначается символом возле имени пользователя. Символ r обозначает зарегистрированного в системе пользователя, у которого есть своя запись в файле /etc/passwd. Символ g означает гостевую регистрацию, а символ a — анонимную.

Направление передачи, как вы уже догадались, обозначается символами o — прием файла пользователем и i — прием файла сервером. Остается только отметить, что файл `xferlog` используется обоими серверами FTP - **wu-ftpd** и **ProFTPD**.

11.2. Сервер ProFTP

Альтернативой, и, на мой взгляд, достаточно хорошей, сервера wu-ftpd является сервер ProFTPD. Он намного проще в плане конфигурирования, чем сервер wu-ftpd, и обладает достаточно гибкими возможностями. Для его установки достаточно установить пакет **proftpd**. Подобно серверу wu-ftpd, ProFTPD может запускаться автоматически при запуске системы или вызываться суперсервером при наличии запроса на установку соединения. Первый режим называется standalone.

Сервер ProFTPD может вызываться с параметрами, указанными в табл. 11.4.

Параметры сервера ProFTPD *Таблица 11.4*

Параметр	Описание
-h	Справочная информация
-n	Запускает сервер в автономном режиме. Для этого в файле конфигурации нужно указать режим запуска standalone (см. ниже)
-d уровень_отладки	Устанавливает уровень отладки сервера (1-5)
-c файл_конфигурации	Задает использование альтернативного файла конфигурации вместо стандартного /etc/proftpd.conf
-p 0 | 1	Запрещает (0) или разрешает (1) использование постоянного пароля. Для получения более подробной информации (смотрите документацию по серверу)
-l	Выводит список всех модулей, откомпилированных для использования сервером ProFTPD
-v	Выводит версию

11.2.1. Файл /etc/proftpd.conf

Сервер ProFTPD использует всего один файл конфигурации – /etc/proftpd.conf, который по своей структуре чем-то похож на файл конфигурации Web-сервера Apache (а именно — директивами конфигурирования).

В листинге 11.8 представлен простейший файл конфигурации сервера ProFTPD.

Листинг 11.8. Пример файла конфигурации /etc/proftpd.conf

```
# Этот файл устанавливает один сервер и одну
# учетную запись
ServerName "My ProFTPD server"
ServerType  standalone
DefaultServer on
# Используем стандартный порт
Port 21
Umase 022
MaxInstances 30
# Пользователь и группа, обслуживающие сервер
User nobody
Group nobody
# Параметры корневого каталога. Блочная директива Directory
<Directory /*>
# Директива, определяющая параметр AllowOverwrite
  AllowOverwrite on
</Directory>
```

Директивы конфигурации делятся на две группы: директивы, определяющие некоторые параметры, и блочные директивы. Блочные директивы конфигурирования похожи на тэги языка HTML. С помощью блочных директив задаются блоки, содержащие директивы, которые определяют параметры (см. листинг 11.8). При этом используются начальные и конечные блочные директивы. Конечная директива имеет то же имя, что и начальная, но с наклонной чертой в начале. Например, начальная директива <Directory /*>, а конечная — </Direcory> (см. листинг 11.8). Действия каждой пары директив распростра-

няются только на блок, который они задают (который расположен между ними). Директива **<Directory>** определяет свойства какого-нибудь каталога. В вышеприведенном листинге (листинг 11.8) определяются свойства корневого каталога.

В табл. 11.5 представлены все директивы файла конфигурации сервера **ProFTPD**.

Директивы файла конфигурации сервера ProFTPD *Таблица 11.5*

Директива	Описание
AccessGrantMsg сообщение	Ответное сообщение, которое будет отправлено пользователю в случае его регистрации или получения анонимного доступа. Символы %и будут заменены на имя пользователя, которое он ввел при регистрации
Allow from all | host | network [,host | *network*[, ...]]	Используется внутри блока Limit. Ограничивает доступ к серверу (а именно разрешает доступ). По умолчанию allow from all
AllowAll	Разрешает доступ к блокам Directory, Anonymous, Limit
AllowForeignAddress on | off	Разрешает клиенту указывать при соединении соединения адрес, который не соответствует ему. По умолчанию off. Может использоваться в блоках VirtualHost, Anonymous, <Global>
AllowGroup список_групп	Разрешает доступ определенным группам. Используется в блоке Limit
AllowUser список_пользователей	Разрешает доступ определенным пользователям. Используется в блоке Limit
AnonRequirePassword on | off	Требует пароль при анонимной регистрации. Пароль должен совпадать с паролем того пользователя, который запустил демон. По умолчанию данная опция выключена
<Anonymous directory>	Создает анонимную учетную запись, directory — корневой каталог анонимного сервера
AuthGroupFile путь	Позволяет указать путь к альтернативному файлу group. По умолчанию используется файл /etc/group
AuthUserFile путь	Указывает альтернативный файл passwd
Bind Ip-адрес	Разрешает привязку дополнительного IP-адреса к основному или виртуальному хосту
DefaultRoot каталог	Задает корневой каталог по умолчанию
Deny from all | host | network	Запрещает доступ к серверу. Используется в блоке Limit
DenyAll	Запрещает анонимным пользователям доступ к объектам, указанным в блоке Limit
DenyUser список_пользователей	Запрещает доступ определенным пользователям
<Directory> путь	Используется в VirtualHost, Anonymous для того, чтобы определить особенные параметры доступа к каталогу и его подкаталогам
DisplayFirstChdir файл_сообщения	Указанный текстовый файл будет выводиться, когда пользователь впервые за время сеанса войдет в данный каталог. Используется в VirtualHost, Directory, Anonymous
DisplayLogin файл_сообщения	Этот файл будет отображен, когда пользователь зарегистрируется
<Global>	Используется для задания параметров, которые будут использоваться как основным, так и всеми виртуальными серверами
<Limit command>	Ограничение на выполнение данной FTP-команды, например, LOGIN, WRITE, READ, STOR
MaxClients number | none сообщение	Ограничение на количество клиентов. Приведенное сообщение будет отображено, если пользователю будет отказано в доступе. Блоки Anonymous, Global
MaxLoginAttempts	Максимальное количество попыток зарегистрироваться. По умолчанию 3. Блоки VirtualHost, Global
Order allow, deny | deny, allow	Порядок выполнения директив Allow и Deny в блоке Limit
PersistentPassword on | off	При значении on будут использованы системные файлы /etc/passwd и /etc/group, несмотря на то, что командой enroot корневой каталог был изменен
RequireValidShell on | off	Разрешает или запрещает регистрацию при использовании оболочек (shells), которые не указаны в файле /etc/shells

Продолжение табл. 11.5

Директива	Описание
ServerAdmin email	Определяет email администратора сервера
ServerType	Определяет режим работы сервера standalone (по умолчанию) или inetd. В первом случае сервер будет запускаться автоматически из стартовых сценариев системы, во втором — его будет запускать сервер inetd при попытке соединения
TimeoutIdle секунды	Время в секундах, в течение которого пользователь имеет право не проявить активности. По умолчанию 60 (1 минута)
Umask маска	Определяет права доступа для созданного файла. Маска — число в восьмеричной системе, определяющее набор прав доступа (см. главу 4)
User имя_пользователя	Имя пользователя, присвоенное демону ProFTP
UserAlias псевдоним пользователь	Создает указанный псевдоним для указанного пользователя
<VirtualHost address>	Создает виртуальный сервер

11.2.2. Ограничение доступа

Я считаю необходимым подробно рассмотреть блочную директиву **Limit.** Эта директива определяет вид и параметры доступа к тому или иному каталогу. Рассмотрим листинг 11.9.

Листинг 11.9. Пример использования директивы Limit

```
<Directory  incoming>
    <Limit  WRITE>
        AllOwAll
    </Limit>
    <Limit  READ>
        DenyAll
    </Limit>
</Directory>
```

Директива **Directory** определяет свойства каталога incoming, а директива **Limit** задает вид доступа к этому каталогу. Команда WRITE директивы **Limit** вместе с директивой **AllowAll** разрешает всем пользователям записывать информацию в этот каталог. Команда READ директивы **Limit** задает ограничение на чтение этого каталога. В рассматриваемом случае чтение запрещено для всех пользователей. Кроме команд WRITE и READ в директиве Limit можно задавать команды STOR и LOGIN (см. табл. 11.6).

В блоке **Limit** можно задавать директивы **Allow, AllowAll, AllowGroup, AllowUser, Deny, DenyAll, DenyUser** (см. табл. 11.5). Например, в листинге 11.10 запрещается доступ всем пользователям, кроме den. Пользователь den может регистрироваться со всех компьютеров, кроме компьютера с IP-адресом 111.111.111.111. Также запрещена регистрация из сети 192.168.2.0

Команды директивы Limit, ограничивающие права доступа *Таблица 11.6*

Команда	Описание
LOGIN	Ограничивает регистрацию
WRITE	Ограничивает запись
READ	Ограничивает чтение
STOR	Ограничивает прием файлов

Листинг 11.10. Пример блока Limit

```
<Limit LOGIN>
    DenyAll
    AllowUser den
    Deny from 111.111.111.111
    Deny from 192.168.2.
</Limit>
```

11.2.3. Файл .ftpaccess

Для конфигурирования отдельного каталога может также использоваться файл `.ftpaccess`, который расположен в этом каталоге. В нем содержатся такие же директивы, что и в файле `proftpd.conf`, но файл `.ftpaccess` имеет приоритет перед файлом `proftpd.conf`.

11.3. Организация анонимного FTP-сервера

Анонимный FTP-сервер можно построить с помощью **wu-ftpd**, установив пакет anonftp. Этот пакет нельзя использовать вместе с сервером ProFTPD. Пакет **anonoftp** поставляется в составе большинства дистрибутивов.

Сейчас рассмотрим, как организовать анонимный FTP-сервер с помощью сервера ProFTPD. Для организации анонимного доступа сервер ProFTPD имеет директиву **Anonymous.** При этом в блок **Anonymous** нужно поместить директивы, конфигурирующие анонимную службу. В самой же директиве **Anonymous** необходимо указать каталог, который будет использоваться в качестве корневого для анонимной службы. Сервер ProFTPD выполнит для этого каталога команду **enroot,** превращая этот каталог в корневой для удаленного пользователя. Перед тем, как сделать это, сервер ProFTPD прочитает все необходимые ему файлы конфигурации из реального каталога /etc.

При анонимной регистрации, по умолчанию в качестве имени пользователя указывается anonymous, а вместо пароля — адрес электронной почты пользователя. Вы же можете изменить параметры анонимного доступа, добавив проверку пароля для анонимного пользователя с помощью директивы **AnonRequirePassword** (см. табл 11.5). В следующем примере представлен типичный блок **Anonymous,** подходящий для большинства анонимных серверов (см. листинг 11.11.)

Листинг 11.11. Типичный блок Anonymous

```
<Anonymous /var/ftp>
  User ftp
  Group ftp
  UserAlias anonymous ftp
  RequireValidShell off
  <Directory *>
    <Limit WRITE>
        DenyAll
    </Limit>
    <Limit STOR>
```

```
        AllowAll
    </Limit>
  </Directory>
</Anonymous>
```

Директивы **User и Group** задают имя пользователя для анонимного доступа. В обоих случаях применяется имя **ftp.** Для имени **ftp** определяется псевдоним *anonymous.* Вместо пароля нужно указать адрес электронной почты.

Директива **RequireValidShell** отключает проверку командного интерпретатора пользователя. По умолчанию сервер ProFTPD ищет список допустимых интерпретаторов в файле /etc/shells. Если используемый пользователем интерпретатор не указан в файле /etc/shells, то соединение будет разорвано. Директива **RequireValidShell off** отключает такую проверку.

Директива **<Directory *>** определяет свойства для всех каталогов. При этом всем пользователям запрещено записывать файлы на сервер, но разрешено сохранять файлы сервера на свой локальный компьютер.

Желательно также добавить в блок **Anonymous** директиву **MaxClients,** которая указывает максимальное число клиентов. Нужно учитывать нагрузку на сервер и пропускной канал для определения максимального числа анонимных клиентов. Настоящих пользователей сервера FTP по возможности не следует ограничивать, в отличие от анонимных. При малоскоростном канале связи, например, 33 Кбит/с, установите маленькое максимальное количество анонимных клиентов, например, 5 или даже 3. Конечно, число клиентов также зависит от объема информации, расположенной на сервере. Если размеры файлов небольшие, например, документация, число клиентов можно установить несколько большим (10...15).

11.4. Вспомогательные программы

При работе с серверами **wu-ftpd и ProFTPD** вы можете использовать программы **ftpshut, ftpwho, ftpcount.** Инструментальные средства обоих серверов имеют похожие опции, но вспомогательные программы для **ProFTPD** выводят больше полезной информации.

Программа **ftpshut** останавливает сервер. Программа **ftpwho** выводит информацию о пользователях, подключенных к вашему серверу. Программа **ftpcount** сообщает о количестве установленных соединений.

Я рекомендую использовать расширенный вывод программы **ftpwho.** В этом режиме предоставляется больше информации (см. листинг 11.12).

Листинг 11.12. Результат работы программы ftpwho

```
# ftpwho -v
Master proftpd process 759:
 1113 2m55s  proftpd: ftp — localhost.localdomain: anonymous/
den@den.com: IDLE
     (host: localhost.localdomain [127.0.0.1])
     (cwd: /)
 1150 0m20s  proftpd: den -- localhost.localdomain: IDLE
```

```
    (host: localhost.localdomain [127.0.0.1])
    (cwd: /home/den)
Service class              2 users
```

Программа ftpcount только сообщает об общем количестве пользователей (см. листинг 11.13).

Листинг 11.13. Результат работы программы ftpcount

```
Master proftpd process 759:
Service class              2 users
```

11.5. Виртуальный узел FTP

В этом разделе книги пойдет речь о конфигурировании виртуального FTP-сервера. Создание виртуального FTP-сервера будет происходить с использованием сервера **ProFTP**.

Для начала определимся, что такое **виртуальный сервер?** Как и подразумевает слово «виртуальный», такой сервер не будет существовать физически, но пользователь будет считать, что он работает с реальным сервером. ОС Linux может поддерживать несколько IP-адресов, благодаря чему имеется возможность создать виртуальные узлы. Если вы располагаете дополнительными IP-адресами, то они как раз могут использоваться для создания виртуальных узлов. При конфигурировании виртуальных FTP-узлов каждому из них нужно присвоить отдельный IP-адрес.

Виртуальные FTP-узлы нужны, если вы, например, хотите организовать несколько узлов FTP, один — для одной рабочей группы, другой — для второй, третий может быть анонимным и так далее. Обслуживать сразу несколько FTP-узлов позволяет все тот же демон **proftpd.**

Настройка виртуального FTP-узла очень похожа на настройку виртуального Web-сервера, настройка которого будет рассмотрена в следующей главе. Даже используется одна и та же директива **VirtualHost**. При этом, для конфигурирования виртуального FTP-сервера, в файл `proftpd.conf` нужно добавить директиву **VirtualHost,** содержащую IP-адрес (см. листинг 11.14).

Листинг 11.14. Директива VirtualHost

```
<VirtualHost 192.168.1.5>
 ServerName "Virtual FTP Server"
</VirtualHost>
```

В блоке **VirtualHost** можно использовать другие директивы, например, директиву **Anonymous,** которая создаст гостевой узел. Можно также задать каталог или порт (см. листинг 11.15).

Листинг 11.15. Пример использования директивы VirtualHost

```
<VirtualHost ftp.library.com>
    ServerName  "Online library"
    MaxClients  15
    MaxLoginAttempts    1
    DeferWelcome    on
 <Limit LOGIN>
    Allow    from    192.168.1
```

```
    Denyfromall
  </Limit>
  <Limit WRITE>
    AllowUser libadmin
    DenyAll
  </Limit>
  <Anonymous  /var/ftp/library/books>
    User          library
    Group             library
    AnonRequirePassword on
</Anonymous>
  <Anonymous /var/ftp/library>
    User          ftp
    Group             ftp
    UserAlias   anonymous     ftp
  </Anonymous>
</VirtualHost>
```

В листинге 11.15 приведена конфигурация виртуального сервера ftp.library.com. В директиве **VirtualHost** используется доменное имя, которое **должно быть прописано** в сервере DNS. IP-адрес должен указывать на узел сети, на котором запущен демон **ProFTPD**. В листинге 11.15 также конфигурируются две анонимных учетных записи — **library** и **ftp.** Причем учетная запись library требует ввода пароля при регистрации. Пароль должен совпадать с паролем того пользователя, который запустил демон. Доступ к виртуальному серверу разрешен только для подсети 192.168.1.0. Записывать данные на сервер может только пользователь libadmin.

В качестве FTP-клиента для X Window я рекомендую использовать программу **gFTP**, которая обладает довольно удобным интерфейсом и богатыми функциями. Использование **gFTP** настолько простое, что я не буду останавливаться на его рассмотрении (см. рис. 11.2).

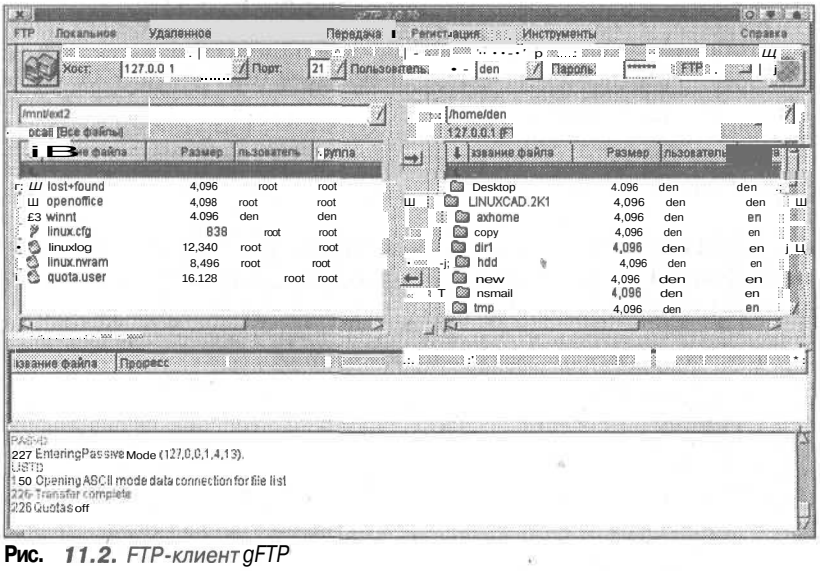


Рис. ***11.2.*** *FTP-клиент gFTP*

12

Сервер Apache

Эта глава посвящена популярному WWW-серверу **Apache.** Сервер **Apache** разработан и поддерживается организацией Apache Project. Первоначально сервер Apache был разновидностью сервера Web-сервера NCSA, разработанного в Национальном центре разработок суперкомпьютеров Иллинойского университета. Возникновение Apache было связано с тем, что в 1994 году ушел из проекта главный разработчик NCSA, оставив многих последователей самостоятельно разбираться в своем сервере. Со временем начали появляться исправления и дополнения к серверу NCSA -- так называемые patches (патчи). А в апреле 1995 года вышла первая версия сервера Apache, основанного на версии 1.3 сервера NCSA. Первая версия Apache просто вобрала в себя все известные исправления сервера NCSA. Да и само название Apache именно от этого — «А PatCHy». Позже Apache стал самостоятельной разработкой. Сейчас сервер Apache поддерживается группой программистов-добровольцев Apache Group.

Сервер Apache разрабатывался для ОС Linux и Unix, но со временем были выпущены его версии и для ОС Windows, и OS/2.

Хочу также отметить, что кроме Apache, для ОС Linux существуют другие Web-серверы: Red Hat Secure Server, Apache-SSL, Netscape Enterprise Server и др.

12.1. Установка Apache

Для установки сервера Apache необходимо установить пакеты **apache и apache-docs.** В первом из них находится сам сервер, а во втором — документация. Желательно устанавливать самую новую версию. В последних версиях вам нужно установить еще и пакет **apache-common,** содержащий необходимые файлы для запуска сервера. Проще всего установку Apache можно произвести, введя следующую команду:

```
rpm -ih apache*
```

После установки сервер конфигурируется для запуска в режиме **standalone,** то есть он будет постоянно находиться в памяти. Я не рекомендую изменять этот режим. Для запуска и останова сервера Apache вы можете воспользоваться командами:

```
/etc/rc.d/init.d/httpd start
/etc/rc.d/init.d/httpd stop
```

После успешной установки сервера отредактируйте файл /etc/httpd /conf/httpd.conf. В нем исправьте всего одну директиву — **ServerName**. При этом, на данном этапе (пока новое имя сервера Apache не зарегистрировано на вашем сервере DNS), вместо того имени, которое будет использоваться в дальнейшем (например, www.host.domain), установите обыкновенное имя вашего сервера, например, server.firma.ru. Данное имя должно быть зарегистрировано на DNS-сервере вашей сети. После этого запустите сервер. Затем откройте любой браузер и попробуйте обратиться к серверу (см. рис. 12.1):

```
netscape http://localhost
```

Теперь попробуйте обратиться к этому серверу с другого компьютера вашей сети:

```
netscape http://server.firma.ru
```

Если вы увидите приветствие сервера в первом и во втором случаях, значит, ваш сервер Apache нормально работает и можно приступать к его дальнейшему конфигурированию. Если в первом случае у вас произошла

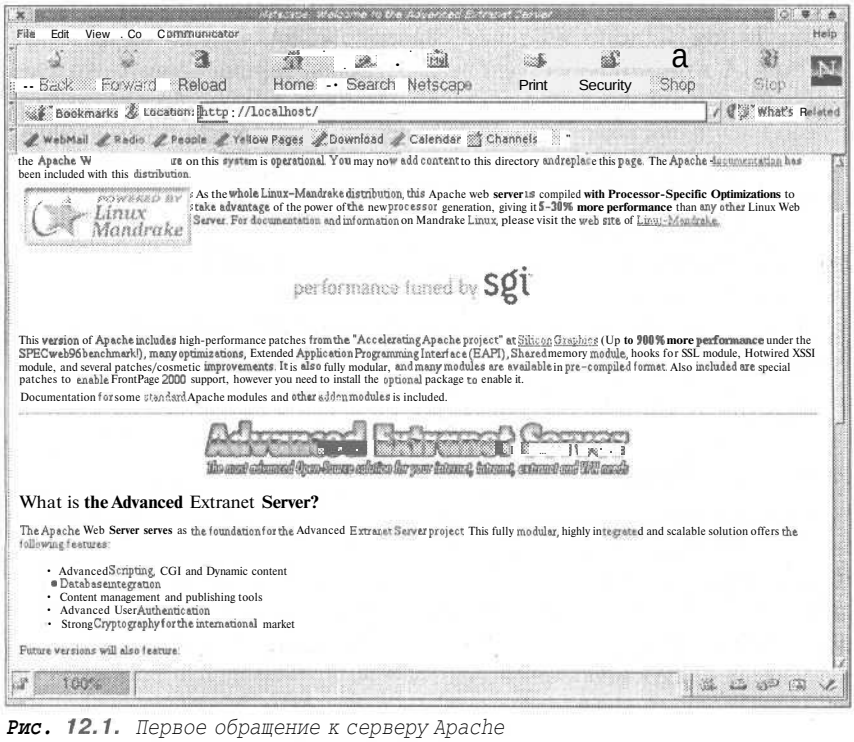


Рис. 12.1. *Первое обращение к серверу Apache*

ошибка, значит, искать ее нужно на локальном уровне. При этом, если сеть нормально работает, то, скорее всего, вы просто забыли запустить сервер. Появление ошибки во втором случае может быть связано с неправильной установкой директивы **ServerName** или же используемое вами имя не прописано в системе DNS.

72.2. *Файлы конфигурации сервера*

Сервер Apache имеет три конфигурационных файла: httpd.conf, srm.conf, access.conf. Обычно эти файлы находятся в каталоге /etc/httpd/conf. Вся настройка сервера заключается в редактировании этих трех файлов. Рассмотрим, какие функции выполняют эти файлы.

Файл httpd.conf — это основной файл конфигурации сервера. В нем содержится техническое описание работы сервера. В файле srm.conf задаются параметры документов, которые размещены на сервере. Файл access . conf содержит параметры доступа к серверу.

Начиная с версии 1.3, рекомендуется все директивы, которые раньше находились в файлах srm.conf и access.conf, помещать в файл httpd.conf. Я использую сервер Apache версии 1.3.14-2, который входит в состав дистрибутива Linux Mandrake 7.2. В этой версии существует еще несколько конфигурационных файлов: apache-mime.types, vhosts/vhosts.conf, vhosts/VirtualHomePages.conf, vhosts/DynamicVHosts.conf. В файле apache-mime.types содержатся типы MIME, поддерживаемые сервером Apache. Файлы vhosts.conf, VirtualHomePages.conf, DynamicVHosts.conf относятся к конфигурированию виртуальных Web-серверов, о которых речь пойдет немного позже.

Примечание.

MIME *(Multipurpose Internet Mail Extensions) — многоцелевое расширение электронной почты в сети Интернет. Используется не только при работе с электронной почтой, но и служит для описания различных типов данных, например, текстовых, графических. Описание типа MIME включает в себя наименование типа, подтипа и расширение (например, text/plain txt).*

12.2.1. Файл httpd.conf

Как уже отмечалось ранее, этот файл содержит практически все директивы, необходимые для работы сервера. Директивы конфигурационного файла сервера Apache можно условно разделить на такие группы:

1. **Общие.** К общим директивам относятся глобальные директивы, влияющие на работу всего Web-сервера. Это директивы ServerName, ServerType, Port, User и Group, ServerAdmin, ServerRoot, PidFile, DocumentRoot, UserDir.
2. **Директивы протоколирования:** ErrorLog, TransferLog, HostnameLookups.
3. **Директивы ограничения доступа:** AllowOverride, Options, Limit.
4. **Директивы управления производительностью:** StartServers, MaxSpareServers, MinSpareServers, а также директива CacheNegotiatedDocs.
5. **Директивы обеспечения постоянного соединения с клиентом:** Timeout, KeepAlive, KeepAliveTimeout.

6. **Директивы настройки отображения каталога.** Оформить отображение каталогов можно с помощью директив настройки отображения каталогов: DirectoryIndex, FancyIndexing и AddIconByType.
7. **Директивы обработки ошибок.** Директивой обработки ошибок HTTP-сервера является директива ErrorDocument. С ее помощью можно установить реакцию на любую ошибку сервера, например, на ошибку 404 (документ не найден).
8. **Директивы перенаправления:** Redirect, Alias и ScriptAlias.
9. **Директивы для работы с многоязычными документами:** AddLanguage и LanguagePriority.
10. **Директивы обработки MIME-типов.** Настроить свой сервер для обработки различных MIME-типов можно с помощью директив DefaultType, AddEncoding, AddType, AddHandler и Action.
11. **Директивы создания виртуальных узлов:** VirtualHost, Listen, BindAddress.

Все эти директивы редактировать вам вряд ли придется — нужно будет задать только значения директив **ServerName и ServerAdmin.** По умолчанию остальные директивы содержат вполне разумные значения. Далее приведено описание директив, используемых в файле `httpd.conf`.

ServerName — директива, которая определяет имя сервера Apache. Причем, здесь должно быть задано официальное имя сервера в таком виде, в котором оно должно появляться в адресной строке браузера. Данное имя должно быть зарегистрировано на сервере DNS вашей сети.

ServerType — директива, которая определяет тип сервера. По умолчанию используется значение **standalone.** Если вы хотите достичь максимальной производительности вашего Web-сервера, не изменяйте эту опцию.

Сервер Apache для каждого соединения запускает отдельную копию, которая будет обрабатывать запросы клиента. Управлять запущенными копиями позволяют директивы **StartServers, MinSpareServers, MaxSpareServers.**

StartServers, MaxSpareServers, MinSpareServers. Как уже отмечалось выше, для каждого нового соединения создается новая копия процесса сервера. Директива StartServers задает количество копий, которые будут созданы при запуске исходной копии сервера. При этом исходная копия сервера получает запросы и передает их свободным копиям. Это позволяет равномерно распределить нагрузку между отдельными процессами и повысить производительность сервера, однако на практике все не так хорошо, как хотелось бы. Существенного прироста производительности можно добиться только в случае большой загрузки сервера. По умолчанию запускается пять копий сервера.

Если число поступающих запросов превышает количество запущенных копий сервера, запускаются дополнительные процессы-серверы. Эти процессы не завершаются после обработки своего запроса, а продолжают находиться в памяти. Директива **MaxSpareServers** позволяет указать максимальное число таких процессов. Если это количество превышено, то лишние процессы завершаются. Если количество «серверов на подхвате» меньше, чем задано директивой **MinSpareServers,** запускаются дополнительные копии.

Для работы этих директив необходимо, чтобы сервер был запущен в автономном режиме.

Port — директива, задающая номер порта, который будет использоваться для установки соединения. По умолчанию используется порт 80. Если вы хотите запустить сервер Apache с использованием этого или любого другого порта, номер которого меньше 1024, вы должны обладать правами суперпользователя. Но даже если у вас нет таких прав, вы можете запустить сервер для работы с портом, номер которого превышает значение 1024. Обычно используется номер 8080 или 8000.

HostnameLookups on | off. Сервер Apache ведет журнал доступа других компьютеров. Если вы включите данную опцию **(on),** то в журнал будет записано доменное имя компьютера-клиента. Если эта опция выключена **(off),** в журнал будет записан IP-адрес клиента. Включение данной опции замедляет работу сервера, так как требуется дополнительное время на ожидание ответа от сервера DNS.

User и Group. Директивы **User и Group** определяют идентификаторы пользователя и группы, от имени которых будет работать сервер. Данные идентификаторы присваиваются серверу, если он запущен в автономном режиме. Можно использовать как имена пользователей, так и их числовые эквиваленты — UID. По умолчанию используется имя пользователя *nobody.* Из соображений безопасности не рекомендуется изменять это значение и присваивать имя реального пользователя. В этом случае Web-сервер получит доступ только к тем файлам, которые разрешены для чтения всем пользователям. Нужно заметить, что указанный пользователь и группа должны существовать в вашей системе. **Ни в коем случае не запускайте сервер от имени пользователя root!**

ServerAdmin — директива, которая задает электронный адрес Web-мастера вашего Web-узла. Если возникнут какие-то проблемы, связанные с работой сервера, то по этому адресу будет отправлено соответствующее сообщение. Обычно используется значение webmaster@Your_Host.com. Пользователь webmaster, как правило, не существует реально в системе. Для определения имени (псевдонима) webmaster используется файл псевдонимов электронной почты /etc/aliases. Данный файл используется для определения псевдонимов пользователей (см. приложение А). Формат файла /etc/aliases следующий:

```
псевдоним: имя
```

После модификации этого файла нужно ввести команду **newaliases.** Данную команду нужно вводить, зарегистрировавшись в системе как суперпользователь.

ServerRoot — в этой директиве указывается местонахождение файлов конфигурации сервера Apache. По умолчанию для этих целей используется каталог /etc/httpd.

BindAddress. Данная директива используется для поддержки виртуальных хостов и применяется для сообщения серверу, какой IP-адрес следует прослушивать. Значением данной директивы может быть «*» (любой адрес), IP-адрес или полное имя домена.

ErrorLog и TransferLog — эти директивы определяют расположение журналов сервера Apache. Обычно для этих целей используется каталог /etc/httpd /logs, который является ссылкой на каталог /var/log/httpd или на

любой другой. В журнале **errorlog** протоколируются диагностические сообщения, а также сообщения об ошибках, которые порождают CGI-сценарии. В журнале **transferlog** протоколируются запросы клиентов. Если включена директива **HostNameLookups,** то вместо IP-адреса клиентов будут регистрироваться имена компьютеров.

PidFile - - с помощью этой директивы указывается имя файла, в котором исходный процесс сервера будет регистрироваться. Этот файл содержит свой идентификатор процесса (PID). Данную информацию можно использовать для останова или перезапуска сервера при написании собственных сценариев. Данный файл будет создан, только если сервер Apache запущен в автономном режиме.

CacheNegotiatedDocs — данная директива позволяет прокси-серверу, например SQUID, не кэшировать документы, которые не генерируются автоматически, то есть в процессе выполнения различных сценариев. Согласно протоколу HTTP/1.0, сервер Apache с каждым пакетом посылает заголовок **Pragma: no-cache** прокси-серверу, что позволяет отключить кэширование документов (в протоколе HTTP/1.1 вместо **Pragma** используется **Cache-Control).** Если вы включите данную директиву, то вы разрешите прокси-серверу кэшировать документы. К сожалению, далеко не все прокси-серверы отключают кэширование после получения данного заголовка. При написании своего CGI-сценария, вам, скорее всего, придется самому выводить заголовок Pragma (или Cache-Control) и мета-тэги, которые указывают на дату последнего обновления документа.

Timeout — задает промежуток времени в секундах, в течение которого сервер продолжает попытки возобновления приостановленной передачи данных. Значение директивы **Timeout** распространяется не только на передачу, но и на прием данных. Если вам нужно получать большие файлы, рекомендую увеличить данное значение.

KeepAlive — разрешает постоянные соединения, то есть такие соединения, в которых производится более одного запроса за один раз.

KeepAliveTimeOut — данная директива определяет таймаут для постоянного соединения.

MaxClients. Иногда поступающих запросов настолько много, что компьютеру не хватает ресурсов для загрузки новых копий сервера в память и их выполнения. Директива **MaxClients** (значение по умолчанию — 150) определяет максимальное число копий сервера, которые могут выполняться одновременно.

MaxRequestsPerChild. После обработки определенного количества запросов, указанного в директиве **MaxRequestsPerChild,** копия сервера завершается, а вместо нее запускается новая.

Listen — позволяет вам связывать Apache с определенным IP-адресом и (или) дополнительными портами.

DocumentRoot — директива, определяющая местонахождение корневого каталога документов вашего сервера.

UserDir — эта директива задает названия подкаталога в домашнем каталоге пользователя, из которого берутся документы. В этом случае, вы активизи-

руете возможность использования пользовательских каталогов. Если вы не хотите включать эту возможность, укажите **UserDir** DISABLED. Более подробно эта директива будет рассмотрена позже.

DirectoryIndex — позволяет задать название документа, который будет возвращен по запросу, который не содержит имя документа. С помощью данной директивы можно задать несколько имен файлов. Значениями по умолчанию являются *index.html index.php index.htm index.shtml index.cgi Default.htm default.htm index.php3.* Например, если вы введете в строке адреса браузера http://localhost, то будет возвращен один из указанных в директиве **Directorylndex** документов.

FancyIndexing. При получении запроса, не содержащего имя документа, сервер передаст один из файлов, указанных в директиве **Directorylndex.** Если такой файл не существует, клиенту будет возвращено оглавление каталога. При включении директивы **Fancylndexing,** в оглавлении каталога будут использованы значки и описания файлов. Если директива **Fancylndexing** выключена, оглавление будет представлено в более простом виде.

AddIconByType — сопоставляет значок с типом файла. Значок будет использоваться при выводе каталога, если включена директива **Fancylndexing.** Директива **AddIconByType** имеет следующий формат:

```
AddIconByType (TEXT, URL) mime-type
```

Параметр TEXT определяет текстовое описание типа, которое увидят пользователи, использующие текстовый браузер или пользователи, у которых отключено отображение рисунков. Параметр URL определяет адрес значка, а параметр **mime-type** — это тип файла, с которым нужно сопоставить значок. Полный перечень MIME-типов приведен в файле `apache-mime.types`. В качестве имени файла можно задать не только MIME-тип, но и символы, которыми заканчивается имя файла (см. листинг 12.1), но для этого нужно использовать директиву **AddIcon** вместо **AddIconByType.**

Листинг 12.1. Фрагмент файла httpd.conf

```
AddIconByType (VID,/icons/movie.gif) video/*
AddIcon /icons/binary.gif .bin .exe
```

Первая директива в листинге сопоставляет типу **video** значок `/icons/movie.gif`. Вторая директива сопоставляет бинарным файлам ***.bin** и *.exe значок `/icons/binary.gif`. Значок по умолчанию задается директивой **DefaultIcon**.

DefaultType. Если запрашиваемый клиентом тип не соответствует ни одному из MIME-типов, используется MIME-тип, указанный в директиве **DefaultType.**

AddEncoding. Для сокращения времени передачи файла клиентам используется сжатие данных. Браузеры имеют встроенные программы для распаковки, запускаемые при получении архивов определенных MIME-типов. Именно эти MIME-типы и указываются в директиве AddEncoding.

AddLanguage. В большинстве браузеров можно задать предпочитаемый язык. Благодаря этому вы можете предоставлять документы на разных языках.

Директива AddLanguage сопоставляет расширение файла аббревиатуре языка. Для русского языка используется аббревиатура *ru*, для английского — *en*. При этом в корневом каталоге вашего сервера могут находиться несколько индексных файлов на разных языках. Например, для русского языка нужно использовать имя файла `index.html.ru`, а для английского — `index.html.en`.

LanguagePriority. Если на вашем сервере размещены документы на разных языках, то с помощью директивы **LanguagePriority** можно указать приоритеты различных языков. Например, вы установили директиву **LanguagePriority так:**

```
LanguagePriority en ru
```

Клиент вводит в строке адреса своего браузера адрес http://www.server.com//. Если в свойствах браузера имеется возможность задать предпочитаемый язык, то возвращен будет файл на нужном языке, если такой существует. Если браузер клиента не поддерживает такую возможность, будет возвращен файл на языке, имеющим наболее высокий приоритет. В рассмотренном случае (см. пример) это английский язык. Если файл на нужном языке, например, на немецком, не существует, то будет возвращен файл на английском языке. Для того, чтобы сервер поддерживал нужный вам язык, предварительно установите правильное значение директивы **AddLanguage.**

Redirect. Используйте директиву **Redirect,** когда нужно перенести документы в другой каталог или на другой сервер. Например, вам нужно перенести данные из каталога /users/den в каталог /den. Если при этом старый URL-адрес был http://www.host.com/users/den, то новый станет http://www.host.com/den/. Используйте для этого следующую директиву:

```
Redirect /users/den /den
```

Можно также перенаправить запрос на другой сервер:

```
Redirect /users/den/ www.den.domain.com
```

При этом допускается использование как нового, так и старого URL-адреса.

Alias — с помощью директивы Alias можно предоставить доступ не только к файлам, находящимся в каталоге, указанном директивой **DocumentRoot** и его подкаталогах, но и в других каталогах. По умолчанию определен только псевдоним для каталога `/icons`.

ScriptAlias — аналогична директиве **Alias,** но позволяет задать месторасположение каталога для CGI-сценариев.

AddType — с помощью этой директивы можно добавить новый MIME-тип, который не указан в файле `apache-mime.types`.

AddHandler и Action. Директива **AddHandler** позволяет сопоставить определенному MIME-типу какой-нибудь обработчик. А с помощью директивы **Action** можно определить какое-нибудь действие для обработчика. Например, вы можете запустить какую-нибудь программу для обработки файла данного типа. Использование этих директив, я думаю, лучше всего продемонстрировать на примере (см. листинг 12.2).

Листинг 12.2. Применение директив AddHandler и Action

```
AddHandler text/dhtml dhtml
Action text/dhtml /cgi-bin/dhtml-parse
```

ErrorDocument — директива, сопоставляющая коды ошибок сервера URL-адресам на этом же сервере.

Теперь рассмотрим директивы управления доступом к отдельным каталогам. Данные директивы очень похожи на блочные директивы сервера **ProFTPD**, которые обсуждались в предыдущей главе.

Directory — обычно определяет свойства каталога (см. листинг 12.3).

Листинг 12.3. Директива Directory

```
<Directory />
Options Indexes Includes FollowSymLinks
AllowOverride None
</Directory>
```

Свойства каталога можно указывать в директиве **Directory** или в файле .htaccess, который находится в том каталоге, для которого необходимо установить нужные параметры.

В блоке **Directory** могут находиться директивы управления доступом. К ним относятся директивы **AllowOverride, Options, Limit.** Рассмотрим по порядку все эти директивы. Директива **AllowOverride** может принимать значения, указанные в табл. 12.1.

Значения директивы AllowOverride *Таблица 12.1*

Значение	Описание
None	Сервер Apache будет игнорировать файлы .htaccess. Рекомендую установить данную опцию, так как это повысит производительность сервера
All	Пользователи имеют право переопределять в файлах .htaccess глобальные параметры доступа. Из соображений безопасности лучше не использовать этот режим
Options	Разрешает использовать директиву Options
Limit	Разрешает использовать директиву Limit
AuthConfig	Разрешает использование директив AuthName, AuthType, AuthUserFile и AuthGroupFile
FileInfo	Разрешает использовать в файлах .htaccess директивы AddType и AddEncoding

С помощью директивы **Options** можно определить функции сервера, которые будут доступны для использования в определяемом каталоге. Данную директиву можно использовать как в файле `httpd.conf`, так и в файлах `.htaccess`. Допустимые опции для директивы **Options** представлены в табл. 12.2.

Limit *метод.* Директива **Limit** ограничивает доступ к файлам в определенном каталоге.

Параметр **метод** определяет метод передачи, например, GET или POST. Директиву Limit можно использовать внутри блоков **Directory, Location** или в файле `.htacccess`.

Допустимые опции директивы Options *Таблица 12.2*

Опция	Описание
None	Не разрешается использование каких-либо функций
All	Разрешаются все функции
FollowSymLinks	Разрешается использовать символические ссылки. С точки зрения безопасности не рекомендуется использовать этот режим
SymLinksIfOwnerMatch	Разрешается использование символических ссылок, если ссылка указывает на объект, который принадлежит тому же пользователю, что и ссылка
ExecCGI	Разрешается выполнение CGI-сценариев
Indexes	Если эта опция выключена, сервер не будет передавать содержимое каталога при отсутствии файла index.html
Includes	Разрешено использование серверных включений. Рекомендую отключить эту опцию, поскольку это сильно нагружает сервер
IncludesNoExec	Разрешает использование серверных включений, но запрещает запуск из них внешних программ

Примечание.

В интерфейсе CGI (Common Gateway Interface) определены два метода передачи данных пользователя сценарию: Get и Post. Метод передачи определяется в форме передачи данных. Например:

```
<form action=script.cgi method=GET>
```

Сервер Apache позволяет ограничить с помощью директивы Limit передачу данных одним из этих методов.

В блоке **Limit** можно использовать такие директивы: **allow** (разрешить), **deny** (запретить), **order** (порядок), **require** (требуется). Директивы **allow и deny** аналогичны директивам **allow и deny** файла конфигурации сервера ProFTPD. После директивы **allow** следует слово **from,** после которого можно указать IP-адрес, адрес сети, домен или просто имя компьютера. Слово all обозначает все компьютеры. Например, вам требуется запретить доступ всем компьютерам, кроме компьютеров, которые входят в домен ш (см. листинг 12.4).

Листинг 12.4. Директивы allow, deny

```
order deny, allow
deny from all
allow from ru
```

Следующий пример показывает, как разрешить доступ компьютерам только из вашей сети (см. листинг 12.5). Пусть, при этом, ваша сеть имеет адрес 192.168.1.0

Листинг 12.5. Разрешения доступа подсети 192.168.1.0

```
order deny, allow
deny from all
allow from 192.168.1.
```

Директива **order** определяет порядок выполнения директив allow и deny. Кроме значений allow, deny и deny, allow, директива **order** может содержать значение ***mutual-failure.*** В этом случае доступ будет отказан всем компьютерам, которые явно не указаны в списке allow.

Директиву require можно использовать для защиты каталога паролем. После названия директивы должен следовать список элементов: имена пользователей, групп, которые заданы в директивах **AuthUserFile** и **AuthGroupFile**. Можно использовать параметр *valid-user,* который укажет серверу предоставить доступ любому пользователю, имя которого имеется в директиве **AuthUserFile,** если он введет правильный пароль. Пример использования директив Limit, require, AuthUserFile приведен в листинге 12.6.

Листинг 12.6. Использование директивы require

```
<Directory *>
AuthUserFile /var/secure/.htpasswd
AuthName Security
AuthType Basic
<Limit GET>
order deny,allow
deny form all
allow from mydomain.ru
require valid-user
</Limit>
</Directory>
```

В листинге 12.6 для аутентификации используется файл паролей .htpasswd, который можно создать с помощью программы **htpasswd.** Директива **Limit** разрешает доступ к любому каталогу сервера только пользователям домена mydomain.ru.

Кроме параметра **valid-user** допускается использование параметра **users или grpoups**. Данные параметры разрешают доступ только определенным пользователям или группам пользователей. Пример использования параметра **users** приведен в листинге 12.7.

Листинг 12.7. Применения параметра users

```
<Directory /users>
AuthType Basic
AuthUserFile /var/users/.htpasswd
AuthName UsersDir
<Limit GET POST>
 require users denis igor evg
</Limit>
</Directory>
```

Location — с помощью этой директивы можно задать определенный URL-адрес, который предназначен для обозначения каталогов, файлов или групп файлов. Обозначить группу файлов можно с помощью шаблонов, например, шаблон ***.html** определяет все файлы, имена которых заканчиваются на .html. В URL-адрес не включается протокол и имя сервера. Пример описания блока Location представлен в листинге 12.8.

Листинг 12.8. Блок Location

```
<Location URL>
директивы управления доступом
</Location>
```

12.2.2. Конфигурирование Apache с помощью netconf

Практически все параметры Web-сервера Apache можно установить, используя конфигуратор **netconf** (или **linuxconf).** Для этого запустите **netconf** и перейдите на вкладку ***Server Tasks,*** а затем нажмите на кнопку «Apache Web-server» (см. рис. 12.2).

С помощью netconf вы легко можете установить основные параметры Apache (см. рис. 12.3), определить виртуальные хосты, установить параметры подкаталогов, определить спецификацию каталогов и модулей, а также установить параметры модуля **mod_ssl** (см. рис. 12.4), конфигурирование которого рассмотрено ниже в этой главе.

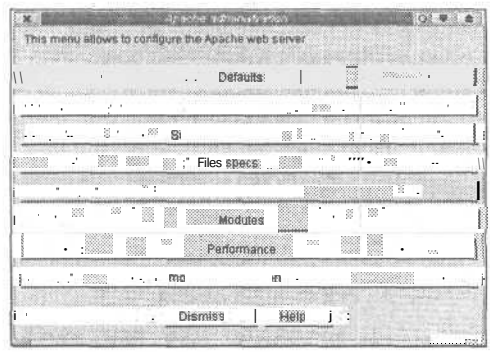


Рис. 12.2. *Конфигурирование Apache с помощью netconf*

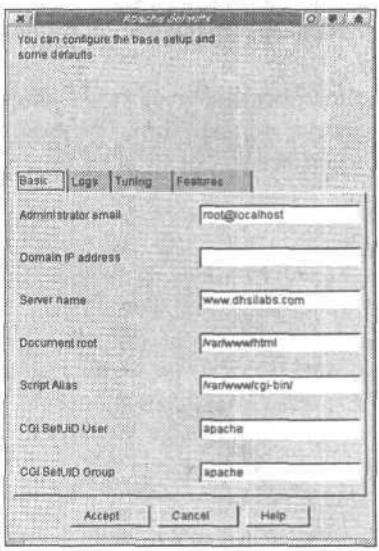


Рис. ***12.3.*** *Основные параметры Apache*

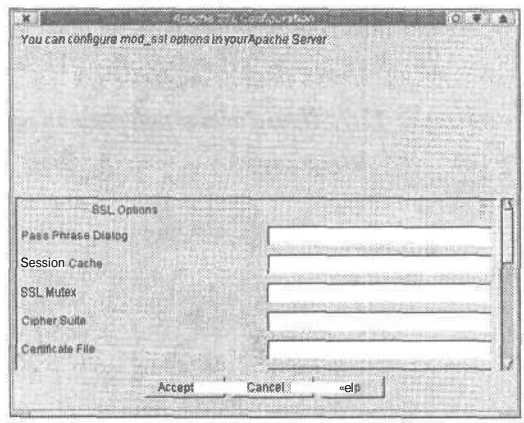


Рис. ***12.4.*** *Конфигурирование модуля mod_ssl*

12.3. Каталоги пользователей

Директива **UserDir** включает поддержку пользовательских каталогов. Эта директива определяет общее название подкаталога в домашних каталогах всех пользователей. По умолчанию используется каталог **public_html**. Данная возможность очень удобна при использовании ее в большой корпорации, где каждый сотрудник имеет собственную страничку. Раньше эта возможность часто использовалась на серверах, предоставляющих бесплатный хостинг. Может быть, помните адреса вида http://www.chat.ru/~mypage? Сейчас же все чаще используется технология виртуальных серверов, которую мы рассмотрим в следующем пункте, но знать что такое каталоги пользователей и как с ними работать тоже не помешает. Тем более, что домашние каталоги настраиваются намного быстрее и проще, чем виртуальный сервер — нужно всего лишь определить директиву **UserDir** и указать месторасположения домашних каталогов.

Доступ к файлам, расположенным в этих каталогах, производится с помощью указания через наклонную черту пользователя после имени сервера. Например, пусть имя сервера www.server.com, имя пользователя — denis, тогда URL-адрес будет выглядеть так: http://www.server.com/~denis/. При этом сервер самостоятельно определит, где именно расположен домашний каталог пользователя. Если домашний каталог пользователя /home/den, то сервер передаст клиенту файл `/home/den/public_html/index.html`.

12.4. Виртуальный HTTP-сервер

Концепция виртуальных хостов позволяет серверу Apache поддерживать несколько Web-узлов. Получается, что один Web-сервер заменяет несколько серверов, и вместо одного узла пользователи видят отдельные Web-узлы. Это очень удобно, если нужно организовать персональные Web-узлы пользователей или собственные Web-узлы подразделений компании, например, develop.mycompany.com.

Сервер Apache можно настроить несколькими способами: чтобы запускался один сервер, который будет прослушивать ВСЕ обращения к виртуальным серверам, или запускать отдельный процесс для каждого виртуального сервера. В первом случае один сервер будет одновременно обслуживать все виртуальные. Если вас интересует такой вариант, нужно настраивать виртуальные сервера с помощью директивы **VirtualHost**. Настройка отдельных процессов для каждого сервера осуществляется с помощью директивы **Listen** и **BindAddress**.

В этом разделе я буду рассматривать именно первый случай. Внутри блока директивы **VirtualHost** можно использовать любые директивы, кроме **ServerType, BindAddress, Listen, NameVirtualHost, ServerRoot, TypesConfig, PidFile, MinRequestPerChild, MaxSpareServers, MinSpareServers**, так как некоторые из них относятся к основному HTTP-серверу (например, ServerType), а некоторые — ко второму варианту настройки виртуальных серверов и здесь неприемлемы. Обязательно должны присутствовать директивы **ServerName, DocumentRoot, ServerAdmin** и **ErrorLog**.

В зависимости от версии и от настроек Apache виртуальные узлы могут прописываться или в файле httpd.conf, или в файле vhosts.conf. Виртуальные серверы можно идентифицировать по имени или по IP-адресу.

12.4.1. Виртуальные серверы с идентификацией по имени

Идентификация по имени имеет существенное преимущество перед идентификацией по IP-адресу: вы не ограничены количеством адресов, имеющемся у вас в распоряжении. Вы можете использовать любое количество виртуальных серверов, и при этом вам не потребуются дополнительные адреса. Такое возможно благодаря использованию протокола HTTP/1.1. Данный протокол поддерживается всеми современными браузерами.

Поддержка виртуальных хостов обеспечивается директивами **VirtualHost** и **NameVirtualHost**. Если ваша система имеет только один IP-адрес, его нужно указать в директиве **VirtualHost.** Внутри блока директивы **VirtualHost** записывается директива **ServerName**. Эта директива задает доменное имя для создаваемого виртуального сервера. Это обязательно нужно сделать, чтобы избежать поиска службой DNS — вы же не хотите, чтобы при неудачном поиске виртуальный сервер был заблокирован? Все директивы **VirtualHost** используют один и тот же IP-адрес, заданный директивой **NameVirtualHost.** В блоке **VirtualHost** записываются параметры виртуального сервера, причем они записываются для каждого виртуального сервера отдельно. Пример приведен в листинге 12.9.

Листинг 12.9. Два виртуальных сервера — www 121- и lib

```
ServerName den.dhsilabs.com
<NameVirtulaHost 192.168.1.1>
<VirtualHost 192.168.1.1>
    ServerName www.dhsilabs.com
    ServerAdmin webmaster@den.dhsilabs.com
    DocumentRoot /var/httpd/www/html
    ErrorLog /var/https/www/logs/error.log
    TransferLog logs/access.log
</VirtualHost>
<VirtualHost 192.168.1.1>
    ServerName lib.dhsilabs.com
    ServerAdmin webmaster@den.dhsilabs.com
    DocumentRoot/var/httpd/lib/html
    ErrorLog /var/https/lib/logs/error.log
    TransferLog logs/access.log
</VirtualHost>
```

Если ваша система имеет только один IP-адрес, доступ к основному серверу станет невозможным, то есть вы не сможете использовать его напрямую. Можно основной сервер использовать в качестве виртуального, что я и сделал в листинге 12.9: основной сервер www является виртуальным. При наличии двух IP-адресов можно один присвоить основному серверу, а другой — виртуальному.

Сервер Apache позволяет использовать несколько доменных имен для доступа к одному серверу, например:

```
ServerAlias www.dhsilabs.com www2.dhsilabs.com
```

При этом запросы, посланные по IP-адресам, которые присвоены вашим виртуальным хостам, должны соответствовать одному из указанных доменных имен. Чтобы зафиксировать запросы, не соответствующие ни одному их этих имен, нужно с помощью опции **default:*** создать виртуальный хост, который будет обслуживать такие запросы:

```
<VitrualHost _default_:*>
```

Обратите внимание на то, что в рассмотренном примере адреса www.dhsilabs.com и lib.dhsilabs.com должны быть прописаны в DNS.

12.4.2. Виртуальные серверы с идентификацией по IP-адресу

В директиве **VirtualHost** в качестве адресов можно использовать доменные имена, но лучше указывать IP-адрес, причем действительный, а не виртуальный. В этом случае вы не будете зависеть от DNS при разрешении имени. Также потребуется один IP-адрес для вашего основного сервера. Если же распределить все адреса между виртуальными серверами, то нельзя будет получить доступ к основному серверу.

Листинг 12.10. Идентификация по IP-адресу

```
<VitrualHost 192.168.1.2>
    ServerName www.dhsilabs.com
    ServerAdmin webmaster@den.dhsilabs.com
    DocumentRoot /var/httpd/www/html
    ErrorLog /var/https/www/logs/error.log
</VirtualHost>
<VirtualHost lib.dhsilabs.com>
    ServerName lib.dhsilabs.com
    ServerAdmin webmaster@den.dhsilabs.com
    DocumentRoot /var/httpd/lib/html
    ErrorLog /var/https/lib/logs/error.log
</VirtualHost>
```

В приведенном примере (см. листинг 12.10) сконфигурированы два виртуальных сервера с идентификацией по IP-адресу. Один из них использует сам IP-адрес, а другой — доменное имя, соответствующее IP-адресу.

При конфигурировании виртуальных серверов можно использовать опцию **ExecCGI,** которая разрешает выполнение CGI-скриптов на виртуальном сервере. Ниже приведен пример для почтового Web-интерфейса (листинг 12.11).

Листинг 12.11. Подключение почтового Web-интерфейса

```
# Файл httpd.conf
<Directory /home/httpd/mail>
    order deny,allow
    deny from all
    allow from localhost
```

```
    allow from 192.168
    allow from 123.123.123.123
    Options ExecCGI
</Directory>
# Файл vhosts.conf
<VirtualHost 123.123.123.123>
    ServerAdmin webmaster@den.dhsilabs.com
    DocumentRoot /home/httpd/mail
    ServerPath /mail
    ServerName wwwmail.dhsilabs.com
    ErrorLog logs/error_log
    TransferLog logs/access_log
    ErrorDocument 403 http://www.dhsilabs.com/messages/
error403.html
</VirtualHost>
# Error 403 — доступ извне, то есть почтовый интерфейс будет
доступен только
# из локальной сети
```

12.5. SSL и Apache

12.5.1. Установка SSL

SSL (Secure Sockets Layer) является методом шифрования, разработанным компанией Netscape для обеспечения безопасности в Интернет. Этот метод поддерживает несколько способов шифрования и обеспечивает аутентификацию как на уровне клиента, так и на уровне сервера. SSL работает на транспортном уровне и поэтому обеспечивает надежное шифрование всех типов данных. Более подробно о реализации SSL можно прочитать на сайте компании Netscape — http://home.netscape.com/info/security-doc.html

Протокол **S-HTTP** является еще одним «безопасным» Интернет-протоколом. Он был разработан для предоставления конфиденциальности данных, передаваемых через соединение. Конфиденциальность нужна, например, при передаче номеров кредитных карточек и прочей важной информации.

Модуль **mod_ssl** реализует в сервере Apache слой SSL, который осуществляет шифрование всего потока данных между клиентом и сервером. Для всех остальных частей Web-сервера модуль **mod_ssl** является прозрачным. Для работы в этом режиме требуется браузер, поддерживающий механизм SSL (этому условию удовлетворяют все современные распространенные браузеры).

Что касается установки, то вам необходимо установить пакет OpenSSL (http://www.openssl.org), хотя возможно у вас в системе уже установлен этот пакет. Для проверки этого введите openssl, и если вы увидите в ответ приглашение OpenSSL>, значит — OpenSSL уже установлен. В противном случае для установки OpenSSL выполните следующие шаги (перед этим выполните команду su для того, чтобы приобрести привилегии суперпользователя):

1. Распакуйте последнюю версию OpenSSL командой:

   tar zxvf openssl-x.y.z.tar.gz (x.y.z — номер версии).

2. Перейдите в каталог openssl-x.y.z и выполните команду:

```
./config.
```

3. Если все нормально (нет ошибок), введите команду:

```
make
```

Примечание.

При возникновении ошибок, скорее всего вам придется установить недостающие пакеты.

4. Затем нужно ввести команды **make install** и **ldconfig**. Перед выполнением команды **Idconfig** убедитесь, что в файле `/etc/ld.so.conf` прописан путь к библиотекам OpenSSL (по умолчанию это `/usr/local/ssl/lib`).

12.5.2. Подключение SSL к Apache

Теперь осталось подключить **mod_ssl** к Apache. При этом следует учитывать, что вам нужна версия **mod_ssl,** которая совместима с вашей версией Apache. Иначе модуль **mod_ssl** будет некорректно работать или вообще откажется что-либо делать. Последние цифры в названии модуля указывают на совместимость с определенной версией Apache. Например, для Apache 1.3.14 нужен файл **mod_ssl-2.7.1-1.3.14.tar.gz.** Здесь 2.7.1 — версия **mod_ssl.**

Распакуйте модуль командой:

```
tar zxvf mod_ssl-x.y.z-2.0.0.tar.gz
```

и выполните команду:

```
./configure –with-apache=../apache_1.3.14 –with-ssl=../openssl-0.9.5
```

В данном примере я использую OpenSSL O.9.5. Теперь перейдите в каталог с Apache, откомпилируйте его и установите сертификат:

```
cd ../apache-1.3.14
make
make certificate
make install
```

Таким образом вы установите Apache в каталог, указанный в опции — **prefix** (по умолчанию `/usr/local/apache`). Теперь попробуйте запустить Apache. Это можно сделать с помощью команды:

```
usr/local/apache/bin/apachectl startssl
```

Параметр **startssl** необходим для включения SSL. Сервер Apache уже функционирует, однако обратиться по протоколу **https** вы еще не можете. Для этого вам нужно сконфигурировать виртуальные хосты, которые будут использовать протокол **https.** Но для начала необходимо настроить Apache для прослушивания порта 443. С этой целью откройте любым редактором файл `/etc/httpd/conf/httpd.conf` и добавьте в него следующие строки:

```
Listen 443
NameVirtualHost x.x.x.x:443
```

Примечание.

Порт 443 — это стандартный порт для протокола HTTPS. Именно по этому порту Apache будет прослушивать виртуальные хосты.

Теперь непосредственно приступите к созданию виртуального сервера, работающего по протоколу https, для чего продолжите редактирование файла /etc/httpd/conf/httpd.conf. Пример того, что необходимо при этом ввести, приведен в листинге 12.12:

Листинг 12.12. Виртуальный https-сервер

```
<VirtualHost x.x.x.x:443>
# Эти строки нужны для поддержки SSL
SSLEngine on
SSLLogLevel warn
SSLOptions+StdEnvVars
SSLCertificateFile /usr/local/apache/conf/ssl.crt/server.crt
SSLCertificateKeyFile /usr/local/apache/conf/ssl.key/server.key
SSLLog /usr/local/apache/logs/ssl_engine_log
# ---------------------
ServerName www.dhsilabs.com
ServerAdmin webmaster@den.dhsilabs.com
DocumentRoot /var/httpd/www/html
ErrorLog /var/https/www/logs/error.log
</VirtualHost>
```

После таких строк вы можете конфигурировать свой виртуальный хост как обычно. Теперь нужно перезапустить сервер httpd. При запуске Apache потребует ввести пароль. Если вы не хотите вводить его при каждом запуске системы — перейдите в каталог, где находится файл ssl.key и выполните следующие команды:

```
cp server.key server.key.org
openssl rsa -in server.key.org -out server.key
chmod 400 server.key
```

Почти все готово! Теперь сервер не должен запрашивать пароль и будет работать в нормальном режиме. При обращении https://host.domain браузер запросит вас на предмет использования сертификата.

Чтобы Apache по умолчанию стартовал с поддержкой SSL, исправьте в файле bin/apachectl условие start на **startold**, а **startssl** на просто start. Затем, находясь в каталоге .usr/local/bin, установте ссылку openssl:

```
ln -s /usr/local/ssl/bin/openssl openssl
```

12.5.3. Генерирование сертификатов

Сертификат гарантирует безопасное подключение к Web-серверам и (или) удостоверяет личность владельца. Идентификация обеспечивается путем применения личного ключа, известного только пользователю данной системы. Когда пользователь посещает защищенный узел для передачи секретной информации (например, номеров кредитных карточек) по протоколу https, узел автоматически пошлет ему сертификат.

Итак, давайте приступим к генерированию сертификатов. Для этого сначала выполните команду:

```
openssl genrsa -des3 -out server.key 1024
```

Она создаст файл server.key. После этого вы должны подать запрос в службу верификации:

```
openssl req -new -key server.key -out server.csr
```

Здесь вам придется ответить на вопросы. Если вы ошибетесь — ничего страшного, все можно будет повторить заново. В том случае, если запрос сгенерирован правильно, вы должны получить такую надпись:

```
You now have to send this Certificate Signing Request (CSR)
to a Certifying Authority (CA) for signing
```

Отвечая на вопросы, будьте очень внимательны — ваши ответы увидит весь мир.

По всем правилам, вы сейчас должны подписать сертификат у доверенного лица, но за неимением желания платить за это деньги, подпишите сами себя:

```
openssl genrsa -des3 -out ca.key 1024
openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

В данном случае у вас получится «самоподписанный» сертификат. Если вы хотите получить полноценный сертификат, то вам придется заплатить за подпись деньги. Для этого добро пожаловать на сайт **www.thawte.com.** В России представителем этого сайта является **solutions.rbc.ru**. Компания ThawTe занимается генерированием полноценных сертификатов. Получить сертификат от ThawTe можно, естественно, за деньги. На сайте же **solutions.rbc.ru** просто перепродают услуги компании ThawTe.

Возвращаясь к генерированию сертификата, скопируйте sign.sh из пакета **mod_ssl** в каталог с ключами и подпишите себя:

```
./sign.sh server.csr
```

Если на экране появится надпись:

```
Now you have two files: server.key and server.crt.
These now can be used as following
```

то это означает, что все собрано правильно. Затем скопируйте новые файлы server.key и server.crt на место старых. Выполните команду **make** в каталоге с **.crt**-файлом.

В заключении вы получите полностью работающий Apache, защищенный SSL. Для сбора полной информации о работе SSL введите:

```
openssl s_client -connect localhost:443 -state -debug
```

72.6. *Пример файла httpd.conf*

В этом разделе приведен пример стандартной конфигурации сервера Apache (см. листинг 12.13). К каждому блоку листинга сопутствуют комментарии на русском языке, которые помогут вам разобраться с различными опциями сервера.

Листинг 12.13. Пример файла httpd.conf

```
##
## httpd.conf — файл конфигурации сервера HTTP Apache
##

# -------------------------------------------------------------------

tt Установите имя сервера
ServerName www.dhsilabs.com
ResourceConfig /dev/null
AccessConfig /dev/null

# Поддержка динамических разделяемых объектов
#  (Dynamic Shared Object — DSO)
# Для более подробной информации о DSO прочтите файл README.DSO,
# входящий в дистрибутив Apache.
# Модуль расширяет возможности сервера Apache,
# добавляет в его состав новые функции.
# Подключить модуль можно так:
# LoadModule foo_module libexec/mod_foo.so
# Вы можете найти документацию по модулями в файле
# "/var/www/manual/mod"

tLoadModule mmap_static_module modules/mod_mmap_static.so
LoadModule env_module          modules/mod_env.so

### The first module activates buffered logs.
# Первый модуль обеспечивает протоколирование.
# Он запишет информацию в протокол, когда буфер объемом 4K
# переполнится. Используется файл журнала access_log
#LoadModule config_buffered_log_module modules/mod_log_config_buffered.so
LoadModule config_log_module  modules/mod_log_config.so

LoadModule agent_log_module    modules/mod_log_agent.so
LoadModule referer_log_module modules/mod_log_referer.so
#LoadModule mime_magic_module  modules/mod_mime_magic.so
LoadModule mime_module         modules/mod_mime.so
LoadModule negotiation_module modules/mod_negotiation.so
LoadModule status_module       modules/mod_status.so
LoadModule info_module         modules/mod_info.so

# Вы должны выбрать директиву mod_include или mod_include_xssi,
# но не обе одновременно! Директива mod_include более безопасна,
# но xxsi содержит больше функций.
LoadModule includes_module    modules/mod_include.so
#LoadModule includes_module    modules/mod_include_xssi.so

LoadModule autoindex_module   modules/mod_autoindex.so
LoadModule dir_module          modules/mod_dir.so
LoadModule cgi_module          modules/mod_cgi.so
```

```
LoadModule asis_module          modules/mod_asis.so
LoadModule imap_module          modules/mod_imap.so
LoadModule action_module        modules/mod_actions.so
#LoadModule speling_module      modules/mod_speling.so
LoadModule userdir_module       modules/mod_userdir.so
LoadModule proxy_module         modules/libproxy.so
LoadModule alias_module         modules/mod_alias.so

# Модуль mod_jserv должен быть объявлен до mod_rewrite.
<IfDefine HAVE_JSERV>
LoadModule jserv_module         modules/mod_jserv.so
</IfDefine>

LoadModule rewrite_module       modules/mod_rewrite.so
LoadModule access_module        modules/mod_access.so
LoadModule auth_module          modules/mod_auth.so
LoadModule anon_auth_module     modules/mod_auth_anon.so
#LoadModule dbm_auth_module     modules/mod_auth_dbm.so
#LoadModule db_auth_module      modules/mod_auth_db.so
LoadModule digest_module        modules/mod_digest.so
#LoadModule  cern_meta_module   modules/mod_cern_meta.so
LoadModule expires_module       modules/mod_expires.so
LoadModule headers_module       modules/mod_headers.so
LoadModule usertrack_module     modules/mod_usertrack.so
#LoadModule example_module      modules/mod_example.so
#LoadModule unique_id_module    modules/mod_unique_id.so
LoadModule setenvif_module      modules/mod_setenvif.so

# В полном списке модулей должны быть перечислены все доступные
# модули (статических или общедоступных), чтобы достичь
# правильного порядка выполнения.
ClearModuleList
#AddModule mod_mmap_static.c
# AddModule mod_php.c
# AddModule mod_php3.c
#AddModule mod_php4.c
#AddModule mod_perl.c
# LoadModule php_module modules/mod_php.so
# LoadModule php3_module modules/mod_php4.so
tLoadModule php4_module modules/mod_php4.so
AddModule mod_env.c
AddModule mod_log_config.c
#AddModule mod_log_config_buffered.c
AddModule mod_log_agent.c
AddModule mod_log_referer.c
#AddModule mod_mime_magic.c
AddModule mod_mime.c
AddModule mod_negotiation.c
AddModule mod_status.c
AddModule mod_info.c
AddModule mod_include.c
```

```
#AddModule mod_include_xssi.c
AddModule mod_autoindex.c
AddModule mod_dir.c
AddModule mod_cgi.c
AddModule mod_asis.c
AddModule mod_imap.c
AddModule mod_actions.c
#AddModule mod_speling.c
AddModule mod_userdir.c
AddModule mod_proxy.c
AddModule mod_alias.c

t Модуль mod_jserv должен быть объявлен до mod_rewrite.
<IfDefine HAVE_JSERV>
AddModule mod_jserv.c
</IfDefine>

AddModule mod_rewrite.c
AddModule mod_access.c
AddModule mod_auth.c
AddModule mod_auth_anon.c
#AddModuleraod_auth_dbm.c
tAddModule mod_auth_db.c
AddModule mod_digest.c
tAddModule mod_cern_meta.c
AddModule mod_expires.c
AddModule mod_headers.c
AddModule mod_usertrack.c
#AddModule mod_example.c
#AddModule mod_unique_id.c
AddModule mod_so.c
AddModule mod_setenvif.c


#----------------Name Space and Server Settings ----------------
t Настройки пространства имен и сервера
t В этом разделе вы определяете, какие имена будут видеть
t пользователи вашего HTTP-сервера. Этот файл также определяет
# настройки сервера, которые раньше содержались в отдельном файле
t srm.conf. Теперь этот файл входит в состав httpd.conf.

t Директива DocumentRoot определяет местонахождение
t корневого каталога документов вашего сервера.

DocumentRoot  /var/www/html

t Директива UserDir задает названия подкаталога в домашнем
t каталоге пользователя, из  которого берутся документы
t в том случае, если вы активизируете возможность  использования
t пользовательских каталогов.
UserDir public_html
```

```
I Директива Directorylndex позволяет задать название документа,
I который будет возвращен по запросу, который не содержит имя документа.
Directorylndex index.html index.php index.htm index.shtml index.cgi
Default.htm default.htm index.php3

# Директива FancyIndexing определяет оформление каталога -
# стандартное или индексируемое.

Fancylndexing on

# Директивы Addlcon* указывают серверу, какие пиктограммы
# использовать для показа различных типов файлов

AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddlconByType (IMG,/icons/image2.gif) image/*
AddlconByType (SND,/icons/sound2.gif) audio/*
AddlconByType (VID,/icons/movie.gif) video/*

Addlcon /icons/binary.gif .bin .exe
Addlcon /icons/binhex.gif .hqx
Addlcon /icons/tar.gif .tar
Addlcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
Addlcon /icons/compressed.gif .Z . z .tgz .gz .zip
Addlcon /icons/a.gif .ps .ai .eps
Addlcon /icons/layout.gif .html .shtml .htm .pdf
Addlcon /icons/text.gif .txt
Addlcon /icons/c.gif .c
Addlcon /icons/p.gif .pl .py
Addlcon /icons/f.gif .for
Addlcon /icons/dvi.gif .dvi
Addlcon /icons/uuencoded.gif .uu
Addlcon /icons/script.gif .conf . sh .shar .csh .ksh .tcl
Addlcon /icons/tex.gif .tex
Addlcon /icons/bomb.gif core

Addlcon /icons/back.gif ..
Addlcon /icons/hand.right.gif README
Addlcon /icons/folder.gif ^^DIRECTORY^^
Addlcon /icons/blank.gif ^^BLANKICON^^

I Директива DefaultIcon определяет пиктограмму по умолчанию.

DefaultIcon /icons/unknown.gif

# Директива AddDescription задает описание файла
# Формат: AddDescription "описание" filename

# Директива ReadmeName определяет имя файла README по умолчанию
# Формат: ReadmeName name
```

```
ReadmeName README
HeaderName HEADER

# Директива IndexIgnore определяет набор файлов, которые
# будут проигнорированы при индексировании
# Формат: Indexlgnore name1 name2...

Indexlgnore . ??* *~ *# HEADER* README* RCS

# Директива AccessFileName определяет имя файла, содержащего
# директивы управления доступом

AccessFileName .htaccess

# Директива TypesConfig задает местонахождение файла mime.types

TypesConfig /etc/httpd/conf/apache-mime.types

# С помощью директивы DefaultType можно указать MIME-тип по умол-
чанию, # для документов, тип которых сервер определить не может

DefaultType text/plain

# Директива AddEncoding разрешает вашему браузеру распаковывать
I информацию "на лету"

AddEncoding x-compress Z
AddEncoding x-gzip gz

# AddLanguage разрешает определять язык документа

AddLanguage en .en
AddLanguage fr .fr
AddLanguage de .de
AddLanguage da .da
AddLanguage el .el
AddLanguage it .it

f Директива LanguagePriority определяет приоритет языков

LanguagePriority en fr de

# Директива Redirect позволяет перенаправить клиента на другой URL
# Вы можете перенаправить клиента на другой url или же url,
# который находится в вашем пространстве имен, то есть на любой
# документ, который находится в одном из подкаталогов каталога
# DocumentRoot. Вы не можете, например, перенаправить клиента
# к каталогу /etc, потому что он не находится в вашем
# пространстве имен.
# URL - это идентификатор ресурса, поэтому вы должны его
# указывать в виде протокол://адрес.домен, например,
```

```
# http://www.linux.ru. Если вы укажите просто каталог, например,
I /images, этот каталог должен быть подкаталогом каталога
tt DocumentRoot, а не корневого каталога вашей основной файловой
# системы. Формат: Redirect несуществующий_url url

# С помощью директивы Alias можно предоставить доступ не только
I к файлам, находящимся в каталоге, указанном директивой
# DocumentRoot, и его подкаталогах, но и в других каталогах.
t Формат:
# Alias несуществующее_имя нормальное_имя

Alias /icons/ /var/www/icons/

# ScriptAlias определяет расположение каталога сценариев CGI
# Формат: ScriptAlias подставное_имя настоящее_имя

ScriptAlias /cgi-bin/ /var/www/cgi-bin/
ScriptAlias /protected-cgi-bin/ /var/www/protected-cgi-bin/


t С помощью директивы AddType можно добавить новый тип MIME,
# который не указан в файле apache-mime.types.
# Формат: AddType type/subtype ext1

# Обычно для модуля PHP3 (он не является частью Apache)
i директива AddType используется так:
AddType application/x-httpd-php4 .php3 .phtml .php .php4
tt AddType application/x-httpd-php3-source .phps
tt Для PHP/FI (PHP2):
tt AddType application/x-httpd-php .phtml
# ScriptAlias /_php/ /usr/bin/php
# Action application/x-httpd-php /usr/bin/php
# Action application/x-httpd-php3 /usr/bin/php
I Action application/x-httpd-php4 .

# Директива AddHandler позволяет сопоставить определенному
# типу MIME какой-нибудь обработчик.
# Формат: AddHandler action-name ext1

# Для использования сценариев CGI :
AddHandler cgi-script .cgi

# Для использования генерируемых сервером файлов HTML
AddType text/html .shtml
AddHandler server-parsed .shtml

# Раскомментируйте нижерасположенную строку, чтобы включить
# функцию Apache "отправь-как-есть" (send-as-is)
#AddHandler send-as-is asis
```

```
# Если вы хотите использовать карты изображений:
AddHandler imap-file map

# Для включения карт типов, используйте:
#AddHandler type-map var
# С помощью директивы Action можно определить какое-нибудь
# действие для обработчика. Например, вы можете запустить
# какую-нибудь программу для обработки файла данного типа.
I Формат: Action media/type /cgi-script/location
# Формат: Action handler-name /cgi-script/location

# Директива MetaDir определяет имя каталога, в котором сервер
# Apache может найти информационные файлы meta. Эти файлы содержат
# дополнительные заголовки HTTP, которые будут добавлены к
# документу перед его передачей клиенту.

#MetaDir .web

# Директива MetaSuffix определяет имя суффикса файла, который
# содержит meta-тэги.

#MetaSuffix .meta

# Здесь можно определить сообщения об ошибках.
# Это можно сделать тремя методами:
# 1) обыкновенный текст
# ErrorDocument 500 "The server made a boo boo.
# n.b.  the (") marks it as text, it does not get output
#
# 2) локальное перенаправление
# ErrorDocument 404 /missing.html
# to redirect to local url /missing.html
# ErrorDocument 404 /cgi-bin/missing_handler.pl
# n.b. can redirect to a script or a document using server-side-includes.
#
# 3) внешнее перенаправление
# ErrorDocument 402 http://some.other_server.com/subscription_info.html
f
<Location /manual>
ErrorDocument 404 "The document you requested has not been
installed on your system. Please install the apache-manual
package.
</Location>

f Модуль mod_mime_magic позволяет серверу использовать различные
# подсказки из файла для определения его типа.
# MimeMagicFile /etc/httpd/conf/magic

# Следующие директивы необходимы для браузеров Netscape 2.x и
# Internet Explorer 4.0b2
```

```
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-
response-1.0

# Следующие директивы отключают ответы HTTP/1.1 для браузеров,
# которые не поддерживают протокол HTTP/1.1

BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0


#----------------------Настройки доступа ----------------------

# В этом разделе определяются настройки сервера, которые управляют
# доступом к серверу. Раньше эти настройки находились в файле
I access.conf.

# Каждый каталог, к которому Apache может получить доступ,
# может быть сконфигурирован определенным образом. Можно
# запретить выполнение некоторых операций, доступ определенных
# пользователей или узлов сети.
# Установки доступа распространяются на весь каталог и на все его
# подкаталоги.

# Прежде всего, конфигурируем корневой каталог для установки
# полномочий доступа.

<Directory />
Options Indexes Includes FollowSymLinks
AllowOverride None
</Directory>


<Directory /home>

# Здесь должны быть определены директивы "Includes", "FollowSymLinks",
i "ExecCGI",  "MultiViews" или любая комбинация "Indexes"

Options Indexes Includes FollowSymLinks

AllowOverride All

# Разрешает доступ всем
order allow,deny
allow from all

</Directory>

# Каталоги /var/www/cgi-bin и /var/www/protected-cgi-bin должны быть
I определены с помощью директивы ScriptAliased
```

```
<Directory /var/www/cgi-bin>
AllowOverride All
Options ExecCGI
</Directory>

<Directory /var/www/protected-cgi-bin>
order deny,allow
deny from all
allow from localhost
#allow from .your_domain.com
AllowOverride None
Options ExecCGI
</Directory>

# Разрешает отчеты о состояния сервера

<Location /server-status>
SetHandler server-status

order deny,allow
deny from all
allow from localhost, 127.0.0.1
# Установите здесь имя вашего домена
# allow from .your_domain.com
</Location>

# Разрешает доступ к файлам документации для локальной машины.
Alias /doc /usr/share/doc
<Directory /usr/share/doc>
order deny,allow
deny from all
allow from localhost, 127.0.0.1
# allow from .your_domain.com
Options Indexes FollowSymLinks
</Directory>

#--------------------Конфигурация сервера----------------------------

# Тип сервера: inetd или standalone.

ServerType standalone

# Если вы используете тип inetd, перейдите к директиве "ServerAdmin"

# Директива Port - только для standalone-сервера.
# Если вы хотите запустить сервер Apache с использованием этого
# или любого другого порта, номер которого меньше 1024,
# вы должны обладать правами суперпользователя. Но
# даже если у вас нет таких прав, вы можете запустить сервер
# для работы с портом, номер которого превышает значение 1024.
```

```
t Обычно используется
# номер 8080 или 8000.

Port 80

t Сервер Apache ведет журнал доступа других компьютеров.
# Если вы включите данную опцию, то в журнал будет записано
t доменное имя компьютера-клиента. Если эта опция выключена,
# то в журнал будет записан IP-адрес клиента. Включение данной
# опции замедляет работу сервера, так как ему требуется
i дополнительное время на ожидание ответа от сервера DNS.

HostnameLookups off

# Директивы User и Group определяют идентификаторы пользователя
# и группы.
# Данные идентификаторы присваиваются серверу, если он запущен в
# автономном режиме. Можно использовать как имена пользователей,
# так и их числовые эквиваленты - UID. По умолчанию используется
# имя пользователя nobody или apache. Из соображений безопасности
# не рекомендуется изменять это значение и присваивать имя
# реального пользователя.

User apache
Group apache

# Директива ServerAdmin задает электронный адрес вебмастера вашего
# Web-узла. В случае возникновения ошибок именно по этому адресу
# будет отправлено сообщение.

ServerAdmin root@localhost

# В директиве ServerRoot указывается местонахождение файлов
# конфигурации сервера Apache.
# По умолчанию используется каталог /etc/httpd.

ServerRoot /etc/httpd

# Данная директива используется для компьютеров, которые имеют
# несколько IP-адресов. Обычно данная директива используется
# для конфигурирования виртуальных хостов.
# BindAddress *

# Прослушивать порт 80
Listen 80

# Директивы ErrorLog и TransferLog определяют расположение
# журналов сервера Apache. Обычно используется каталог
# /etc/httpd/logs, который является ссылкой на каталог
# /var/log/httpd или на любой другой.
```

```
ErrorLog logs/error_log

# LogLevel: устанавливает уровень протоколирования.
# Протоколируются предупреждающие сообщения сервера (warn)
# и ошибки. Если вы хотите протоколировать только ошибки,
# установите error

LogLevel warn

# Определяет формат файлов протокола, то есть информация,
# которая будет протоколироваться. Обычно их не нужно изменять.

LogFormat "%h %1 %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

# Можно определить тип протокола
# Если вы хотите протоколировать общую информацию
# CustomLog logs/access_log common
# Если вы хотите протоколировать referer
# CustomLog logs/referer_log referer
# Если вы хотите протоколировать название пользовательских
# агентов (браузеров)
# CustomLog logs/agent_log agent
# По умолчанию используется комбинированный тип протоколирования,
# то есть будет протоколироваться вся информация

CustomLog logs/access_log combined

# С помощью директивы PidFile указывается имя файла,
# в котором исходный процесс сервера будет регистрироваться.

PidFile /var/run/httpd.pid

# ScoreBoardFile: Этот файл используется для сохранения внутренней
# информации процесса сервера.

ScoreBoardFile /etc/httpd/httpd.scoreboard

# Директива LockFile определяет файл блокировки, который
# используется сервером. Сервер должен быть скомпилирован с опцией
# USE_FCNTL_SERIALIZED_ACCEPT или
# USE_FLOCK_SERIALIZED_ACCEPT. Файл блокировки должен быть
f сохранен НА ЛОКАЛЬНОМ ДИСКЕ.
#
LockFile /etc/ittpd/httpd.lock

I Директива ServerName позволяет вам установить имя узла вашего
# сервера, которое будет использоваться клиентами,
```

```
# то есть установить "www" вместо реального имени сервера.
# Имя, которое вы укажете, должно быть указано в сервере DNS
# вашего домена.

ServerName new.host.name

# Директива UseCanonicalName появилась в версии Apache 1.3.
# Она разрешает использовать каноническое имя для сервера узла.

UseCanonicalName on

# Данная директива позволяет прокси-серверу, например SQUID,
# не кэшировать документы, которые не генерируются автоматически,
# то есть в процессе выполнения различных сценариев.

CacheNegotiatedDocs

# Директива Timeout задает промежуток времени в секундах,
# в течение которого сервер продолжает попытки возобновления
# приостановленной передачи данных. Значение директивы Timeout
I распространяется не только на передачу, но и на прием данных.

Timeout 300

# Директива KeepAlive разрешает постоянные соединения,
I то есть такие соединения, в которых производится более
# одного запроса за один раз.

KeepAlive off

# MaxKeepAliveRequests: Максимальное количество запросов,
# разрешенное в течение постоянного соединения. Установите О
tt для снятия ограничения. Для повышения производительности
# рекомендуется установить это число сравнительно большим.

MaxKeepAliveRequests 100

# Директива KeepAliveTimeout определяет таймаут для постоянного
# соединения.

KeepAliveTimeout 15

# Минимальное и максимальное число серверов в пуле
MinSpareServers 8
MaxSpareServers 20

t Количество серверов для запуска

StartServers 10
```

```
# Ограничивает общее количество клиентов. Когда это число будет
# превышено,новые клиенты получат отказ, поэтому это число
# не должно быть слишком маленьким.

MaxClients 150

# После обработки определенного количества запросов, указанного
# в директиве MaxRequestsPerChild, копия сервера завершается,
# а вместо нее запускается новая.

MaxRequestsPerChild 500

# Директивы конфигурации прокси-сервера

# ProxyRequests On

# Для включения кэширования, раскомментируйте строки ниже:

#  CacheRoot  /var/cache/httpd
# CacheSize 5
# CacheGcInterval 4
# CacheMaxExpire 24
# CacheLastModifiedFactor 0.1
# CacheDefaultExpire 1
# NoCache a_domain.com another_domain.edu joes.garage_sale.com

####################################
# Настройки производительности SGI    #
### ################################
#
#
# Для использования этой функции, раскомментируйте модуль
# mod_mmap_static в разделе описания модулей.
<IfModule mod_mmap_static.c>
QSC on
</IfModule>

# Если вы хотите использовать буферизированное протоколирование,
# раскомментируйте модуль mod_log_config_buffered в разделе
# описания модулей.

# Для использования карты памяти, раскомментируйте эту строку:
# mmapfile /var/www/html/file_to_map_in_memory
#
# Если вы хотите настроить процессы потомки, пожалуйста,
# прочитайте документацию на вашем сервере
# http://localhost/manual/misc/perf-mja.html.
# Эта страница объясняет как привязать определенный IP-адрес
# или порт к flpVroMy процессору.
# SingleListen On
```

```
####################################
# Виртуальные серверы #
####################################
#
# Поддержка модуля perl
t Замечание: не удаляйте расположенные далее строки, иначе это может
# разрушить вашу конфигурацию.
<IfDefine PERLPROXIED>
 ProxyPass     '/perl/ http://127.0.0.1:8200/
 ProxyPassReverse   /perl/  http://127.0.0.1:8200/
</IfDefine>

# Файл, в котором находятся директивы конфигурирования
# виртуальных узлов.
Include conf/vhosts/Vhosts.conf

#
t Для поддержки динамических виртуальных хостов и виртуальных
# домашних каталогов, раскомментируйте следующие строки:
# LoadModule vhost_alias_module   modules/mod_vhost_alias. so
# AddModule mod_vhost_alias.c
# Include conf/vhosts/DynamicVhosts.conf
t Include conf/vhosts/VirtualHomePages.conf

# Директивы конфигурирования PHP
Include conf/addon-modules/php.conf
```

12.7. Перекодирование русскоязычных документов «на лету»

С тех пор, когда в русском языке появилось слово кодировка, появилась и проблема перекодировки. Стандартной кодировкой русского языка для большинства Unix-серверов является КОИ-8. Аббревиатура КОИ расшифровывается как «Код для Обмена Информацией». Все бы было хорошо, если бы существовала одна-единственная кодировка KOI8-r. Но, как всегда, найдутся добрые люди, которые захотят помочь нам кодировать наш русский язык. И таких нашлось немало. Например, всем известная Microsoft сначала использовала кодировку CP-866 (еще известную как альтернативная кодировка — ALT) в своей операционной системе DOS. ОС DOS тогда была установлена на большинстве домашних и офисных компьютеров, поэтому особой разницы, что использовать: KOI8-r или CP-866, для пользователей не было по одной простой причине: у меня отображается так же, как и у соседа. Потом та же Microsoft разработала кодировку Windows-1251 (ANSI) для своей новой операционной системы Windows. Стараясь нам «помочь», она создала проблемы с перекодировкой уже на локальном уровне: файлы, созданные в ОС DOS, без предварительного перекодирования нельзя было прочитать в Windows, и наоборот. Заметьте, об Интернет и Apache я еще не сказал ни слова.

Кроме Microsoft, «облегчили» нам жизнь также компании Apple и Sun, разработав соответственно кодировки Apple и ISO8859-5. Компания IBM также не отстала от них, разработав собственную кодировку русского языка. В общем, развитие таблиц перекодирования русского языка происходило по принципу: а мы новый дом построим...

А теперь представьте: все эти кодировки смешались в Интернет. Я работаю под Unix и использую KOI8, а вы под Windows и мы не можем прочитать файлы друг друга из-за проблем с кодировкой. Это довольно неудачный пример, так как средств для перекодирования из KOI8 в Windows-1251 разработано довольно много. А вот с кодировкой Apple могут возникнуть проблемы. Мне встречались браузеры, некорректно отображающие русские символы в этой кодировке.

Возникает задача: настроить автоматическое перекодирование документов из одной кодировки в другую. Для начала необходимо настроить хотя бы перекодирование «на лету» из KOI8 в Windows-1251, так как большинство клиентов в Сети используют именно эту кодировку (от Windows, как от смерти, не уйдешь).

Обыкновенный англоязычный Apache, входящий в состав большинства дистрибутивов, не сможет вам в этом помочь. Для корректных операций по перекодированию нужно загрузить и установить сервер **Russian Apache.** Или же просто установить модуль **Apache-RUS**. Скачать данный модуль (как и полностью Russian Apache) можно по адресу: **ftp://apache.lexa.ru/pub/apache-ras/.** При этом старшая часть версии соответствует оригинальному серверу Apache, младшая — версии модуля Apache-Rus.

Рассматривать процесс установки и настройки я буду на примере не очень новой версии сервера — 1.3.3/PL27.3, это не принципиально. Загрузив файл, распакуйте его:

```
tar xvzf apache_1.3.3rusPL27.3.tar.gz
```

После этого перейдите в каталог `apache_1.3.3rusPL27.3` и запустите сценарий configure:

```
# cd apache_1.3.3rusPL27.3
t ./configure
```

Можно также указать некоторые параметры сценария, например, параметр **-prefix=path** следует использовать при необходимости установить сервер в другой каталог. Далее введите ставшие уже привычными вам команды:

```
# make
# make install
```

После этого нужно настроить сервер, то есть отредактировать его файлы конфигурации. Настройка Russian Apache не отличается от настройки обыкновенного сервера Apache, за исключением настройки модуля перекодирования. Ниже будет рассмотрена настройка именно перекодирования «на лету», так как настройка самого сервера описана в предыдущих пунктах этой главы.

Директивы перекодирования можно разделить на три группы. Первые указывают в какой кодировке хранятся файлы на диске, вторые — определяют названия кодировок, их псевдонимов, таблиц символов. Третья группа

определяет порядок перекодирования документов. Приведу небольшой пример использования первой группы кодировок:

```
CharsetSourceEnc koi8-r
CharsetByExtension windows-1251 .txt
```

Эти директивы находятся в файле `httpd.conf`. Первая указывает на то, что все файлы на диске, кроме файлов с расширением .txt, хранятся в кодировке koi8-r. Для последних указывается кодировка — windows-1251.

Данные директивы можно включать в блок **Location** или в файлы `.htaccess`.

Ко второй группе относятся **директивы CharsetDecl, CharsetAlias, CharsetRecodeTable и CharsetWideRecodeTable.** Они находятся в блоке **<IfModule mod_charset.c> — </IfModule>** и не нуждаются в изменении. Необходимо только отметить (это нужно для понимания дальнейшего материала), что директива **CharsetDecl** используется для объявления кодировки, а **CharsetAlias** — для объявления псевдонима кодировки, например:

```
CharsetDecl windows-1251 ru
CharsetAlias windows-1251 win x-cp1251 cp1251 cp-1251
```

Название языка (ru) должно быть определено в файле `conf/srm.conf` в директивах **AddLanguage и LanguagePriority**.

С помощью третьей, самой многочисленной, группы вы можете настроить сервер для автоматической перекодировки символов на основании информации о клиенте. Например, определив, что клиент работает в операционной системе Windows и кодировкой браузера по умолчанию является Windows-1251, сервер самостоятельно перекодирует файлы в нужную кодировку. Если сервер сделает выбор неправильно, пользователь всегда сможет сам изменить кодировку вручную. Существует три способа выбора кодировки:

По каталогу, например:

http://www.server.ru/koi/file.html
http://www.server.ru/win/file.html

По имени сервера, например:

http://koi.www.server.ru/file.html
http://win.www.server.ru/file.html

По порту, например:

http://www.server.ru:8000/file.html
http://www.server.ru:8001/file.html

Теперь рассмотрим каждый из этих способов.

Для перекодирования по каталогу (точнее, по его префиксу) нужно добавить в блоке **VirtualHost** псевдоним, например:

```
Alias /koi /www/docs
```

Или же просто создать в нужном каталоге ссылку на самого себя:

```
# cd /www/docs
# ln -s . koi
```

Несмотря на свою простоту, этот способ имеет множество недостатков. Если у вас небольшой сервер, вы можете использовать перекодировку по

каталогу. В другом случае, лучше используйте перекодировку по имени сервера или по порту.

При использовании перекодировки по имени сервера следует обратить внимание на то, чтобы указанный вами сервер был прописан на сервере DNS. После регистрации поддомена внесите следующие строки в ваш файл httpd.conf:

```
# Естественно, укажите здесь свой IP-адрес
<VirtualHost 111.111.111.1>
ServerName www.server.ru
ServerAlias *.www.server.ru
# далее следует обычная конфигурация
...
</VirtualHost
```

В качестве имени поддомена нужно использовать один из псевдонимов кодировки, указанный с помощью директивы CharsetAlias, например, koi или win.

Если же сервер DNS администрируете не вы, а кто-то другой и возможности внести изменения в записи DNS у вас нет, то используйте перекодировку по порту. Для этого удалите (закомментируйте) директиву **Port** в файле конфигурации httpd. conf и вместо нее добавьте следующие директивы:

```
Listen 80
Listen 8000
Listen 8001
Listen 8002
Listen 8003
CharsetByPort koi8-r 8000
CharsetByPort windows-1251 8001
CharsetByPort ibm866 8002
CharsetByPort iso-8859-5 8003
```

Номера нортов при этом не очень важны. Думаю, это пример настолько прозрачен, что не нуждается в особых комментариях. Правда, есть одно но: если сеть клиента защищена брандмауэром (см. гл. 14), не позволяющем обращения к выбранному вами порту, клиент не сможет установить соединение с вашим сервером.

Схема (порядок) выбора кодировки определяется директивой CharsetSelectionOrder:

CharsetSelectionOrder Dirprefix Useragent Portnumber Hostname UriHostname — для выбора по каталогу.

CharsetSelectionOrder Hostname UriHostname Useragent Portnumber Dirprefix — для выбора по имени домена.

CharsetSelectionOrder Portnumber Useragent Hostname UriHostname Dirprefix — для выбора по порту.

Для начальной настройки перекодирования ваших знаний уже достаточно, а дополнительную информацию вы найдете в документации по серверу Russian Apache.

13

Почтовый сервер

Для начала небольшое отступление. Вкратце напомню о протоколах SMTP и POP, которые будем конфигурировать.

SMTP (Simple Mail Transfer Protocol) — сервис в сетях TCP/IP для передачи сообщение (т.е. почты). Обычно для SMTP используется порт 25 (см. файл /etc/services).

POP (Post Office Protocol) — используется для получения почты с сервера. Порт по умолчанию — 110 (для протокола POP3).

Основными MTA (Mail Transfer Agent) — агентами пересылки почты на сегодняшний день являются **sendmail**, **postfix** и **qmail**. Кроме основной функции — отправка сообщения электронной почты, каждый их них имеет собственные эксклюзивные функции. **Sendmail** является одной из самых ранних программ MTA — своеобразным стандартом, который использовался еще в самых ранних версиях Unix. Этот агент прошел долгий путь со дня своего создания. Раньше (до 1998 года) в конференциях вообще рекомендовалось вместо использования **sendmail** перейти на **qmail,** обладающий лучшей защищенностью. Действительно, в то время **sendmail** имел много брешей с точки зрения безопасности, а тем, кто решался все-таки использовать его, рекомендовалось регулярно обновлять **sendmail.** Сейчас **sendmail** обладает должным уровнем безопасности и достаточно легко настраивается, в чем вы убедитесь в этой главе. Возможно, **postfix** обладает и более простой настройкой, но у меня почему-то не сложилась работа с этой программой. О **qmail** могу сказать только то, что редко встретишь программу с более сложной настройкой. Для заинтересовавшихся я разместил на прилагаемом компакт-диске руководство по настройке **qmail.**

Сейчас я попробую объяснить, как настроить небольшой почтовый сервер, использующий POP3 и SMTP. Для начала установите необходимое программное обеспечение. При этом вам понадобятся пакеты **sendmail и imap**.

Да простят меня почитатели **PostFix, но** я действительно предпочитаю **sendmail.** Возможно, **postfix** проще конфигурируется, но это дело вкуса.

13.1. Настройка sendmail

Если вы используете дистрибутив, совместимый с RedHat, то вам потребуется установить соответствующие пакеты. Я использую **sendmail-8.11.0 и imap-4.7c2.** Последнюю версию **sendmail** можно выкачать из Интернет по адресу **http://www.sendmail.org.**

Прежде, чем приступить к настройке **sendmail,** вам необходимо правильно настроить DNS. Настройка сервера DNS подробно обсуждалась в гл. 10. Если вы настраиваете только почтовый сервер, вам необязательно настраивать сервер DNS на этом же компьютере. Достаточно будет указать DNS-сервера вашей сети в файле `/etc/resolv.conf`, чтобы система разрешения имен корректно работала. Впрочем, **sendmail** можно настроить для работы без использования DNS, но этот вариант я рассматривать не буду.

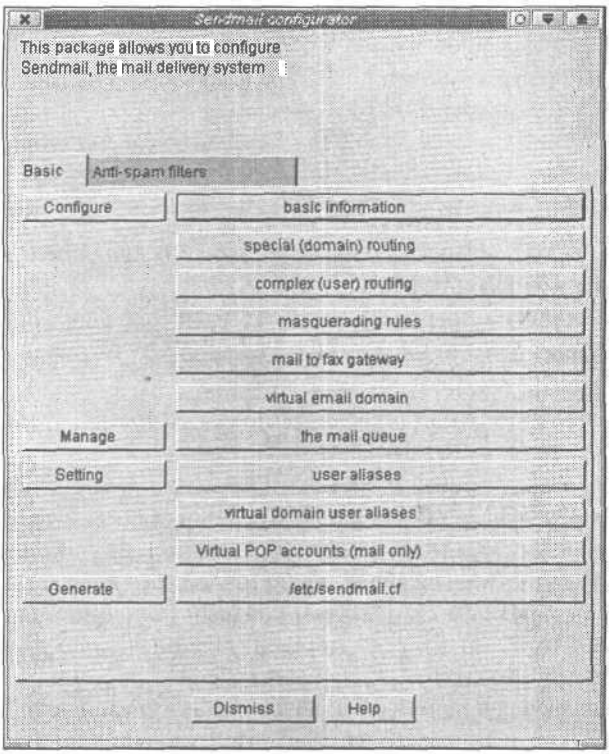


Рис. 13.1. *Конфигуратор netconf*

Для того, чтобы приступить к базовой настройке **sendmail,** запустите утилиту **netconf** (см. рис. 13.1). Она работает как и из-под X-Window, так и из-под консоли. Утилита **netconf** есть в RedHat Linux, Mandrake, ASPLinux и в других дистрибутивах, ее точно нет в KSI Linux. Естественно, вы должны быть зарегистрированы в системе как root. Если у вас нет программы **netconf,** вы получите незабываемое удовольствие от редактирования файла `/etc/sendmail.cf` вручную. Именно в этом файле хранятся все настройки **sendmail.**

Выберите в меню **Mail delivery system,** затем **Basic sendmail configuration.** В поле **Present your system as** просто введите свое доменное имя. Затем обязательно отметьте флажок «Accept email for your_domain.com» (см. рис. 13.2).

Если вы этого не сделаете, через ваш сервер можно будет перенаправить сообщения на другой сервер. При этом через ваш сервер будет идти лишний трафик, который вам совсем ни к чему. Когда-то даже существовал такой вид атаки на отказ через электронную почту. Принцип его следующий: отправляется письмо от несуществующего пользователя not_exists@A.com другому несуществующему пользователю not_exists@B.com. Письмо отправ-

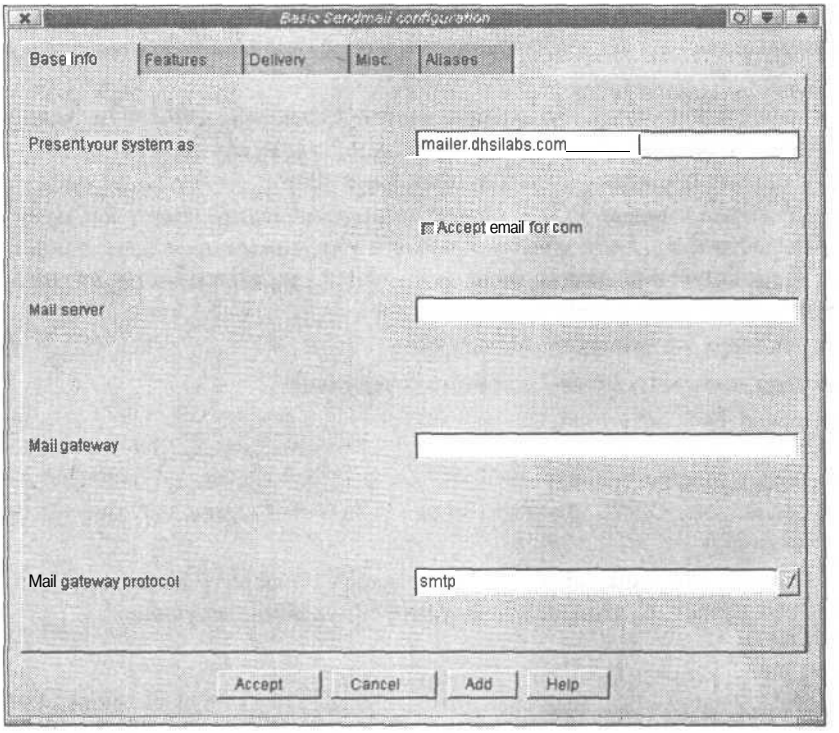


Рис. ***13.2.*** *Основная конфигурация sendmail*

ляется через компьютер hostcom, который позволяет перенаправить сообщение (не включен режим «Accept email for domain.com»). Почтовый сервер домена B.com отправляет сообщение по адресу not_exists@A.com о том, что пользователь not_exists@B.com не существует. В свою очередь, почтовик домена A.com сообщает, что пользователя not_exists@A.com также не существует и отправляет сообщение по адресу not_exists@B.com. Возникает косвенная рекурсия. А теперь представьте, что такое сообщение не одно, а, скажем, 100 и объем каждого хотя бы 1 Мб! В результате один из серверов в цепочке domainl.com — host.com — domain2.com должен «упасть».

Прошу извинения, я немного отошел от темы, продолжим конфигурировать sendmail. Протокол отправки сообщений установите **smtp** (Mail gateway protocol).

Этой информации уже вполне достаточно, чтобы ваш sendmail функционировал. Вы можете установить дополнительные опции программы **sendmail** с помощью **netconf**.

Теперь сделайте так, чтобы **sendmail** принимал почту только с разрешенных адресов. Для этого даже не нужно настраивать сам **sendmail** — нужно только подправить файлы `/etc/hosts.allow` и `/etc/hosts.deny`. В первом из них содержится список хостов, которым разрешен доступ к этой машине, а во втором — которым запрещен. Обратите внимание: несмотря на название — «разрешен» или «запрещен», ограничения, которые устанавливаются первым

файлом намного строже. Например, для запрещения доступа всем хостам, кроме компьютеров вашей сети, в файл /etc/hosts.allow добавьте строку:

```
'192.168.1.
```

Здесь имеется в виду, что сеть имеет адрес 192.168.1.0 и маску 255.255.255.0. Более подробно о формате файлов hosts.allow и hosts.deny вы можете прочитать, введя команду man **hosts.allow**.

Теперь настала очередь POP3. После установки пакета **imap** у вас практически все настроено, т.е. я хочу сказать, что уже можно проверять конфигурацию. Перезапустите демон **inetd** или **xinetd**, в зависимости от того, который вы используете, и введите следующее:

```
telnet <имя_только_созданного_почтовика> 25
```

В ответ вы должны увидеть примерно следующее:

```
Trying 192.168.1.1 ...
Connected to 192.168.1.1
Escape character is '^]'
220 de.dhsilabs.com ESMTP Sendmail 8.11.0/8.8.7 Sun, 17 Jun
2001 10:54:22 +300
```

Это означает, что **sendmail** работает, осталось проверить насколько правильно он это делает. С этой целью введите примерно следующее:

```
mail from: me@my.host.com
220 2.1.0 me@my.host.com .... Sender Ok
rcpt to: den@den.dhsilabs.com
220 2.1.5 den@den.dhsilabs.com .... Recipient Ok
```

После этого введите команду **data,** потом текст сообщения, а для окончания ввода поставьте точку в пустой строке. **Sendmail** сообщит, что сообщение отправлено (точнее помещено в очередь на отправку). Запись den@den.dhsilabs.com — имя пользователя, которому вы отправляете почту. Пользователь должен реально существовать. Запись den.dhsilabs.com — имя вашего почтовика.

Примечание.

Имена den@den.dhsilabs.com и dhsilabs.com даны в качестве примера. Вместо них вы должны указать свои значения.

Обратите внимание, что узла my.host.com и в природе нет, а программа **sendmail** сообщает, что «Sender Ok». Вот поэтому, в настройках **sendmail** лучше включить опцию Wait **for DNS.**

Теперь нужно запустить какой-нибудь почтовый клиент, например **kmail**, и получить почту. Используйте следующие настройки сети в программе **kmail: Сеть → Отправка почты** установите SMTP, порт 25, имя сервера — имя вашего почтовика, в рассматриваемом примере — это den.dhsilabs.com. Затем добавьте аккаунт для POP3:

Имя пользователя.....den
Пароль.....пароль, который используется для входа в систему.
Сервер.....den.dhsilabs.com
Порт.....НО

В результате вы должны получить то сообщение, которое ввели после **data.** При этом возможны проблемы при разрешении имени. Чтобы их избежать необходимо правильно настроить DNS или вместо имени почтового сервера использовать его IP-адрес. При добавлении нового пользователя не забудьте установить его пароль для входа в систему. Если этого не сделать, а пытаться получить почту без указания пароля, вы получите сообщение «Сбой аутентификации».

Базовая настройка программы **sendmail** выполняется очень просто с использованием конфигуратора, но иногда базовой настройки недостаточно. Для более точной настройки нужно ознакомиться с файлами конфигурации программы **sendmail.**

Примечание.

В большинстве случаев, вас будет устраивать базовая настройка до тех пор, пока вас не возьмут под свой «протекторат" спаммеры — тогда вам нужно будет прочесть гл. 23 о методах защиты от спама. Однако может сработать один из законов Мерфи — для вашей системы не будет разработан конфигуратор sendmail и тогда вам все равно придется разбираться с файлами конфигурации.

Основным файлом конфигурации **sendmail** является файл /etc /sendmail.cf. В некоторых дистрибутивах этот файл расположен в каталоге /etc/mail. Об этом файле говорят, что он длиннее, чем лимузин у Билла Гейтса и что его редактирование происходит в режиме «глаза боятся, руки делают». Если вы не верите мне, откройте этот файл, и вы убедитесь в этом. Редактировать данный файл вручную могут только профессионалы-администраторы или разработчики программы **sendmail.**

Обычно для редактирования этого файла используется макропроцессор **m4**. Сначала вы подготавливаете специальный **mc-файл**. В этом файле записаны настройки **sendmail**, но в более «читабельном» виде. При редактировании конфигурационных файлов также сказывается и их размер. Для сравнения: размер моего **mc-файла** — 2459 байт, а файла sendmail.cf — 46302 байт. Одно дело, редактировать файл размером в два килобайта, и совсем другое, если размер файла равен 46 килобайтам. В отличие от конфигурационного файла **sendmail,** вы сразу поймете, для чего предназначен тот или иной **mc-файл.** Затем, отредактировав **mc-файл,** нужно запустить макропроцессор **m4** для создания файла конфигурации **sendmail:**

```
m4 my_config.mc > /etc/sendmail.cf
```

Перед выполнением этой команды я настоятельно рекомендую сохранить где-нибудь исходный файл sendmail.cf. В случае некорректной настройки вы всегда сможете восстановить его.

Файл конфигурации по умолчанию, который используется макропроцессором **т4** для создания файла конфигурации программы **sendmail** (sendmail.cf), находится в каталоге /usr/share/sendmail-cf/cf. В более старых версиях программы **sendmail** он может быть расположен в каталоге /usr/lib/sendmail.

Как правило, этот файл называется sendmail.mc. Иногда он может называться и по-другому, например, redhat.mc, если вы используете операционную систему Red Hat или совместимую с ней.

Пример стандартного файла /usr/share/sendmail-cf/cf/redhat.mc приведен в листинге 13.1.

Листинг 13.1. Стандартный файл redhat.mc

```
divert(-1)
dnl This is the sendmail macro config file. If you make changes to this file,
dnl you need the sendmail-cf rpm installed and then have to generate a
dnl new /etc/sendmail.cf by running the following command:
dnl
dnl         m4 /etc/mail/sendmail.mc > /etc/sendmail.cf
dnl
include('../m4/cf.m4')
VERSIONID('linux setup for Red Hat Linux')dnl
OSTYPE('linux')
define('confDEF_USER_ID',"8:12")dnl
undefine('UUCP_RELAY')dnl
undefine('BITNET_RELAY')dnl
define('confAUTO_REBUILD')dnl
define('confTO_CONNECT', '1m')dnl
define('confTRY_NULL_MX_LIST',true)dnl
define('confDONT_PROBE_INTERFACES',true)dnl
define('PROCMAIL_MAILER_PATH','/usr/bin/procmail')dnl
define('ALIAS_FILE', '/etc/aliases')dnl
dnl define('STATUS_FILE', '/etc/mail/statistics')dnl
define('UUCP_MAILER_MAX', '2000000')dnl
define('confUSERDB_SPEC', '/etc/mail/userdb.db')dnl
define('confPRIVACY_FLAGS', 'authwarnings,novrfy,noexpn,restrictqrun')dnl
define('confAUTH_OPTIONS', 'A')dnl
dnl TRUST_AUTH_MECH('DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define('confAUTH_MECHANISMS', 'DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define('confTO_QUEUEWARN', '4h')dnl
dnl define('confTO_QUEUERETURN', '5d')dnl
dnl define('confQUEUE_LA', '12')dnl
dnl define('confREFUSE_LA', '18')dnl
dnl FEATURE(delay_checks)dnl
FEATURE('no_default_msa','dnl')dnl
FEATURE('smrsh','/usr/sbin/smrsh')dnl
FEATURE('mailertable', 'hash -o /etc/mail/mailertable.db')dnl
FEATURE('virtusertable', 'hash -o /etc/mail/virtusertable.db')dnl
FEATURE(redirect)dnl
FEATURE(always_add_domain)dnl
FEATURE(use_cw_file)dnl
FEATURE(use_ct_file)dnl
FEATURE(local_procmail, '', 'procmail -t -Y -a $h -d $u')dnl
FEATURE('access_db','hash -o /etc/mail/access.db')dnl
FEATURE('blacklist_recipients')dnl
EXPOSED_USER('root')dnl
dnl This changes sendmail to only listen on the loopback device 127.0.0.1
dnl and not on any other network devices. Comment this out if you want
dnl to accept email over the network.
```

```
DAEMON_OPTIONS('Port=smtp,Addr=127.0.0.1, Name=MTA')
dnl NOTE: binding both IPv4 and IPv6 daemon to the same port requires
dnl         a kernel patch
dnl DAEMON_OPTIONS('port=smtp,Addr=::1, Name=MTA-v6, Family=inet6')
dnl We strongly recommend to comment this one out if you want to protect
dnl yourself from spam. However, the laptop and users on computers that do
dnl not have 24x7 DNS do need this.
FEATURE('accept_unresolvable_domains')dnl
dnl FEATURE('relay_based_on_MX')dnl
MAILER(smtp)dnl
MAILER(procmail)dnl
Cwlocalhost.localdomain
```

С помощью директивы FEATURE можно подключить ту или иную функцию программы **sendmail**. Например, функция **mailertable** предназначена для переопределения маршрутизации для конкретных доменов. Вы можете легко расширить функциональные возможности программы **sendmail,** добавив нужные вам функции в **mc-файл**.

Предположим, что вы хотите, чтобы названия компьютеров домена были скрыты. Это легко достигается с помощью добавления функции **masquerade_envelope в ваш mc-файл**. Для этого скопируйте файл `redhat.mc` в файл hide_hosts `.mc` и добавьте в конец файла hide_hosts `.mc` строки:

```
MASQUERADE_AS(my-domain.ru)dnl
FEATURE(masquerade_envelope)dnl
```

Затем выполните команду:

```
m4 /usr/share/sendmail-cf/cf/hide_hosts.mc > /etc/sendmail.mc
```

Вот и все! Названия узлов будут скрыты. Описание прочих функций представлено в табл. 13.1.

Функции программы sendmail *Таблица 13.1*

Функция	Описание
access_db	Определяет таблицу доступа. В этой таблице указаны хосты, которым разрешена или запрещена отправка почты через ваш почтовый сервер. Эта опция эффективно используется для борьбы со спамом. Защита от спама подробно рассмотрена в одноименном разделе в гл. 23
accept_unresolvable_domains	Разрешает отправлять почту доменам, которые не могут быть распознаны
bestmx_is_local	Сообщения будут приниматься только в том случае, если запись MX-сервера DNS указывает на этот почтовый сервер
blacklist_recipients	«Черный список». Еще одна опция для борьбы со спамом. Для ее работы необходима опция access_db
dnsbl	Используется для работы с «черным списком». dnsbl — это сокращение от DNS Black List. В более ранних версиях эта опция называлась rbl (Resolve Black List)
domaintable	Используется для разрешения имен доменов
genericstable	Используется для изменения адреса отправки в сообщениях
local_procmail	Указывает, что доставлять почту нужно с помощью локальной утилиты procmail
mailertable	Переопределяет маршрутизацию для конкретных доменов
masquerade_entire_domain	Используется для маскировки (сокрытия) всего домена. Данная функция должна использоваться вместе с директивой MASQUERADE AS (или MASQUERADE_DOMAIN), например, MASQUERADE_AS(f117.ru)dnl

Продолжение табл. 13.1

Функция	Описание
masquerade_envelope	Позволяет скрыть имена хостов домена. Заменяет поле received from заголовка сообщения перед передачей сообщения другим MTA
redirect	Используется для перенаправления на другой почтовый сервер. Означает отказ от принятия почты с выдачей сообщения please try <address> (попытайтесь использовать этот адрес)
relay_based_on_MX	Разрешает перенаправление (ретрансляцию) почты только для узлов, которые указаны в записях MX-сервера DNS
relay_hosts_only	Разрешает ретрансляцию только для узлов, указанных в access_db
relay_mail_from	Разрешает ретрансляцию только, если отправитель указан в списке RELAY базы access_db
smrsh	Использование ограниченной оболочки sendmail
use_cf_file	При указании этой функции sendmail будет обращаться к файлу sendmail. cf за списком доверенных пользователей
use_cw_file	При указании этой функции sendmail будет обращаться к файлу sendmail.cw за списком локальных узлов
virtusertable	Преобразует адрес получателя в адрес локального пользователя

В файле `/etc/mail/sendmail.cw` перечислены все псевдонимы данного почтового сервера. Предположим, что имя вашего сервера mail.dhsilabs.ru. Если отправитель отправит почту по адресу den@mail.dhsilabs.ru, письмо будет без проблем доставлено пользователю den. А если кто-то отправит письмо по адресу den@dhsilabs.ru, то его доставка вызовет определенные трудности, так как не ясно какому узлу домена dhsilabs адресовано сообщение? Для решения этой проблемы в файл `sendmail.cw` нужно поместить строку:

```
dhsilabs.ru
```

Теперь, когда будет приходить почта формата `user@dhsilabs.ru`, она будет доставлена почтовому серверу mail.dhsilabs.ru.

Напомню, что перед изменением файла `sendmail.cf` желательно остановить программу **sendmail.** Это делается с помощью команды:

```
/etc/init.d/sendmail stop
```

Запустить **sendmail** заново можно с помощью команды:

```
/etc/init.d/sendmail start
```

Конечно, удобнее сначала отредактировать файл `sendmail.cf` с помощью т4, а потом выполнить команду **/etc/init.d/sendmail restart** для перезапуска программы **sendmail.**

13.2. Аутентификация в sendmail

Программы Sendmail 8.10/8.11 поддерживают **SMTP AUTH,** согласно стандарту RFC 2554. Аутентификация базируется на SASL. Она позволит вам несколько повысить безопасность вашей сети, но создаст определенные неудобства для пользователей, потому что не все почтовые клиенты ее поддерживают.

Вам потребуются библиотеки **Cyrus** SASL, исходные коды которых вы можете найти по адресу **ftp://ftp.andrew.cmu.edu/pub/cyrus-mail/.** Последней версией на момент написания этих строк являлась 1.5.14. Желательно также

выкачать последнюю версию sendmail, хорошо бы в исходных кодах. Скачать последнюю версию программы **sendmail** можно по адресу: **http:// www.sendmail.org.** Распакуйте библиотеку Cyrus SASL выполнив следующую последовательность:

```
t tar -xzf cyrus-sasl-1.5.24.tar.gz
tt cd cyrus-sasl-1.5.24/
t ./configure —prefix=/usr
# make
I make install
```

Распаковывать лучше всего, зарегистрировавшись в системе root. Сама по себе операция извлечения файлов из архива не требует таких полномочий, однако, если у вас не было прав суперпользователя, вам нужно будет изменить права доступа для файлов, которые будут распакованы в каталоги /usr/lib и /usr/include. Распаковав исходные коды, отредактируйте файл /usr/lib/sasl/Sendmail.conf. Если он не существует, создайте его. В конец этого файла необходимо добавить строку:

```
pwcheck_method: sasldb
```

Это укажет **sendmail,** что аутентификацию нужно проводить с использованием SASL. Теперь займитесь созданием базы данных всех пользователей, которые могут отправлять почту. Для этого используются две программы: **saslpasswd и sasldblistusers.** Они должны находиться в каталоге /sbin. Запускать их нужно от имени пользователя root.

```
# saslpasswd -a sendmail newuser
password:<type passwdord for newuser>
```

Эту процедуру требуется провести для всех пользователей, которым разрешено отправление почты. Затем используйте вторую программу:

```
sasldblistusers
```

Она применяется для просмотра всех записей в базе данных. После ее запуска вы должны увидеть что-то наподобие этого:

```
user: newuser realm: dhsilabs.com mech: CRAM-MD5
user: newuser realm: dhsilabs.com mech: DIGEST-MD5
user: newuser realm: dhsilabs.com mech: PLAIN
```

Отображенная информация означает, что пользователь newuser может аутентифицироваться тремя методами: CRAM-MD5, DIGEST-MD5, PLAIN. Рекомендую использовать метод CRAM-MD5, но в крайнем случае подойдет и PLAIN.

Далее проверьте, поддерживает ли ваш **sendmail** библиотеку SASL:

```
sendmail -d0.1 -bv root I grep SASL
```

При отсутствии поддержки SASL от вас потребуется перекомпилировать **sendmail.** Вот для чего я просил в начале раздела приготовить исходные коды **sendmail.** Итак, распакуйте **sendmail,** как обычно, программой **tar:**

```
tar -xzf senmail-x.xx.xx.tar.gz
cd sendmail-x.xx.xx/
```

Теперь нужно создать файл sendmail-x.xx.x/devtools/Site/site.config.m4, в котором необходимо прописать следующие строки:

```
APPENDDEF('confENVDEF', '-DSASL')
APPENDDEF('conf_sendmail_LIBS', '-lsasl')
APPENDDEF('confLIBDIRS', '-L/usr/lib/')
APPENDDEF('confINCDIRS', '-I/usr/include/')
```

Напомню, создать файл в простейшем случае можно командой:

```
cat > sendmail-x.xx.x/devtools/Site/site.config.m4
```

Самое время запустить скрипт **Build:**

```
./Build
./Build install
```

Если вы все сделали правильно, ваш **sendmail** теперь должен поддерживать SMTP AUTH. Проверить это можно с помощью уже знакомой команды:

```
sendmail -d0.1 -bv root | grep SASL
```

После этого приступите к настройке самой программы **sendmail.** Для этого в файл sendmail.mc внесите следующие строки:

```
TRUST_AUTH_MECH('GSSAPI DIGEST-MD5 CRAM-MD5 PLAIN')dnl
define('confAUTH_MECHANISMS', 'GSSAPI DIGEST-MD5 CRAM-MD5 PLAIN')dnl
define('confDEF_AUTH_INFO', '/etc/mail/auth/auth-info')dnl
FEATURE('no_default_msa')dnl turn off default entry for MSA
DAEMON_OPTIONS('Port=25, Name=MSA, M=E')dnl
```

Метод PLAIN, как самый ненадежный, можно было бы убрать из списка авторизации, но я не советую вам этого делать, так как не все почтовые клиенты поддерживают не только метод PLAIN, а и SMTP аутентификацию вообще.

Запустите интерпретатор m4:

```
m4 senmail.mc > sendmail.cf
```

Скопируйте новый файл sendmail.cf на место старого, обычно он находится в каталоге /etc/mail:

```
cp ./sendmail.cf /etc/mail/sendmail.cf
```

Почти все! Осталось проверить работоспособность **sendmail** и убедиться, что он работает корректно. С этой целью запустите клиент telnet и присоединитесь к порту 25 вашего компьютера:

```
telnet localhost 25
Trying 127.0.0.1...
Connected to localhost
Escape character is '^]'.
220 local.sendmail.ORG ESMTP Sendmail 8.10.0/8.10.0; Thu, 9 Sep
1999 10:48:44 -0700 (PDT)
ehlo localhost
250-local.sendmail.ORG Hello localhost [127.0.0.1], pleased to
meet you
250-ENHANCEDSTATUSCODES
250-DSN
250-AUTH DIGEST-MD5 CRAM-MD5 PLAIN
250 HELP
quit
```

Теперь желательно добавить описания поддерживаемых вашим сервером методов аутентификации. Откройте файл /etc/mail/sendmail.cf в любом текстовом редакторе и найдите следующие строки:

```
##########################
I Format of headers #
##########################
```

После этих строк вам нужно добавить следующее:

```
$.$?{auth_type}(auth_type is ${auth_type}, user
${auth_author}$.)
```

Хочу заметить, что файл sendmail.cf у вас может находиться в другом каталоге. Это делается для того, чтобы в заголовке письма появилось такое сообщение:

```
(auth_type is CRAM-MD5, user den)
```

Указание метода авторизации поможет вам при дальнейшей настройке вашего сервера.

13.3. Настройка почтовых клиентов

В предыдущих разделах данной главы мы рассмотрели два типа настройки программы **sendmail** — без аутентификации и с наличием таковой. Поэтому сначала рассмотрим настройку почтовых клиентов для первого случая, а потом — для второго. Ради написания этого материала мне даже пришлось установить **Outlook Express.** Этой программы не наблюдалось на моей машине с момента выхода первой версии **TheBat!**

Итак, рассмотрим настройку трех самых популярных почтовых клиентов:

1. TheBat!, версия 1.38e.
2. Outlook Express, версия 5.00.
3. Netscape Messenger, версия 4.76.

Я специально не стал использовать более новые версии программ, потому что не у всех пользователей они могут оказаться. Это, правда, не относится к Outlook — более ранней версии я не нашел. Все более новые версии программ настраиваются абсолютно аналогично.

Начнем с предпочитаемой мною программы — **TheBat!**

Для создания новой учетной записи выберите пункт меню **Account→** New... В появившемся окне введите название новой учетной записи, затем собственную информацию, серверы исходящей почты (SMTP server) и сервер для входящих сообщений (POP3 server). Затем введите имя пользователя и пароль. Пароль должен быть установлен на сервере с помощью команды **passwd!** Нажмите на кнопку «Finish» и учетная запись готова. Проверить параметры учетной записи вы можете с помощью меню **Account -> Properties.** Для нашего «виртуального» почтового сервера, используемого в примерах, настройки будут выглядеть так как это показано на рис. 13.3.

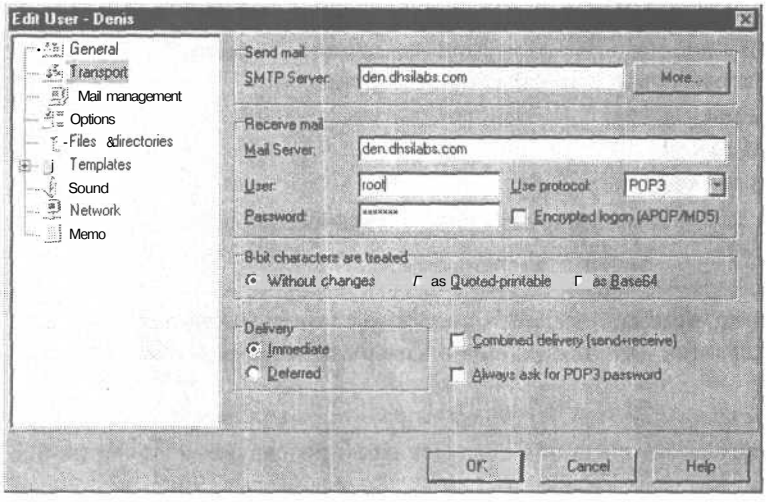


Рис. 73.3. *Параметры учетной записи (программа TheBat!)*

На рис. 13.3. видно, что в качестве сервера для входящей и исходящей почты используется компьютер den.dhsilabs.com. Протокол сервера входящей почты — POP3. Параметры протокола (порт, порядок удаления сообщения) можно указать в разделе **Mail management.** Остальные параметры не представляют интереса. Имя пользователя и пароль должны быть такими же, какие вы используете для локальной регистрации на своем компьютере, в данном случае на компьютере den.dhsilabs.com.

Теперь перейдем к **Outlook Express.** Выберите пункт меню **Сервис** → **Учетные записи** и нажмите на кнопку «Добавить». Затем выберите пункт подменю «Почта» и введите аналогичные параметры. После создания учетной записи нажмите на кнопку «Свойства» и еще проверьте ее параметры. На вкладке **Серверы** можно указать серверы для приема и передачи сообщений, а на вкладке **Дополнительно** — установить параметры протоколов.

Почтовый клиент **Netscape Messenger** настраивается так: выберите пункт меню **Edit** → **Preferences.** Причем это можно сделать из любой программы, входящей в пакет Netscape Communicator. В появившемся окне **Preferences** (рис. 13.4) перейдите в раздел **Почтовые серверы** (Mail servers) и укажите необходимые вам параметры. Параметры протокола POP можно задать, выбрав почтовый сервер и нажав на кнопку «Edit».

Я уже успел рассмотреть конфигурирование всех трех программ без использования аутентификации. Теперь же давайте настроим эти же программы, но уже используя аутентификацию. Я буду использовать те же учетные записи, чтобы не создавать новые. Начнем по-порядку.

Программа TheBat!

Возле поля ввода сервера для отправления сообщений есть кнопка «More» (см. рис. 13.3). При нажатии на нее откроется окно **Advanced SMTP Options.**

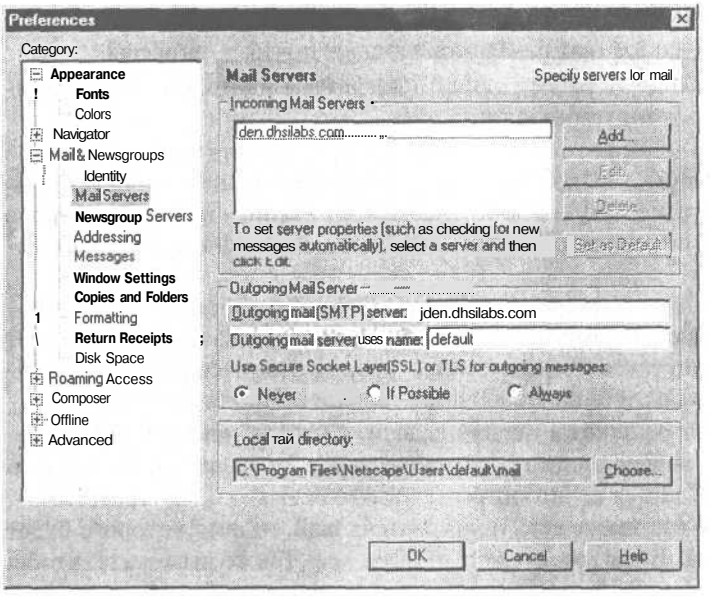


Рис. 13.4. *Настройка Netscape Messenger*

Установите режим **Perform SMTP authenication** (RFC 2554). Если имя пользователя и пароль на сервере POP совпадают с именем пользователя и паролем на сервере SMTP, а это обычно так, установите режим **Use POP server login.** В противном случае, укажите нужное имя пользователя и пароль.

Программа Outlook Express

Откройте окно свойств учетной записи и перейдите на вкладку **Серверы.** Включите режим «Проверка подлинности пользователя», а затем нажмите на кнопку «Настройка» и установите параметры аутентификации.

Netscape Messenger

Здесь все аналогично предыдущим программам — установить имя пользователя для сервера SMTP можно в разделе **Mail servers** (см. рис. 13.4), включить аутентификацию там же. Хочу заметить, что Netscape Messenger (версия 4.76) поддерживает только метод авторизации PLAIN.

13.4. Другие программы для работы с электронной почтой

Кроме агентов MTA и демонов POP-сервиса существуют также программы для получения почты, POP-клиенты и программы для сортировки почты.

Конечно, любая почтовая программа, например, **kmail** или **Outlook,** обладают встроенным POP-клиентом, но, кроме сбора почты, почтовые программы, как правило, обладают массой других не менее полезных функций — создание, отправка сообщений и тому подобные.

В качестве POP-клиента рассмотрим стандартный почтовый клиент Linux — **fetchmail**, а также программу сортировки почты — procmail.

Лучше всего объяснять работу любой программы на практическом примере. Сначала сформулируем задачи:

1. На один ящик mail@firma.ru поступает почта для всех отделов какой-либо организации. Вам нужно выполнить сортировку приходящей почты по названию отдела, например, если в теле письма или в одном из его заголовков упоминается название отдела, отправить это сообщение одному из пользователей отдела.
2. Настроить автоответчик электронной почты. Этот автоответчик будет работать подобно обыкновенному автоответчику: когда пользователь получает сообщение, автоответчик автоматически отправит ответ, содержащий примерно такое сообщение: «Ваше сообщение получил. Прочитаю в 19:00».

Теперь займемся решением первой задачи. Сначала немного конкретизируем ее. Пусть у вашей организации существует 3 отдела. Адрес первого отдела — `dep1@firma.ru`, второго — `dep2@firma.ru`, а третьего — `dep3@firma.ru`. У вас также есть пользователь mail, на имя которого будет приходить вся почта. Его адрес — `mail@firma.ru`. Также вы хотите, чтобы рассылка проекта LinuxRSP отправлялась вам по адресу `adm@firma.ru`.

Для начала в домашнем каталоге пользователя mail создайте файл `.procmailrc` примерно такого содержания, как это показано в листинге 13.2 :

Листинг 13.2. Файл ***.procmailrc***

```
:0
* ^Subject:.*dep1
! dep1
:0
* ^Subject:.*dep2
! dep2
:0
* ^Sybject:.*dep3
! dep3
:0
* ^Sybject:.*LinuxRSP
! adm
```

Таким образом, вы определили правила сортировки почты. Если в теме (заголовок Subject) будет присутствовать название отдела, то сообщение будет отправлено нужному пользователю.

Примечание.

Если вы создавали файл .procmailrc, зарегистрировавшись в системе как пользователь root, измените права доступа к этому файлу:

```
chown mail.mail .procmailrc
chmod 600 .procmailrc
```

Можно выполнить сортировку по полю **From** или любому другому, например, последнее правило могло бы выглядеть так:

```
:0
* ^From:.* Subscribe.Ru
! adm
```

Но в этом случае, если кто-нибудь из пользователей вашей системы также подпишется на другую рассылку на сервере Subscribe.Ru, то вся почта попадет к пользователю adm и вам придется читать рассылку вместе.

Примечание.

Подробнее о правилах сортировки вы прочитаете в справочной системе Linux, введя команду ***man procmail****.*

В этом же каталоге ($HOME/mail) создайте файл .forward. Если вы работаете как пользователь root, установите права доступа к этому файлу так же как и для файла .procmailrc. В этом файле задаются правила перенаправления почты. Добавьте в него следующую строку:

```
|IFS=' ' && exec /usr/bin/procmail USER= <mail>
```

Обычно программа **procmail** находится в каталоге /usr/bin. Если вы при самостоятельной сборке программы указали другой каталог, измените команду перенаправления в файле .forward.

Теперь создайте файл .fetchmailrc. Этот файл нужно создать в каталоге того пользователя, от имени которого будет запускаться **fetchmail**. В этот файл добавьте следующие строки:

```
set postmaster "mail"
poll provider.ru proto POP3 no dns
    user "mail" pass "my_password" to mail here
options fetchall
```

Таким образом вы установите постмастера (пользователь mail. provider.ru) — имя почтового сервера, откуда вы будете забирать почту по протоколу POP3. При этом вы будете использовать имя пользователя **mail** и пароль **my_password**. Опция **fetchall** указывает программе получить всю почту и потом удалить полученные сообщения на сервера provider.ru.

Запускать программу **fetchmail** можно как демон, а можно с помощью планировщика **crontab**. В первом случае просто выполните команду:

```
fetchmail -d 12000
```

При этом **fetchmail** будет проверять наличие новой почты через каждые 20 минут. Во втором случае выполните команду **crontab** -e и введите новое задание:

```
0,20,40 * * * * /usr/bin/fetchmail
```

Теперь перейдем к решению второй задачи. Напомню, что наша цель – создание автоответчика. Существуют два типа автоответчиков. Первые посылают автоответ только на определенные сообщения, а вторые — на все. Например, вам нужно отправить клиенту прайс-лист вашей организации по его требованию. Это первый тип автоответчика. Если же вы уезжаете летом на недельку отдохнуть куда-нибудь и хотите, чтобы программа сообщила людям, что вы в отпуске — это второй тип.

Займемся настройкой первого типа. Для этого в ваш файл .procmailrc добавьте строки:

```
0:
* ^Subject.*Price
I (formail -r ; cat $HOME/pricelist.zip)
| sendmail -t
```

Как видите все намного проще, чем ожидалось. А второй тип автоответчика создается еще проще:

```
0:
I (formail -r; cat $HOME/info.txt)
I sendmail -t
```

Заметьте — вы не определяете никаких условий, поэтому ответное письмо будет отправлено всем, кто напишет вам сообщение. В файл info.txt нужно записать ваш автоответ.

Сделаем небольшой вывод. Программа **fetchmail** используется для загрузки сообщений, а **procmail** — для ее сортировки. Рекомендую вам изучить параметры программ **fetchmail и procmail** — вы найдете их в документации по этим программам. Используйте **procmail** с большой осторожностью, потому что если вы неправильно укажете условия сортировки, почта будет просто утеряна без возможности восстановления.

13.5. Создание списка рассылки

Средствами Linux можно создать небольшую рассылку сообщений электронной почты. Для больших систем рассылки я не рекомендовал бы использовать вам этот метод. Обычно системы рассылки создаются средствами, специально предназначенными для этого, например, PHP в связке с MySQL идеально подходят для этого. Язык программирования PHP предназначен для создания Web-приложений и оснащен всеми необходимыми для этого функциями, а сервер баз данных MySQL обеспечит поддержку базы данных адресов подписчиков и параметры рассылки. Таким образом, если вы хотите создать собственный MailList.Ru, воспользуйтесь готовыми решениями или напишите собственную на PHP или Perl.

Однако иногда бывает полезно создать небольшую рассылку внутри одной организации. Приведенное далее решение не отличается оригинальностью и не претендует на звание лучшей системы рассылки. С помощью этого примера вы также узнаете, как использовать стандартную почтовую утилиту Linux — **mail.** Эта программа входит в состав практически каждой Unix-системы.

Итак, допустим у вас есть три отдела: отдел маркетинга, производственный отдел и администрация. К первому отделу относятся пользователи вашей системы marina и oleg, ко второму — igor, dmitry, olya, а к третьему — president, director, secretar. Периодически вам нужно отправлять сообщения в один из отделов. Число пользователей небольшое и, возможно, отправить сообщение можно было бы с помощью групп пользователей почтовой программы, которую вы используете. Однако сейчас я покажу, как элегантно

это можно сделать средствами Linux. Тем более, что для этого не нужно устанавливать никаких почтовых программ.

Создайте файл .mailrc в вашем домашнем каталоге и добавьте в него строки:

```
alias market marina oleg
alias proizv igor dmitry olya
alias adm director secretar
```

В дальнейшем, чтобы отправить сообщение в производственный отдел просто введите команду:

```
mail proizv
```

Программа **mail** попросит вас ввести тему, а затем текст сообщения. Для окончания ввода нажмите Ctrl+D и **mail** отправит сообщения пользователям.

При создании псевдонимов убедитесь, что в вашей системе нет пользователей с таким именем.

Как видите, средствами одной программы **mail** вам удалось сделать немного. Если пользователей много, использовать механизм псевдонимов не очень удобно. Гораздо удобнее, чтобы программа **mail** брала список подписчиков из какого-нибудь файла. К сожалению, разработчики этой программы не предусмотрели такой возможности, однако с помощью небольшого сценария мы можем организовать эту возможность. Посмотрим, что из этого получится, если воспользоваться средствами интерпретатора команд bash.

Создайте сценарий **smaller** в своем домашнем каталоге (см. листинг 13.3).

Листинг 13.3. Сценарий smailer

```
#!/bin/bash
DT='date'
echo $DT >> log
for user in 'cat users'
do
echo "Sending message to $user"
mail $user -s Subscribe < msg 2» log
done
```

Напомню, что сценарий — это обыкновенный текстовый файл с установленным атрибутом выполнения, содержащий команды интерпретатора. Более подробно сценарии рассматриваются в гл. 19, а сейчас попробую объяснить все на интуитивном уровне, чтобы вы поняли сам принцип работы вышеприведенного сценария.

В первой строке указывается, какой интерпретатор будет выполнять ваши команды — /bin/bash. Обратите внимание на одинарные кавычки, в которые заключена команда **date.** Это не обыкновенные кавычки, а те, которые расположены под знаком тильды «~». Во второй и третьей строках заносится дата в файл протокола log. Этот файл находится в домашнем каталоге.

Затем последовательно (в цикле for) читается список пользователей из файла users, который находится в домашнем каталоге, и передается имя пользователя программе **mail.** Программа **mail** поочередно отправит сообщение каждому пользователю из файла users. Тема сообщения устанавливается с помощью параметра -s, а само сообщение хранится в файле msg.

Весь поток ошибок перенаправляется в файл log. **Вы** можете переправить и все сообщения программы **mail,** удалив дескриптор 2 перед символами перенаправления ввода/вывода.

Теперь создайте файл users. Текст моего личного файла приведен в листинге 13.4.

Листинг 13.4 Файл users

```
den
u
synthetic
evg
```

Сообщения, которые вы хотите отправить, запишите в файл msg. Запустите сценарий командой:

```
./smailer
```

На экране вы должны увидеть следующую информацию:

```
Sending message to den
Sending message to u
Sending message to synthetic
Sending message to evg
```

Так как пользователя *evg* в моей системе нет, в файл протокола будет записано соответствующее сообщение:

```
Сбт Май 4 10:43:28 EEST 2002
evg... User unknown /root/dead.letter...
Saved message in /root/dead.letter
```

На экран не будет выведено сообщение об ошибке, потому что мы перенаправили стандартный поток ошибок в файл log. Как видите, все предельно просто.

Программу **mail** можно использовать и для чтения почты. Для этого просто введите команду:

```
mail
```

Если в вашем почтовом ящике нет сообщений, программа сообщит вам об этом:

```
No mail for user
```

где user — это ваше имя пользователя.

Если будут новые сообщения, программа выведет на экран нумерованный список, и вы сможете ввести номер сообщения, которое хотите прочитать. Для удаления сообщения используется **команда d <номер> или d <диапазон>.** Выйти из программы можно, введя команду q.

Использовать программу **mail** в качестве полноценной почтовой программы нельзя — она работает только с вашим локальным ящиком, поэтому если вам нужно будет получить сообщения откуда-то извне, скажем pop.mail.ru, вы не сможете сделать это. Я бы вам порекомендовал использовать программу **kmail,** которая входит в состав KDE. Данная программа поддерживает несколько учетных записей электронной почты, в том числе и локальный ящик, отправку сообщений с помощью SMTP и локального агента MTA **(sendmail),** а также сообщения в формате HTML, что является немаловажным, когда пользователи Outlook забывают включить текстовый формат для исходящих сообщений.

14

Бастионы

14.1. Применение IPChains

Для начала небольшое вступление. Читатель, скорее всего, знает, что весь трафик в сети состоит из пакетов. Каждый пакет состоит из двух частей: заголовка и тела. В заголовке пакета находится информация об источнике, адресате, типе пакета, а также прочая информация, которая характерна для пакетов определенных типов. В теле пакета передается та информация, которую мы хотим передать. Более подробно об этом было сказано в гл. 1 данной книги.

Протокол TCP, в отличие от UDP, перед началом передачи данных требует установки соединения. Перед установкой соединения производится обмен специальными пакетами, а после этого передаются обычные пакеты, содержащие данные.

Бастион (firewall, брандмауэр) — это системный компонент (фильтр), обеспечивающий защиту сети от несанкционированного доступа. **IPChains** представляет собой пакетный фильтр. Пакетный фильтр просматривает заголовок каждого пакета, который проходит через него, а потом решает, что делать со всем пакетом. Фильтрация пакетов встроена в ядро ОС Linux.

Обычно **IPChains** используется на шлюзах, соединяющих две сети, например, локальную и Интернет. При этом вы имеете право разрешить передавать или принимать какие-либо пакеты. Это позволяет обеспечить должный уровень безопасности и экономии.

С помощью **IPChains** можно ограничить свою хорошо организованную сеть от хаоса Интернет. Применение IPChains может решить такие виды атак, как пинг смерти, атака на отказ, IP-спуфинг, фрагментация пакетов. В этой главе будут описаны способы защиты от перечисленных атак на ваш сервер, а также будут приведены рекомендации по выявлению источника атак.

Для поддержки IPChains вам необходимо перекомпилировать ядро. О том, как это сделать рассказано в гл. 18. При этом вам нужно включить опции ядра **IP:firewalling и IP:firewall packet.** Я также рекомендую включить

опции IP: masquerading и IP: always defragment. Скорее всего, они также вам понадобятся. Также вам следует включить **IP-Forwarding** (если вы еще это не **сделали) командой:**

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

Если же вам нужна поддержка динамических IP-адресов (например, для DHCP — см. гл. 8), включите ее с помощью команды:

```
echo "1" > /proc/sys/net/ipv4/ip_dynaddr
```

74.2. *Настройка IPChains*

Настройку IPChains лучше всего рассматривать на практических примерах. Но прежде, чем это сделать попробуем разобраться, как ядро фильтрует пакеты.

Ядро стартует с тремя списками правил: **input, forward, output.** Эти правила называются **firewall-цепочками** или просто цепочками. Когда вы получаете новый пакет, ядро использует цепочку **input** (входная цепочка). Перед этим пакет проверяется: не был ли он поврежден при пересылке? Поврежденные пакеты отвергаются. Если с пакетом все в порядке, пакет проходит проверку целостности: не запутает ли пакет правила фильтрации? Если пакет не проходит проверку целостности, он отвергается.

Если фильтр пропускает пакет, то ядро решает, куда его направить дальше. Это называется маршрутизацией. Если пакет предназначен для другой машины, то пакет должен пройти через цепочку **forward.** Если пакет проходит через фильтр **forward,** то удаленная машина получит этот пакет. Что делать с этим пакетом дальше, будет решать уже получившая его машина и для нас это не представляет интереса.

Если пакет предназначен для нашей машины, пакетный фильтр определяет, на какой интерфейс его следует переслать. Пакет, предназначенный для интерфейса **1o** (обратная петля), сначала проходит выходную цепочку, а потом попадает во входную цепочку интерфейса lo. Если пакет создан локальным процессом, который запущен на бастионе, он попадает в цепочку перенаправления **(forward).** Цепочка перенаправления применяется ко всем пакетам, проходящим сквозь бастион на другие машины.

Прежде чем пакет выйдет из сетевого интерфейса, он должен пройти цепочку **output.**

Цепочка — это список правил. Каждое правило можно сформулировать примерно так: «если заголовок пакета удовлетворяет такому-то условию, то поступить с ним так-то». Если первое правило не применимо к этому пакету, то рассматривается следующее правило в цепочке. Если ни одно правило не применимо к пакету, пакет, скорее всего, будет отвергнут. Если какой-нибудь пакет отвергается, в журнале syslog появляется соответствующее сообщение.

Программу **ipchains** можно вызывать со множеством параметров. Одни параметры создают новые цепочки, другие добавляют правила в цепочки (см. табл. 14.1).

Параметры программы ipchains *Таблица 14.1*

Параметр	Описание
-N	Создать новую цепочку
-X	Удалить пустую цепочку
-P	Изменить стратегию для пустой цепочки
-L	Вывести правила цепочки
-F	Удалить все правила из цепочки
-Z	Обнулить счетчики пакетов и байтов во всех правилах цепочки
-A	Добавить новое правило к цепочке
-I	Вставить новое правило в определенную позицию
-R	Заменить правило
-D	Удалить правило, удовлетворяющее какому-нибудь условию

Один из примеров использования IPChains мы уже рассматривали в главе 8. Напомню, в той главе нужно было организовать маршрутизацию средствами IPChains. При этом использовались следующие команды:

```
ipchains -P forward DENY
ipchains -A forward -s 192.168.1.0/24 -d 192.168.2.0/24 -j ACCEPT
ipchains -A forward -s 192.168.2.0/24 -d 192.168.1.0/24 -j ACCEPT
```

Первая команда изменяет стратегию для пустой цепочки **forward,** которая используется именно для маршрутизации. Вторая команда перенаправляет пакеты из сети 192.168.1.0 в сеть 192.168.2.0. Опция -s означает источник (source), а опция -d означает назначение (destination). Опция -j определяет действие. В том случае, если вы должны принять пакет, используется параметр ACCEPT. Если вам нужно отбросить пакет, используйте параметр DENY.

Адреса источника и пункта назначения можно указывать четырьмя способами:

1. Просто указать IP-адрес, например, 192.168.1.1.
2. Указать имя компьютера, например, www.host.ru.
3. Указать целую группу IP-адресов, например, 192.168.1.0/24.
4. Указать группу адресов и маску сети, например, 192.168.1.0/255.255.255.0.

Для формирования правил также используются опции, представленные в табл. 14.2.

Все эти опции могут иметь предшествующий параметру символ «!». Как и в некоторых языках программирования, этот символ обозначает отрицание. Например, правило -s! **localhost** определяет любой пакет, не исходящий из компьютера localhost.

Опции формирования правил *Таблица 14.2*

Параметр	Описание
B	Источник (IP-адрес или URL-адрес)
-d	Назначение (IP-адрес или URL-адрес)
-i	Интерфейс
-P	Протокол

Указать протокол можно с помощью опции -p. Например, для указания протокола TCP служит правило -p TCP. Иногда нужно указать порт соединения TCP или UDP. Это можно сделать так:

```
IP-addr/ports_range.
```

Диапазон портов определяется параметром **ports_range**. Диапазон указывается через двоеточие, например, 0:1023. Нижний предел диапазона равен О, а верхний — 1023. Если нижний предел не задан, принимается значение по умолчанию — 0. Если не задан верхний предел, считается, что он равен 65535, то есть максимальному числу поддерживаемых портов.

Указать интерфейс можно с помощью опции -i. Например, **-i pppO.** Если у вас несколько интерфейсов, названия которых начинаются символами ppp, указать их всех вы можете опцией **-i ppp+.** Под интерфейсом понимается физическое устройство, на (из) который приходит (уходит) пакет. Для того, чтобы просмотреть список доступных в текущий момент интерфейсов, используйте команду *ifconfig*.

Теперь рассмотрим пару полезных примеров. Возможно, вы захотите запретить использование telnet извне. Это можно сделать с помощью команды:

```
ipchains -A prov -p tcp –destination-port 23 -j REJECT
```

Через цепочку **prov** проходит весь трафик, идущий от провайдера. Ее можно создать командами:

```
ipchains -N prov
ipchains -A input -i pppO -j prov
```

Предполагается, что вы подключаетесь к провайдеру через интерфейс **ppp0**.

Скорее всего, в вашей сети найдутся несколько машин, которые работают под управлением ОС Windows и имеют общие ресурсы. Возможно, вы даже настроили протокол SMB на сервере. Чтобы протокол SMB не был заметен извне, что очень нежелательно, воспользуйтесь следующими командами:

```
ipchains -A prov -p tcp –destination-port 137 -j REJECT
ipchains -A prov -p udp –destination-port 137 -j REJECT
ipchains -A prov -p tcp –destination-port 138 -j REJECT
ipchains -A prov -p udp –destination-port 138 -j REJECT
ipchains -A prov -p tcp –destination-port 139 -j REJECT
ipchains -A prov -p udp –destination-port 139 -j REJECT
```

Можно запретить локальным процессам получать пакеты от определенных узлов. Например, я не хочу, чтобы мой Netscape тратил время на получения баннеров от машины с адресом 911.111.111.111. Для этого, возможно, удобнее было бы воспользоваться прокси-сервером, но сейчас нужно продемонстрировать, как это можно сделать с помощью IPChains.

```
ipchains -A output -d 911.111.111.111 -j REJECT
```

В примере я специально привел несуществующий IP-адрес, чтобы не создать антирекламу какому-нибудь узлу Сети.

Для того, чтобы ваши правила были постоянными (при перезагрузке машины правила IPChains теряются), используйте сценарии **ipchains-save и ipchains-restore.** Настройте свои правила, затем выполните команду:

```
# ipchains-save > /etc/ipchains.rules
```

В листинге 14.1 представлен сценарий, управляющий пакетной фильтрацией.

Листинг 14.1. Сценарий управления пакетной фильтрацией

```
#! /bin/sh
# Сценарий управления пакетной фильтрацией.
# Если правил нет, то ничего не делать.
[ -f /etc/ipchains.rules ] || exit 0
case "$1" in
  start)
    echo -n "Включение пакетной фильтрации:"
    /sbin/ipchains-restore < /etc/ipchains.rules || exit 1
    echo 1 > /proc/sys/net/ipv4/ip_forward
    echo "."  ;;
 stop)
    echo -n "Отключение пакетной фильтрации:"
    echo 0 > /proc/sys/net/ipv4/ip_forward
    /sbin/ipchains -X
    /sbin/ipchains -F
    /sbin/ipchains -P input ACCEPT
    /sbin/ipchains -P output ACCEPT
    /sbin/ipchains -P forward ACCEPT
    echo "."  ;;
 *)
    echo "Использование: /etc/init.d/packetfilter {start|stop}"
    exit 1 ;;
esac
exit 0
```

Этот сценарий нужно добавить в сценарии загрузки системы.

14.3. Различные примеры

В этом пункте представлены несколько примеров для обеспечения безопасности вашей сети.

14.3.1. Пакеты SYN

Пакеты SYN используются для запроса на установку соединения. Вы можете отвергать эти пакеты для того, чтобы прервать попытки установить соединение.

Иногда это необходимо, если вы хотите получать пакеты только в одном направлении, например, рабочая станция должна соединяться с сервером, но сервер не должен соединяться с рабочей станцией.

Для фильтрации пакетов SYN нужно использовать опцию -y. Например, попытки соединения по протоколу TCP от узла 192.168.1.34 указываются так:

```
-p TCP -s 192.168.1.34 -y
```

14.3.2. Фрагментация пакетов

Иногда передаваемый пакет слишком большой, чтобы его можно было бы передавать за один раз. Если такое происходит, то пакет делится на

фрагменты, и эти фрагменты пересылаются. Компьютер, которому этот пакет предназначен, собирает эти фрагменты в один пакет.

Ядро должно анализировать начало пакета, которое содержится в первом фрагменте. Ведь только в первом фрагменте находится заголовок исходного пакета. Для решения этой проблемы вам нужно перекомпилировать ядро с включенной опцией **IP: always defragment.**

В результате любое правило фильтрации не будет работать для фрагментированных пакетов. При этом только первый фрагмент будет обработан как пакет, а остальные не будут обрабатываться. Однако можно определить правило специально для фрагментированных пакетов. Это можно сделать с помощью опции -f. Эту опцию нельзя применять при указании портов протоколов TCP или UDP, кода ICMP или пакетов SYN.

Следующая команда отбросит все фрагменты, приходящие на компьютер server.domain.com:

```
# ipchains -A output -f -d 192.168.1.1 -j DENY
```

14.3.3. Пинг смерти

Есть хорошая новость по этому поводу: ОС Linux невосприимчива к пингу смерти. Напомню, что пинг смерти заключается в посылке большого пакета ICMP, который переполняет буферы стека TCP на компьютере-получателе.

14.3.4. IP-спуфинг

IP-спуфинг — это отправление пакетов с поддельным IP-адресом источника. Так как фильтрация пакета принимает решения на основании адреса источника, то IP-спуфинг используется, чтобы ввести пакетный фильтр в заблуждение.

Для решения этой проблемы мы можем использовать **Проверку Адреса Источника** (Source Address Verification) или использовать следующие команды IPChains:

```
ipchains -A prov -s 192.168.1.1/16 -1 -j DENY
ipchains -A prov -s 127.0.0.1/8 -1 -j DENY
```

Вторая команда нужна для ядер версий 2.0.x, но если мы ее укажем, явно хуже не будет. Опция -1 позволяет протоколировать «плохие» пакеты. Файлом протокола является /var/log/messages. Ядра версий 2.1.x автоматически отклоняют пакеты, приходящие с адресов 127.*, которые зарезервированы для локального интерфейса.

Для включения проверки адреса источника можно воспользоваться сценарием, представленном в листинге 14.2.

Листинг 14.2. Запрещение IP-спуфинга

```
# Наилучший способ: включить Source Address Verification и защи-
тить
# от спуфинга все текущие и будущие интерфейсы.
  if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ] ; then
```

```
    echo -n "Установка защиты от спуфинга... "
    for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
      echo 1 > $f
    done
    echo "готово."
  else
    echo ПРОБЛЕМЫ ПРИ ПОПЫТКЕ ВКЛЮЧИТЬ ЗАЩИТУ ОТ СПУФИНГА.
    echo "Нажмите CONTROL-D для выхода в shell и продолжения сист. загрузки."
    echo
    #, Запуск однопользовательской оболочки на консоли
    /sbin/sulogin $CONSOLE
  fi
```

Защита с использованием проверки адреса источника является более надежной.

14.3.5. Фильтрация фрагментированных бомб

Иногда возникает такая ситуация, когда на машину приходят не все фрагменты. Такое бывает при очень большом количестве фрагментов. Обычно при использовании ОС Linux данная проблема вообще отпадает, но все-таки некоторые администраторы предпочитают «перестраховаться».

Для решения этой проблемы вам нужно всего лишь перекомпилировать ядро с включенной опцией IP: always defragment. При этом нужно учитывать, чтобы ваш шлюз был единственным маршрутом для пакетов. Вторым способом является фильтрация фрагментов, но это может отразиться на нормальных пакетах. Честно говоря, я не особо забочусь об этой проблеме при настройке своего сервера.

14.4. Практический пример

А теперь рассмотрим более серьезный пример. Этот пример частично позаимствован мною из руководства DNS-HOWTO. Но прежде чем перейти непосредственно к практике, попробую объяснить некоторые термины, которые будут встречаться ниже.

Прежде всего, определимся, что называется **маскарадингом**. Объяснение я приведу на сугубо формальном языке. **Маскарадинг** перезаписывает заголовки пакетов, когда они проходят через шлюз так, чтобы казалось, что они всегда исходят от шлюза непосредственно. Затем он перезаписывает ответы так, чтобы клиенту казалось, что они пришли от первоначального получателя. Например, у вас есть шлюз и одна небольшая локальная сеть. Шлюз обладает реальным IP-адресом 1.1.1.1, а адрес вашей локальной сети — 192.168.1.0. Клиент с IP-адресом 192.168.1.5 пытается обратиться к узлу http://www.romb.net. IP-адрес этого узла 62.244.59.193. Пакет клиента проходит через шлюз. IP-адрес шлюза в локальной сети — 192.168.1.1. Шлюз перезаписывает заголовок пакета и устанавливает вместо адреса клиента свой собственный адрес, то есть 1.1.1.1. После получения ответа, он перед

отправлением пакета клиенту опять перезаписывает заголовок пакета и изменяет свой адрес 1.1.1.1 на адрес узла www.romb.net — 62.244.59.193. С точки зрения узла www.romb.net соединение было установлено между адресами 1.1.1.1 и 62.244.59.193. С точки зрения клиента соединение произошло между интерфейсами с адресами 192.168.1.5 и 62.244.59.193.

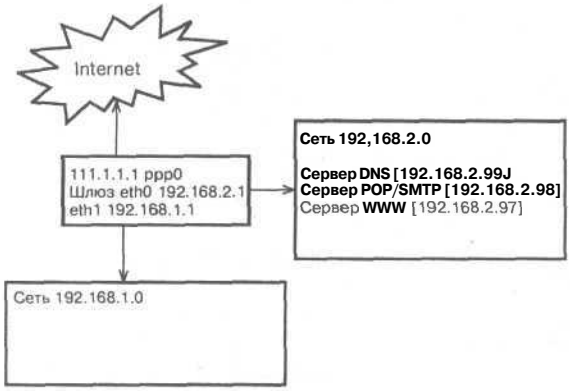


Рис. 14.1. *Структура сети*

Итак, приступим к рассмотрению практического примера. Предположим, у нас имеется соединение с Интернет и две подсети. Интерфейсу `ppp0` назначен реальный IP-адрес — 111.1.1.1, интерфейсу ethO назначен IP-адрес внутренней сети — 192.168.2.1, а интерфейсу `eth1` - 192.168.1.1. (см. рис. 14.1).

Нам нужно обеспечить маршрутизацию между тремя сетями: Интернет, 192.168.2.0 и 192.168.1.0. Другими словами, требуется таким образом настроить пакетную фильтрацию, чтобы можно было пропинговать любую сеть или выполнить трассировку пакетов через любую сеть. Пинговать сеть будем, естественно, с помощью программы ping, а выполнять трассировку будем программой **traceroute**. В ОС Windows NT те же операции можно выполнить с помощью программ **ping и tracert** соответственно.

Нам также нужно, чтобы сервер WWW обрабатывал как запросы из внутренних сетей, так и запросы пользователей Internet. Сервер SMTP должен принимать внутренние и внешние соединения, а также отправлять почту в Internet. Получать почту по протоколу POP3 могут только пользователи внутренних сетей. Сервер DNS также должен обрабатывать запросы от всех сетей.

Еще раз определим, к чему будут иметь доступ пользователи Интернет:

1. Наш внутренний сервер WWW.
2. Наш сервер FTP.
3. Наш сервер DNS.
4. Сервер SMTP.

Пользователи локальных сетей будут иметь доступ к:

1. Серверу WWW нашей сети.
2. Серверу FTP нашей сети.
3. Серверу SMTP для отправки почты, как пользователям локальной сети, так и пользователям Интернет.
4. Серверу DNS нашей сети, а также к серверам DNS сети Интернет.
5. Серверу POP3 для получения почты.
6. Серверам WWW сети Интернет.
7. Серверам FTP сети Интернет.

Наши пользователи также должны иметь возможность использовать программы **ping, traceroute,** ssh. Чуть не забыл! Нам же нужно также обеспечить нормальную работу клиента ICQ. Эта программа стала уже стандартом, как Netscape или Internet Explorer.

Прежде, чем настраивать пакетный фильтр, убедимся, что мы запретили IP-спуфинг и правильно настроили все сетевые интерфейсы. В этой главе уже приводился более подробный пример запрещения IP-спуфинга (листинг 14.2). Эту задачу можно попробовать решить одной командой (при этом вы должны использовать интерпретатор bash):

```
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do echo 1 > $f; done
```

Теперь установим правила, которые запрещают любые пакеты, кроме пакетов обратной петли (loopback):

```
# ipchains -A input -i ! lo -j DENY
# ipchains -A output -i ! lo -j DENY
# ipchains -A forward -j DENY
```

Обратите внимание, что запрет IP-спуфинга и любого трафика, кроме локального, должен быть выполнен до инициализации интерфейсов. В противном случае существует вероятность того, что сквозь наш «бастион» проникнут пакеты.

Очень желательно вставить модуль **ip_masq_ftp** для макарадинга сервера FTP. Благодаря этому наш внутренний FTP-сервер сможет работать в активном и пассивном режимах.

Теперь создадим несколько цепочек. Все они будут отфильтровывать проходящие пакеты, то есть будут аналогичны цепочке **forward.** Название каждой из них определяется направлением передачи пакетов, например, **netl-net2** — по этой цепочке пакеты будут передаваться от сети 192.168.1.0 к сети 192.168.2.0.

```
ipchains -N netl-net2
ipchains -N net1-inet
ipchains -N net2-netl
ipchains -N net2-inet
ipchains -N inet-net2
ipchians -N inet-net1
```

Также создадим цепочку для приема ICMP-сообщений:

```
ipchains -N icmp
```

В цепочке **forward** мы знаем только исходящий интерфейс, а для выяснения входящего интерфейса, то есть того, из которого пришел пакет, мы используем адрес источника. Подделать этот адрес невозможно, так как мы запретили IP-спуфинг. Выполним следующие команды:

```
ipchains -A forward -s 192.168.1.0/24 -i eth0 -j netl-net2
ipchains -A forward -s 192.168.1.0/24 -i pppo -j netl-inet
ipchains -A forward -s 192.168.2.0/24 -i pppo -j net2-inet
ipchains -A forward -s 192.168.2.0/24 -i ethl -j net2-netl
ipchains -A forward -i eth0 -j inet-net2
ipchains -A forward -i ethl -j int-net1
ipchains -A forward -j DENY -1
```

Теперь определим правила для цепочки приема ICMP-сообщений:

```
ipchains -A icmp -p icmp -icmp-type destination-unreachable -j ACCEPT
ipchains -A icmp -p icmp -icmp-type source-quench -j ACCEPT
ipchains -A icmp -p. icmp -icmp-type time-exceeded -j ACCEPT
ipchains -A icmp -p icmp -icmp-type parameter-problem -j ACCEPT
```

Мы будем принимать только ICMP-сообщения об ошибках, все остальные приниматься не будут. Далее определим правила для цепочки **netl-net2.** Как уже было сказано выше, от нас требуется обеспечить доступ к сервисам WWW, FTP, ssh. Также нужно разрешить доступ к серверам SMTP, POP3, DNS, использование программ **traceroute и ping** (все отклоненные пакеты мы будем регистрировать в журнале). С этой целью определим следующие правила:

```
ipchains -A netl-net2 -p tcp -d 192.84.219.128 smtp -j ACCEPT
ipchains -A netl-net2 -p tcp -d 192.84.219.128 pop-3 -j ACCEPT
ipchains -A netl-net2 -p udp -d 192.84.219.129 domain -j ACCEPT
ipchains -A netl-net2 -p tcp -d 192.84.219.129 domain -j ACCEPT
ipchains -A netl-net2 -p tcp -d 192.84.218.130 www -jj ACCEPT
ipchains -A netl-net2 -p tcp -d 192.84.218.130 rsync -j ACCEPT
ipchains -A netl-net2 -p icmp -j icmp
ipchains -A netl-net2 -j DENY -1
```

Эти правила также разрешают вызов **rsync** к серверу Web. Теперь определим правила для цепочки **inet-net2.** Так как в сети 192.168.2.0 находятся серверы SMTP, DNS и Web, то определим ограничения для них. Почтовый сервер должен отправлять почту во внешнюю сеть (Интернет), а также принимать почту из внешней сети. На прием почты по протоколу POP3 имеют право только пользователи внутренней сети. Сервер имен (DNS-сервер) должен посылать запросы во внешнюю сеть, а также принимать запросы из внешней сети через шлюз. Сервер Web должен принимать запросы от пользователей всех сетей. Доступ **rsync** разрешен только для пользователей внутренних сетей. Все это реализуется следующими правилами:

```
ipchains -A inet-net2 -p tcp -d 192.168.2.98 smtp -j ACCEPT
ipchains -A inet-net2 -p udp -d 192.168.2.99 domain -j ACCEPT
ipchains -A inet-net2 -p tcp -d 192.168.2.99 domain -j ACCEPT
ipchains -A inet-net2 -p tcp -d 192.168.2.97 www -j ACCEPT
ipchains -A inet-net2 -p icmp -j icmp
ipchains -A inet-net2 -j DENY
```

Далее задаем правила цепочки **net1-inet**. Пользователи внутренней сети могут получать доступ к сервисам WWW, FTP внешней сети, использовать **traceroute** во внешнюю сеть. Нужно разрешить доступ к почтовому серверу, серверу имен, Web-серверу. Модуль **masq** обеспечит пассивный доступ к серверу FTP. Возможные нарушения будут регистрироваться. Пользователи внутренней сети также смогут использовать программу **ssh** для доступа к внешним узлам, и программу **ping.**

```
ipchains -A netl-inet -p tcp -dport www -j MASQ
ipchains -A netl-inet -p tcp -dport ssh -j MASQ
ipchains -A netl-inet -p udp -dport 33434:33500 -j MASQ
ipchains -A netl-inet -p tcp -dport ftp -j MASQ
```

```
ipchains -A net1-inet -p icmp —icmp-type ping -j MASQ
ipchains -A net1-inet -j REJECT -1
```

Сейчас займемся определением правил для цепочки net2-net1. Пользователи могут получать доступ к серверам Web, SMTP, DNS, POP3. Как и для предыдущего случая, мы будем использовать модуль masq для пассивного режима работы FTP-сервера и будем регистрировать нарушения. Правила для этой цепочки будут таковыми:

```
ipchains -A net2-netl -p tcp ! -y -s 192. 84 .219 .128 smtp -j ACCEPT
ipchains -A net2-netl -p udp -s 192. 84 .219 .129 domain -j ACCEPT
ipchains -A net2-netl -p tcp ! -y -s 192. 84 .219.129 domain -j ACCEPT
ipchains -A net2-net1 -p tcp ! -y -s 192. 84 .218 .130 www -j ACCEPT
ipchains -A net2-netl -p tcp ! -y -s 192. 84 .218 .130 rsync -j ACCEPT
ipchains -A net2-netl -p icmp -j icmp
ipchains -A net2-inet -j DENY -1
```

Правила для цепочки net2-inet выглядят так:

```
ipchains -A net2-inet -p tcp -s 192.84.219.128 smtp -j ACCEPT
ipchains -A net2-inet -p udp -s 192.84.219.129 domain -j ACCEPT
ipchains -A net2-inet -p tcp -s 192.84.219.129 domain -j ACCEPT
ipchains -A net2-inet -p tcp ! -y-s 192.84.218.130 www -j ACCEPT
ipchains -A net2-inet -p icmp -j icmp
ipchains -A net2-inet -j DENY -1
```

Эти правила разрешают отправлять почту во внешнюю сеть, принимать почту из внешней и внутренней сети, получать почту только пользователям внутренней сети. Сервер DNS имеет право посылать запросы во внешнюю сеть, принимать запросы от внутренней и внешней сетей, а также от шлюза. Сервер WWW принимает запросы от пользователей внутренней и внешней сетей. Доступ rsync разрешен пользователям внутренней сети.

Следующая цепочка — это inet-net1. В этом случае мы не разрешаем никакого доступа из внешней сети к машинам внутренней сети.

```
ipchains -A inet-netl -j REJECT
```

Основные правила уже определены, осталось установить правила для цепочки input шлюза. Создадим три цепочки input для каждого возможного адресата:

```
ipchains -N inet-if
ipchains -N net2-if
ipchains -N net1-if
```

По первой цепочке будут приходить пакеты от внешней сети, а по второй и третьей — от внутренних сетей. Правила для этих цепочек выглядят так:

```
ipchains -A input -d 192.84.219.1 -j inet-if
ipchains -A input -d 192.84.219.250 -j net2-if
ipchains -A input -d 192.168.1.250 -j netl-if
```

Непосредственно для цепочки inet-if определим такие правила:

```
ipchaiain-A inet-if -i ! PPpO -j DENY -1
ipchains -A inet-if -p TCP —dport 61000:65096 -j ACCEPT
```

```
ipchains -A inet-if  -p  UDP-dport       61000:65096 -j ACCEPT
ipchains -A inet-if  -p  ICMP-icmp-type  pong         -j ACCEPT
ipchains -A inet-if  -j  icmp
ipchains -A inet-if  -j  DENY
```

Данные правила разрешают пропинговать любую сеть, использовать программу **traceroute** для любой сети, доступ к серверу имен, а также получать ICMP-сообщения об ошибках.

Цепочку **net2-if** определим следующим образом:

```
ipchains -A net2-if -i !     eth0  -j DENY
ipchains -A net2-if -p TCP   ! -y -s 192.168.2.99 53 -j ACCEPT
ipchains -A net2-if -p UDP   -s 192.168.2.99 53 -j ACCEPT
ipchains -A net2-if -p ICMP  -icmp-type pong -j ACCEPT
ipchains -A net2-if -j icmp
ipchains -A net2-if -j DENY  -l
```

Правила этой цепочки разрешают те же операции, что и для цепочки **inet-if,** только в этом случае вместо интерфейса pppO используется ethO.

Входящая цепочка **net1-if** определяется так:

```
ipchains -A netl-if -i ! eth1 -j DENY
ipchains -A netl-if -p ICMP -icmp-type ping -j ACCEPT
ipchains -A netl-if -p ICMP -icmp-type pong -j ACCEPT
ipchains -A netl-if -j icmp
ipchains -A netl-if -j DENY  -l
```

Разрешается доступ к серверам WWW, SMTP, POP3. Можно использовать программы **ping, tracer oute, ssh.**

Теперь осталось удалить правила блокировки:

```
ipchains -D input 1
ipchains -D forward 1
ipchains -D output 1
```

В начале этого пункта я обещал объяснить настройку **ICQ.** Предположим, что сервер SQUID у вас установлен на шлюзе, то есть на той машине, которая обеспечивает пакетную фильтрацию (о прокси-сервере SQUID читайте в следующей главе). В файле конфигурации SQUID разрешите **порт 5190.** Именно этот порт используется новыми клиентами **ICQ.**

```
acl SSL_ports port 443 563 5190
```

Затем установите сервер **socks.** Можно, конечно, настроить **ICQ,** используя маскарадинг, но данный метод, как мне кажется, лучше. В качестве сервера **socks** я рекомендую использовать сервер dame-socks. В файле конфигурации /etc/socks . conf установите внутренний и внешний адреса шлюза:

```
internal: 192.168.1.1 port = 1080
external: 111.1.1.1
```

Затем определите узлы, которые могут использовать **socks:**

```
client pass {
    from: 192.168.0.0/16 to: 0.0.0.0/0
    }
```

Осталось определить, кто может отвечать клиентам:

```
pass {
    from: 0.0.0.0/0 to: 192.168.0.0/16
    command: bindreply udpreply
    log: connect error
}
```

14.5. IPTables

Пакетный фильтр **IPChains** использовался в ядрах Linux до версии 2.4. В новых версиях ядра (начиная с 2.4) вместо IPChains используется пакетный фильтр **IPTables.** Практически все основные опции остаются прежними. Только, естественно, в командной строке вместо **ipchains** следует писать **iptables**.

В этом пункте будут рассмотрены некоторые новые опции ядра, связанные непосредственно с **IPTables.** Эти опции для большей наглядности я представил в табл. 14.3. Если вы еще не знакомы с компилированием ядра, то эта информация вам пригодится после прочтения гл. 18.

Опции ядра ***Таблица 14.3***

Опция	Описание
CONFIG_PACKET	Позволяет использовать программы, которые работают непосредственно с сетевым устройством. Примером такой программы может послужить tcpdump
CONFIG_NETFILTER	Включите данную опцию, если вы собираетесь использовать ваш компьютер в качестве шлюза
CONFIG_IP_NF_CONNTRACK	Позволяет отслеживать соединения. Данная опция необходима для работы функций NAT или IP-маскарадинга. В случае со шлюзом включите данную опцию
CONFIG_IP_NF_FTP	Прослеживает FTP-соединения. Включите эту опцию, если на вашем компьютере установлен FTP-сервер. Из-за большого количества FTP-запросов модуль IP_NF_CONNTRACK не в состоянии проследить все FTP-соединения, поэтому в помощь ему добавлена опция CONFIG_IP_NF_FTP
CONFIG_IP_NF_IPTABLES	Необходимая для работы IPTables опция. Без ее включения вы не сможете использовать IPTables
CONFIG_IP_NF_MATCH_LIMIT	Необязательная опция. Позволяет ограничить количество пакетов передаваемых/принимаемых за промежуток времени
CONFIG_IP_NF_MATCH_MAC	Позволяет блокировать пакеты, используя MAC-адрес (а не IP-адрес)

Все остальные опции, связанные с **IPTables,** содержат в своем названии слово MATCH, например, CONFIG_IP_NF_MATCH_MARK. Эти опции разрешают выполнять определенные действия над пакетами. Действия над пакетами, как вы уже знаете, задаются с помощью опции **-j**. Использование этой опции аналогично как для **IPChains, так и для IPTables.** Назначение всех возможных действий вы можете узнать из документации по **IPTables.** Ранее, на примере **IPChains,** мы рассмотрели два основных действия — ACCEPT и DENY. В **IPTables** вместо действия DENY нужно использовать аналогичное ему действие DROP. А вместо MASQ (см. п. 14.4) — MASQUERADE.

Я не буду рассматривать различия таблиц (tables) и цепочек (chains), не усложняя тем самым вам жизнь избытком лишней информации, остановлюсь лишь на рассмотрении некоторых полезных примеров.

Представим, что вам нужно что-либо сделать с пакетами, приходящими от узла с MAC-адресом 11:12:13:14:15:16. Выделить данный пакет можно с помощью правила iptables -A INPUT **—mac-source** 11:12:13:14:15:16.

Дальнейшие операции над пакетом вы уже определите сами, с помощью опции **-j**. Если же вам нужно выделить все пакеты, кроме тех, которые присылает вам узел с MAC-адресом 11:12:13:14:15:16, то нужно использовать отрицание — !

Допустим, что теперь вам нужно ограничить число пакетов, присылаемым узлом с MAC-адресом 11:12:13:14:15:16, например, не более 10 пакетов в минуту. Делается это следующим образом: iptables -A INPUT -ra limit **—limit 10/minute**. Промежуток времени можно указать в секундах (second), минутах (minute) и часах (hour).

IPTables позволяет выделять (а потом производить с ними операции) пакеты одновременно, указав несколько портов, например:

```
iptables -A INPUT -p tcp -m multiport -source-port 22,53,80,110
```

Порты указываются через запятую. Вы можете указать максимум 15 портов. Вместо портов источника вы можете указать порты назначения, используя опцию **—destination-port**. Если вы хотите одновременно указать как порты источника, так и порты назначения, используйте опцию **—port**:

```
iptables -A INPUT -p tcp -m multiport -port 22,53,80,110
```

Теперь уже перейдем на более высокий уровень. Если раньше мы могли отфильтровывать пакеты, исходящие от определенного компьютера, то теперь мы можем выделять пакеты отдельных пользователей. Например, вам нужно выделить все исходящие от пользователя с UID 500 пакеты. Это можно сделать с помощью команды:

```
iptables -A OUTPUT -m owner —uid-owner 500
```

Естественно, вы сможете это сделать только для исходящих пакетов, поскольку вы не знаете, какой UID имеет пользователь другой системы, тем более, что информация об этом не передается по протоколу TCP.

Аналогично вы можете ограничивать исходящие пакеты группы или процесса:

```
iptables -A OUTPUT -m owner —gid-owner 0
iptables -A OUTPUT -m owner —pid-owner 78
```

Пакетный фильтр IPTables обладает значительно большими возможностями по сравнению с IPChains, но на практике вы вряд ли будете их использовать: в основном используются возможности IPChains, описанные в п. 14.1...14.4.

15

Прокси-сервер SQUID

15.1. Что такое SQUID?

SQUID — это программа, которая получает HTTP/FTP-запросы клиентов и по ним обращается к ресурсам Интернет. Применение прокси-сервера (squid) дает возможность использовать фиктивные IP-адреса во внутренней сети (Masquerading — маскарадинг), увеличивает скорость обработки запроса при повторном обращении (кэширование), а также обеспечивает дополнительную безопасность.

Нет смысла устанавливать прокси на своей домашней машине, так как функции кэширования выполняет браузер. Прокси-сервер стоит применять лишь в том случае, если в вашей сети три-четыре компьютера, которым нужен выход в Интернет. В этом случае запрос от браузера к прокси-серверу обрабатывается быстрее, чем от браузера к ресурсам Интернет, и таким образом увеличивается производительность. При этом можно смело установить размер кэша в браузерах клиентов равным нулю.

SQUID — это нечто большее, чем просто прокси-сервер. Это своеобразный стандарт кэширования информации в сети Интернет. В силу повсеместной распространенности SQUID, в книге я уделил его конфигурированию большое внимание.

Прокси-сервер Squid образуется несколькими программами, в числе которых: сама программа сервера squid, а также программа **dnsserver** — программа обработки DNS-запросов. Когда запускается программа squid, то она сначала запускает заданное количество процессов dnsserver, каждый из которых работает самостоятельно и может осуществлять только один поиск в системе DNS. За счет этого уменьшается общее время ожидания ответа DNS.

15.2. Установка SQUID

SQUID может быть установлен из исходных текстов или в виде RPM-пакета. Установка RPM-пакета SQUID очень проста — для этого нужно ввести команду **rpm -ih squid-2.3.STABLE2-3mdk.i586.rpm**.

Я использую версию squid 2.3. Более новая версия доступна в виде исходных кодов. Исходники можно получить по адресу **ftp://ftp.squid.org.** Для распаковки исходных кодов, выполните следующие команды:

```
cd /usr/src/
gunzip squid-2.3.STABLE2-3-src.tar.gz
tar xvf squid-2.3.STABLE2-3-src.tar.gz
cd squid
```

Теперь перейдем непосредственно к установке:

```
./configure –prefix=/usr/local/squid
make all
make install
```

SQUID будет установлен в каталог, заданный ключом **prefix — /usr/local/squid.** Помимо **prefix** можно пользоваться ключами, представленными в табл. 15.1.

Ключи сценария configure *Таблица 15.1*

Ключ	Описание
--enable-icmp	Измерять путь до каждого HTTP-сервера при запросах с помощью ICMP
--enable-snmp	Включить SNMP-мониторинг
--enable-delay-pools	Управление трафиком
--disable-wccp	Отключить Web Cache Coordination Protocol
--enable-kill-parent-hack	Более корректный shutdown
--enable-splaytree	Позволяет увеличить скорость обработки ACL

/5.3. *Настройка SQUID*

Сервер SQUID использует файл конфигурации `squid.conf`, который обычно располагается в каталоге /etc/squid (или /usr/local/squid/etc — более ранние версии). Откройте его в любом текстовом редакторе, например, **joe/usr/local/squid/etc/squid.conf**. Далее выполните следующую последовательность действий:

1. Укажите прокси провайдера:

   ```
   cache_peer proxy.isp.ru
   ```

   В данном случает proxy.isp.ru становится нашим «соседом» (neighbour, peer).

2. Установите объем памяти, доступный **squid,** и каталог для кэша:

   ```
   cache_mem 65536
   cache_dir /usr/local/squid/cache 1024 16 256
   ```

   где: 65536 — объем оперативной памяти в байтах, который можно использовать под кэш;

   1024 • - количество мегабайт, отводимое на диске в указанном каталоге под кэш. В этом каталоге будут храниться кэшированные файлы. Стоит ли говорить, что если у вас несколько жестких дисков, то кэш нужно разместить на самом быстром из них.

3. Укажите хосты, которым разрешен доступ к прокси-серверу:

   ```
   acl allowed_hosts src 192.168.1.0/255.255.255.0
   ```

```
acl localhost src 127.0.0.1/255.255.255.255
```

4. Укажите разрешенные SSL-порты:

```
acl SSL_ports port 443 563
```

5. Запретите метод CONNECT для всех портов, кроме указанных в acl SSL_ports:

```
http_access deny CONNECT !SSL_ports
```

и запретите доступ всем, кроме тех, кому можно:

```
http_access allow localhost
http_access allow allowed_hosts
http_access allow SSL_ports http_access deny all
```

6. Пропишите пользователей, которым разрешено пользоваться **squid** (в рассматриваемом примере это den, admin, developer):

```
ident_lookup on
acl allowed_users user den admin developer
http_access allow allowed_users
http_access deny all
```

Тэги **maxium_object_size и maxium_object** устанавливают ограничения на размер передаваемых объектов.

Ниже приведен пример запрета доступа к любому URL, который соответствует шаблону games и разрешения доступа ко всем остальным:

```
acl GaMS url_regex games
http_access deny GaMS
http_access allow all
```

15.4. Запуск SQUID

Теперь, когда вы выполнили базовую настройку SQUID, его нужно запустить:

```
/usr/local/squid/bin/squid -z
```

Параметр -z необходим для создания (обнуления) каталога, содержащего кэш. Обычно этот параметр нужен только при первом запуске. Некоторые другие полезные параметры SQUID представлены в табл. 15.2.

Параметры SQUID *Таблица 15.2*

Параметр	Описание
-a порт	Задает порт для входящих HTTP-запросов
-d	Включает режим вывода отладочной информации в стандартный поток ошибок (на stderr)
-f файл	Задает файл конфигурации
-h	Выдает справочную информацию
-k reconfigure	Посылает сигнал HUP
-k shutdown	Завершение работы прокси-сервера
-k kill	Завершение без закрытия журналов
-u порт	Задает порт для входящих ICP-запросов
-s	Включает журналирование с помощью syslog
-v	Выдает информацию о версии SQUID
-D	Не делать DNS-тест при запуске
-N	Не становиться демоном (фоновым процессом)
-Y	Более быстрое восстановление после сбоев

15.5. Формат файла squid.conf

В файле `squid.conf` задаются всевозможные параметры конфигурации прокси-сервера. Давайте рассмотрим их все по-порядку.

15.5.1. Параметры сети

Порт для запросов клиентов (см. рис. 15.1): `http_port 3128`

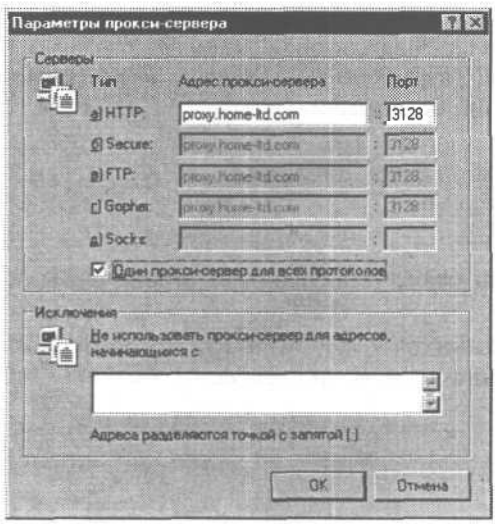


Рис. 15.1. *Параметры прокси-сервера*

Если «соседей» (peer) нет, то установите `icp_port` 0

```
icp_port 3130
```

Порт для общения с соседями — ICP — через TCP. При использовании этого параметра нужно установить ключ **—enable-htcp** при установке **htcp_port 4827.**

Следующий параметр указывает, по какому адресу нужно принимать входящие пакеты, если хост имеет несколько интерфейсов. В версии 2.3 этого параметра нет:

```
tcp_incoming_address  0.0.0.0
```

При отправлении информации указанный адрес будет использован в качестве исходного:

```
tcp_outgoing_address 0.0.0.0
```

То же, но для ICP:

```
udp_outgoing_address 0.0.0.0 (аналогично, для ICP)
```

То же, но для ICP (при приеме):

```
udp_incoming_address 0.0.0.0
```

По умолчанию этот режим включен, но если прокси-сервер находится за бастионом (firewall), то параметр **passive_ftp** нужно выключить:

```
passive_ftp on | off
```

15.5.2. Параметры соседей

Описание соседей производится строками следующего формата:

```
cache_peer hostname type proxy-port icp-port options
```

где: **hostname** — имя соседа;
type — тип соседа: **parent** — старший, **sibling** — одного уровня;
proxy-port — порт прокси-сервера;
icp-port — порт ICP;
options — параметры.

При этом каждый сосед прописывается отдельной строкой.

Parent — при отсутствии запроса в локальном кэше он перенаправляется к **parent;** тот, если запроса не окажется в его кэше, пересылает его дальше и

т.д. Возвращает готовый ответ подчиненному. Если **squid** получает TCP_DENIED от **parent,** то обращение к ресурсу будет идти напрямую.

Sibling — при отсутствии запроса в локальном кэше, запрос перенаправляется в **sibling;** при отсутствии запроса в нем, возвращает сообщение об этом, никаких дополнительных действий не предпринимается.

15.5.3. Управление кэшем

cache_swap_high число

При достижении этого уровня заполнения кэша (в процентном соотношении) начинается ускоренный процесс удаления старых объектов.

cache_swap_low 90

Процесс удаления прекращается при достижении этого уровня.

maximum_object_size 4096 KB

Максимальный размер кэшируемого объекта.

minimum_object_size O KB

Файлы меньшего размера не сохраняются.

15.5.4. Протоколирование

Ниже перечислены режимы протоколирования SQUID с указанием соответствующих журналов. Если какой-то журнал вам не нужен, установите **none** вместо имени файла.

cache_access_log /usr/local/squid/logs/access.log

Протоколируется каждый запрос к SQUID. Журнал называется /usr/local/squid/logs/access.log.

cache_log /usr/local/squid/logs/cache.log

Протоколируются запуски процессов. Журнал называется /usr/local/squid/logs/cache.log.

cache_store_log /usr/local/squid/logs/store.log

Протоколируются записи объектов в кэш. Журнал называется /usr/local/squid/logs/store.log.

15.5.5. Параметры внешних программ

ftp_user email-адрес

Указанный здесь email будет использоваться вместо пароля при анонимном доступе к ftp-серверам.

dns_nameservers список IP-адресов

Значение данного параметра используется вместо того списка DNS-серверов, который определен в файле /etc/resolv.conf; по умолчанию — **none.**

cache_dns_program /usr/local/squid/bin/dnsserver

Данный параметр задает программу разрешения IP-адресов в имена (сервер DNS).

authenticate_program none

Позволяет производить аутентификацию клиентов, делающих запросы. При этом должен быть определен ACL proxy_auth.

authenticate_program /usr/local/squid/bin/ncsa_auth /usr/local/squid/etc/passwd

Традиционная программа аутентификации. Определена в ../auth_modules/NCSA.

15.5.6. *Списки ACL*

ACL (Access Control Lists) — списки контроля доступа. Довольно часто возникает потребность группировки однотипных параметров в единое целое для их последующей обработки. Для эффективного решения этой задачи используются списки контроля доступом (ACL). Например:

acl SSL_ports port 443 563

Эта запись означает, что создается список SSL_ports типа port. Элементами списка являются номера портов 443 и 563.

Добавить новый элемент к уже существующему списку (параметр **add)** можно так:

acl add SSL_ports port 999

Удалить ненужный элемент можно с помощью параметра del:

acl del SSL_ports 999

Переименовать список позволяет параметр **геп** (от rename):

acl ren SSL_ports Allowed_ports

Удалить все списки вместе с их содержимым позволяет параметр **flush:**

acl flush

Стандарт ACL требует, чтобы перед именем списка обязательно был указан символ $. Другими словами, все перечисленные выше примеры по большому счету должны быть неправильными. Например, для создания списка нужно использовать запись:

acl $SSL_ports port 443 563

Однако большинство фильтров, например SQUID, пренебрегают этим требованием, и вы можете указывать имена списков без знака доллара.

Итак, ACL — это определение списка доступа. Имеет следующий формат:

acl имя тип строка

где: тип — это тип объекта;

строка — регулярное выражение.

Можно использовать список:

acl имя тип имя_файла

Перечисление параметров производится по одному параметру в строке. Типы, которые можно использовать при составлении списков ACL, указаны в табл. 15.3.

Типы ACL *Таблица 15.3*

Тип	Описание типа
Src ip-address/netmask	Задает IP-адрес клиентов
Src addr1-addr2/netmask	Указывает диапазон адресов
Dst ip-address/netmask	Задает URL хостов
Time [day-abbrevs] [h1:m1-h2:m2]	Время, где день — это одна буква из SMTWHFA
Port	Список портов
Port port1-port2	Диапазон портов
Proto	Протокол — HTTP или FTP
Method	Метод — GET или POST
Browser [-i] regexp	Сравнивается заголовок User-Agent

[-i] — игнорируется регистр букв.

15.5.7. Параметры доступа

http_access allow|deny aclname

Разрешать доступ к прокси по HTTP.

icp_access allow|deny aclname

Разрешать доступ к прокси по ICP.

miss_access allow|deny aclname

Разрешить получать ответ MISS от вас.

cache_peer_access cache-host allow|deny aclname

Ограничить запросы к данному соседу — расширение для **cache_peer_domain**.

proxy_auth_realm Squid proxy-caching web server

Строка текста, которая будет выдана на экран клиента при запросе имени/пароля доступа к кешу.

15.5.8. Параметры администрирования

cache_mgr email

Данный параметр задает почтовый адрес, на который будет послано письмо, если **squid** перестанет функционировать.

cache_effective_usernobody

При запуске SQUID от имени root изменить UID на указанный в параметре **cache_effective_user**.

cache_effective_group nogroup

При запуске SQUID от имени root изменить GID на указанный в параметре **cache_effective_group**.

visible_hostnameимя_хоста

Это имя будет упоминаться в сообщениях об ошибках.

hostname_aliasesимя

Этот параметр задает список синонимов для имени хоста.

15.6. Отказ от рекламы. Баннерный фильтр

Вам не хочется тратить лишнее время на загрузку рекламных баннеров? Мне тоже. К счастью, SQUID позволяет достаточно просто решить эту проблему. Просто вставьте следующие строки в свой файл /usr/local/etc/squid/squid.conf:

```
acl good_url      url_regex       "/usr/local/etc/squid/acl/good_url"
acl bad_urlpath urlpath_regex "/usr/local/etc/squid/acl/bad_urlpath"
acl bad_url       url_regex       "/usr/local/etc/squid/acl/bad_url"
http_access deny bad_urlpath !good_url
http_access deny bad_url       !good_url
```

Соответственно, нужно будет создать три файла: good_url, bad_url_path и bad_url. В файл bad_url следует поместить «плохие» URL, например: ^http://.*doubleclick

```
^http://.*-ad.flycast.com/server/img/
^http://1000.stars.ru/cgi-bin/1000.cgi
^http://1000.stars.ru/cgi-bin/1000.cgi
^http://12.16.1.10/~web_ani/
```

А в файл bad_url_path — «плохой» путь, например:

```
88x31.*gif
88x31.*GIF
100x80.*gif
100x80.*GIF
100x100.*gif
100x100.*GIF
120x60.*gif
120x60.*GIF
179x69.*gif
193x72.*gif
468x60.*gif
```

Обычно такие имена имеют баннеры.

Примеры файлов good_url, bad_url_path и bad_url можно взять на моей домашней страничке — http://dkws.narod.ru

15.7. Разделение канала

Допустим, вам нужно настроить прокси-сервер таким образом, чтобы одна группа компьютеров могла работать с одной скоростью, а другая — с другой. Это может потребоваться, например, для разграничения пользователей, которые используют канал для работы, и пользователей, которые используют ресурсы канала в домашних целях. Естественно, первым пропускная способность канала важнее, чем вторым. С помощью прокси-сервера SQUID можно разделить канал.

Для начала в файле конфигурации укажите, сколько пулов, то есть групп пользователей, у вас будет:

```
delay_pools 2
```

Затем определите классы пулов. Всего существует три класса:

1. Используется одно ограничение пропускной способности канала на всех.
2. Одно общее ограничение и 255 отдельных для каждого узла сети класса С.
3. Для каждой подсети класса В будет использовано собственное ограничение и отдельное ограничение для каждого узла.

В файл `squid.conf` добавьте следующие директивы:

```
delay_class 1 1 # определяет первый пул класса 1 для домашних пользователей
delay_class 2 2 # определяет второй пул класса 2 для служащих
```

Теперь задайте узлы, которые будут относиться к пулам:

```
acl home src адреса
acl workers src адреса
delay_access 1 allow home
delay_access 1 deny all
delay_access 2 allow workers
delay_access 2 deny all
```

Затем укажите ограничения:

```
delay_parameters 1 14400/14400
delay_parameters 2 33600/33600 16800/33600
```

Как я уже отмечал выше, для пула класса 1 используется одно ограничение для всех компьютеров, входящих в пул — 14400 байт. Первое число задает скорость заполнения для всего пула (байт/секунду). Второе — максимальное ограничение.

Для пула класса 2, соответственно, используются ограничения на всю подсеть и отдельно на каждого пользователя. Если бы у нас был определен пул класса 3, то для него ограничения выглядели примерно так:

```
delay_parameters 3 128000/128000 64000/128000 12800/64000
```

Первые два числа задают соответственно скорость заполнения и максимальное ограничение для всех. Следующая пара чисел определяет скорость заполнения для каждой подсети и максимальное ограничение, а третья — скорость заполнения и максимальное ограничение для индивидуального пользователя.

15.8. Программы для учета трафика

Для мониторинга работы SQUID и вообще для учета трафика можно воспользоваться следующими программами:

sqmgrlog.....http://www.ineparnet.com.br/orso/index.html
mrtg.....http://www.switch.ch/misc/leinen/snmp/perl/
iptraf.....http://dkws.narod.ru/linux/soft/iptraf-2.4.0.tar.gz
bandmin.....http://www.bandmin.org
webalizer (анализ работы Apache) .... http://www.mrunix.net/webalizer/

Вместе с данными программами поставляется довольно читаемая документация, поэтому подробно на их использовании я останавливаться не буду. Программа MRTG описана в п. 8.5.

15.9. Настройка клиентов

После того, как вы настроили прокси-сервер, давайте займемся настройкой клиентов, то есть браузеров пользователя. Я не сомневаюсь, что вы знаете, как настраивать тот или иной браузер, я лишь напомню процедуру настройки для некоторых распространенных браузеров.

Internet Explorer 5

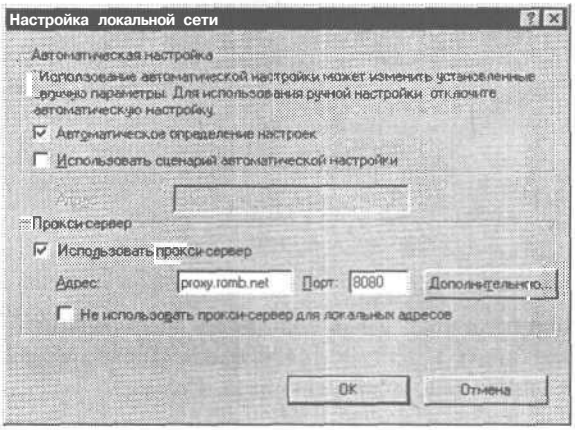


Рис. 15.2. *Настройка Internet Explorer*

Меню **Сервис → Свойства обозревателя → вкладка Подключение → Настройка сети. В** появившемся окне установите необходимые параметры, то есть имя прокси-сервера и его порт (см. рис. 15.2).

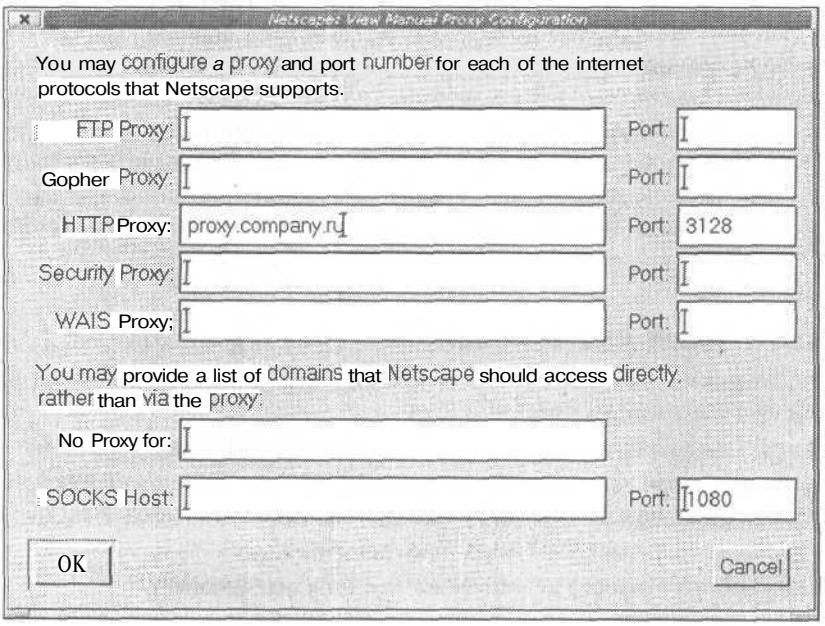


Рис. 15.3. *Настройка Netscape Communicator*

Netscape Communicator

Меню Edit → **Preferences** → **Advanced** → **Proxies** → **Manual Proxy Configuration** → **View** (см. рис. 15.3).

Konqueror

Меню **Настройки** → **Настройки** → **Прокси** (см. рис. 15.4).

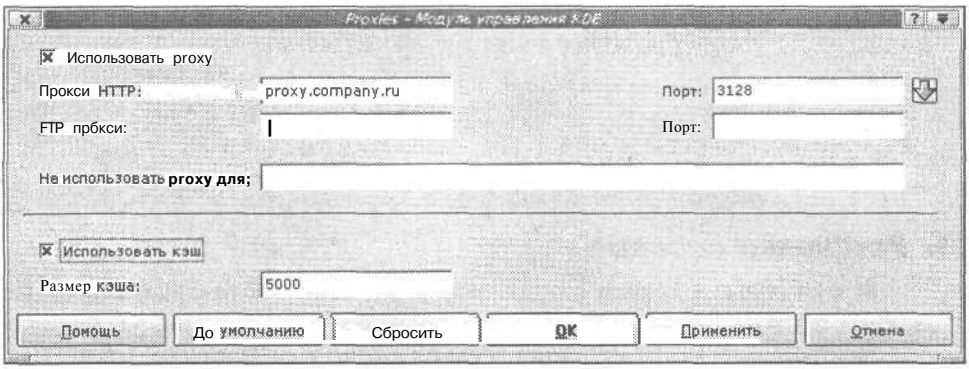


Рис. 15.4. *Настройка Konqueror*

16

Установка MySQL

16.1. Установка сервера

В этой главе я кратко опишу, как установить популярный сервер баз данных MySQL. Благодаря своей простоте сервер MySQL приобрел широкое распространение в сети Интернет. При создании Интернет-сервера вы просто не сможете обойтись без сервера MySQL: большинство провайдеров предоставляет хостинг вместе с сервером баз данных, большинство дизайнеров выбирают MySQL, потому что практически невозможно построить серьезный Интернет-проект без использования баз данных.

Я здесь не буду рассматривать технические характеристики сервера MySQL и не буду сравнивать его с другими серверами баз данных, такими, как InterBase Server, IBM DB/2, Oracle. MySQL идеально подходит практически для любого Интернет-проекта. Естественно, если вы создаете распределенную систему обработки информации, вам лучше использовать InterBase Server или Oracle, поскольку эти СУБД являются более масштабируемыми, чем MySQL.

Итак, приступим к настройке сервера. Прежде всего нужно установить пакеты, необходимые для работы MySQL.

Я использую MySQL версии 3.20, поэтому я установил такие пакеты:

MySQL_GPL-3.20.32a-18mdk.
MySQL_GPL-client-3.20.32a-18mdk.
MySQL_GPL-shared-libs-3.20.32a-18mdk.
MySQL_GPL-bench-3.20.32a-18mdk.
MySQL_GPL-resolveip-3.20.32a-18mdk.

Проще всего воспользоваться командой **rpm** -ih MySQL* для установки сервера. Возможно, вы не захотите устанавливать клиентскую версию MySQL-client, однако на этапе настройки я рекомендую все-таки установить ее — она будет использоваться для тестирования сервера.

После установки пакета нужно создать базу данных **mysql**. Скорее всего, она уже создана, но, чтобы окончательно убедиться в этом, введите команду:

```
mysql_install_db
```

Если основная база данных существует, программа сообщит вам об этом. Теперь нужно установить пароль для пользователя root. По умолчанию пользо-

ватель root не имеет пароля. Я не буду долго объяснять, как важен правильно заданный пароль для безопасности системы, не говоря уже об его отсутствии.

База данных **mysql** является системной базой данных и содержит следующие таблицы:

1. таблицу db;
2. таблицу host;
3. таблицу user.

Сейчас нас интересует таблица user. Она содержит пароли всех пользователей, которые имеют право работать с сервером. На данном (начальном) этапе в базу данных внесен только один пользователь — root. Для изменения пароля запустите сервер командой:

```
safe_mysqld &
```

Эта команда запустит сервер в режиме демона и освободит консоль. Если все пакеты были установлены правильно, вы увидите сообщение:

```
mysql: ready for connections
```

Однако, может быть и другое сообщение, свидетельствующее об удачном запуске. Затем введите команду:

```
mysql -u root mysql
```

Данная команда запускает клиент MySQL. При этом используется имя пользователя root, даже если вы работаете под другой учетной записью. Последний параметр определяет базу данных — mysql.

Измените пароль суперпользователя с помощью команды:

```
UPDATE user SET Password=PASSWORD('new_password') WHERE
user='root';
```

Как вы заметили, это обычный SQL-запрос, обновляющий поле **Password** таблицы **user** для пользователя root. Теперь нужно, чтобы MySQL принял изменения. Для этого выполните еще один запрос SQL:

```
FLUSH PRIVILEGES;
```

Для принятия изменений можно также использовать программу **mysqladmin** с параметром reload. Вызвать программу можно так:

```
mysqladmin -p reload
```

Параметр -p вам обязательно нужно использовать, так как вы только что установили пароль для пользователя root. Выйти из клиента mysql вы можете, введя команду quit.

Установите права доступа к сценарию `/etc/rc.d/init.d/mysql`:

```
chmod +x /etc/rc.d/init.d/mysql
```

Теперь можете перезапустить сервер командой:

```
/etc/rc.d/init.d/mysql restart
```

Если вы забудете пароль, вы его уже не восстановите. Единственный выход из этого положения — удалить каталог `/var/lib/mysql/mysql` и создать базу mysql заново командой **mysql_install_db**.

Теперь вы уже не можете зарегистрироваться на сервере без пароля. Если вы введете команду **mysql -u root mysql,** то получите следующее сообщение:

```
ERROR: Access denied for user: 'root@localhost' (Using password: NO)
```

Для регистрации на сервере теперь нужно использовать команду **mysql -u root -p.** Параметр -p запросит пароль при регистрации.

Последнее, что вам осталось сделать — это добавить сервер MySQL в автозапуск. С этой целью перейдите в каталог /etc/rc.d/rc3.d/ и создайте символическую ссылку на файл /etc/rc.d/init.d/mysql:

```
ln -s S14mysql /etc/rc.d/init.d/mysql
```

Префикс **S14** определяет очередность запуска сервера **mysql.** В данном случае он запустится после сервисов **network** (S10) и **portmap** (S11). У вас эти значения могут быть другими.

В своей работе демон **mysqld** использует файл журнала /var/log/mysql.log. Именно в него заносятся все транзакции, а также все команды, которые ввел пользователь. После установки сервера нужно внести пользователей, которые имеют право работать с сервером баз данных. Введите следующий запрос:

```
GRANT ALL PRIVILEGES ON *.* TO admin@localhost IDENTIFIED BY
'password' WITH GRANT OPTION;
```

Введенный вами запрос создаст пользователя admin, который будет иметь право выполнять любые операции со всеми базами данных. Данный пользователь будет иметь право подключаться к серверу с компьютера localhost, используя пароль password.

Маска *.* определяет, к каким базам данных и таблицам имеет право подключаться тот или иной пользователь. Первая звездочка определяет базу, а вторая — таблицу. Если вам нужно, чтобы пользователь admin имел право подключаться с любого хоста, используйте знак процента вместо имени хоста. В этом случае запрос будет выглядеть так:

```
GRANT ALL PRIVILEGES ON *.* TO admin@"%" IDENTIFIED BY '
password' WITH GRANT OPTION;
```

Вместо всех полномочий вы можете определить, какие действия может выполнять с базой тот или иной пользователь. Если вы являетесь хостинг-провайдером и предоставляете доступ пользователю к его базе данных, то можете использовать следующий запрос:

```
GRANT CREATE,DROP,SELECT,INSERT,UPDATE,DELETE,INDEX ON user.* TO
user@% IDENTIFIED BY 'user_password';
```

Перед выполнением данного запроса необходимо создать базу данных **user.** Данный запрос позволяет пользователю user выполнять все операции с его базой данных.

Полный список полномочий представлен в табл. 16.1.

Если запрос GRANT у вас не работает, то вы можете внести пользователя непосредственно в таблицу user базы данных **mysql.** Структура таблицы **user** выглядит следующим образом:

```
Host User Password Select_priv Insert_priv Update_priv
Delete_priv Create_priv Drop_priv Reload_priv Shutdown_priv
Process_priv File_priv
```

Поля **Host, User, Password** -- это, соответственно, узел, из которого пользователь может получить доступ, имя пользователя, пароль пользователя.

Полномочия пользователей сервера MySQL *Таблица 16.1*

Полномочия	Описание
SELECT, INSERT, UPDATE, DELETE	Одноименные операции с данными: пользователь имеет право просматривать, добавлять, модифицировать, удалять данные в таблицах баз данных
INDEX	Пользователь имеет право производить операции с индексами таблиц
REFERENCES	Пользователь имеет право работать со ссылками в базах данных и таблицах
CREATE, DROP	Создание и удаление таблиц и баз данных
GRANT, ALTER	Операции с полномочиями
RELOAD, SHUTDOWN, PROCESS	Пользователь имеет право перезагружать, останавливать сервер и просматривать все процессы (подключения)

Все остальные поля задают полномочия для пользователя. Если выполнение какой-нибудь операции разрешено пользователю, соответствующее поле должно быть равным «Y». В противном случае установите значение «N».

Например, нам нужно создать пользователя admin, который должен иметь все полномочия. Это можно сделать с помощью такого запроса SQL:

```
INSERT INTO user (Host, User, Password, Select_priv,Insert_priv,Update_priv,Delete_priv,
Create_priv,Drop_priv,Reload_priv,Shutdown_priv,Process_priv, File_priv) VALUES
('localhost','admin',password('4td56ls12'),'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');
```

При вводе запроса обратите внимание на регистр названий полей. Сервер MySQL различает прописные и строчные буквы! С помощью вышеприведенного запроса был создан пользователь admin, который имеет право регистрироваться на сервере только из узла localhost. Если вам нужно разрешить регистрацию из любого узла сети, используйте знак процента, однако это не совсем корректно с точки зрения безопасности. Пользователь admin обладает всеми возможными привилегиями. Пароль пользователя — 4td561sl2.

Для создания обыкновенного пользователя используйте следующий запрос:

```
INSERT INTO
user(Host,User,Password,Select_priv,Insert_priv,Update_priv,Delete_priv)
VALUES('%','user',password('123456'),'Y','Y','Y','Y');
```

76.2. *Клиентская часть*

Удобной программой для просмотра структуры базы данных является mysqlshow. Введите следующую команду:

```
mysqlshow -p mysql
```

В ответ вы увидите список таблиц, которые находятся в базе данных mysql.

```
Database: mysql
+-----------+
I Tables  I
+-----------+
I  db     I
I host    I
|  user   |
+-----------+
```

Программа **mysqlshow** может вызываться с дополнительными параметрами, указанными в табл. 16.2.

Параметры программы mysqlshow *Таблица 16.2*

Параметр	Описание
--host=имя_хоста	Задает имя хоста, к которому вы хотите подключиться
--port=номер_порта	Определяет номер порта для сервера MySQL
--socket=сокет	Указывает сокет
--user=имя_пользователя	С помощью этого параметра можно указать нужное имя пользователя
-P	Запрашивает ввод пароля

Для самих же операций с данными используется программа **mysql**. Она и является клиентом сервера. В этой программе можно использовать те же опции, что и **mysqlshow.** Среди многочисленных параметров программа **mysql** имеет один очень важный параметр -s. Я рекомендую вам всегда его использовать. Этот параметр подавляет большинство ненужных сообщений, выводимых клиентом. На медленных линиях связи это должно повысить производительность. Да и наблюдать за всеми рамочками и ненужными сообщениями особо не хочется.

16.3. Связка Apache + PHP + MySQL

Настроить данную связку, которая очень полезна при Web-программировании, можно двумя способами. Первый из них -- это использовать программы, которые входят в состав дистрибутива и, как правило, устанавливаются из пакетов RPM. Второй способ заключается в загрузке последних версий Apache, MySQL и PHP и в самостоятельной их сборке из исходных текстов. Первый способ я могу порекомендовать начинающим пользователям, так как он более прост. Если же вы чувствуете уверенность в своих силах, приступайте сразу к чтению второго способа.

16.3.1. Первый способ: из пакетов RPM

Могу сразу обрадовать пользователей дистрибутивов Red Hat 7.2 и Mandrake 8.1 (или более поздних версий): вам не нужно настраивать первую часть связки — все настраивается во время установки системы. В состав дистрибутива Red Hat 7.2 входит сервер Apache 1.3.20-16, а при установке системы устанавливаются библиотека **gd** и интерпретатор **php** версии 4.0.6, а также модуль для сервера Apache. Поэтому вы можете сразу приступить к тестированию связки Apache + PHP (см. файл `test.php` ниже). Не забудьте установить сервер Apache, если он еще не установлен (перед выполнением данной команды перейдите в каталог, в котором находятся пакеты RPM):

```
rpm -Uh apache*
```

Затем следует настроить сервер. Настройка Apache подробно обсуждалась в гл. 12 этой книги. Не нужно настраивать сервер полностью: достаточно указать только директиву **ServerName** и попробовать запустить сервер.

Как правило, сервер должен успешно запуститься и функционировать. Для проверки его работы введите команду:

```
lynx http://localhost
```

Текстовый браузер lynx должен отобразить стартовую страницу Apache. После успешного запуска сервера остановите его командой:

```
/etc/init.d/httpd stop
```

Проверьте наличие библиотеки gd -- она необходима для работы с графикой в PHP:

```
rpm -qa I grep gd
```

Если библиотека gd не установлена, установите ее командой, предварительно указав ту версию библиотеки, которая у вас имеется (я использую версию 1.8.4):

```
rpm -Uhv gd-1.8.4-4.i386.rpm
```

Установите пакет php, если вы его еще не установили:

```
rpm -Uh php*
```

После этого установите модуль Apache, обеспечивающий поддержку PHP:

```
rpm -Uh mod_php*
```

Данный модуль должен входить в состав дистрибутива и обычно находится на первом инсталляционном диске. Затем в файле `httpd.conf` найдите и раскомментируйте следующую строчку. После этого файлы с расширением .php будут правильно обрабатываться сервером:

```
AddType application/x-httpd-php4 .php
```

Теперь можно проверять правильность настройки двух компонент связки: Apache и PHP. Создайте тестовый файл `test.php` с таким содержимым:

```
<?
phpinfo();
?>
```

Этот файл сохраните в каталоге **DocumentRoot** сервера Apache. Обычно это каталог `/var/www/html`. После этого запустите любой браузер и введите следующий адрес:

```
http://localhost/test.php
```

При этом на экране вы должны увидеть различную информацию о PHP, сервере Apache и других компонентах и библиотеках, например, о библиотеке gd (см. рис. 16.1).

Функция **phpinfo()** очень информативна: внимательно изучив информацию, которую она предоставляет, вы много узнаете о своем Web-сервере. К тому же эту информацию можно использовать при Web-программировании.

Теперь немного настроим PHP. С помощью функции phpinfo() узнайте, где расположен инициализационный файл PHP. Обычно он называется `php.ini` и находится в каталоге /etc. Откройте этот файл в любом текстовом редакторе и раскомментируйте следующую строку, убедившись, что в вашей системе есть файл `mysql.so` (он устанавливается при установке MySQL):

```
extension=mysql.so
```

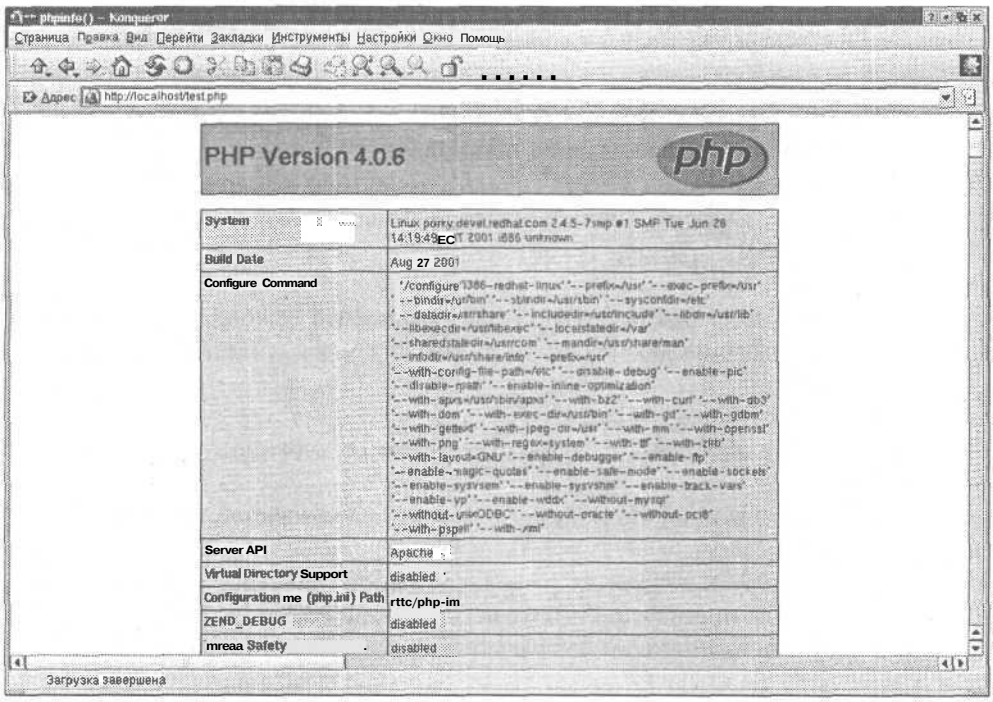


Рис. /6. *1. Функция phpinfo()*

После этого перейдите в секцию MySQL файла `php.ini` и установите параметры сервера MySQL по умолчанию:

```
mysql.default_port =
mysql.default_socket =
mysql.default_host =localhost
mysql.default_user =
mysql.default_password =
```

Эти параметры будут использоваться при установлении соединения с сервером, если в функциях PHP они не будут явно указаны. Никогда не указывайте пользователя root (а тем более его пароль) в качестве пользователя по умолчанию!

Теперь можно приступить к установке и настройке сервера MySQL. В состав современных дистрибутивов, например, Red Hat 7.2, входит сервер MySQL версии 3.23. Вы можете использовать эту версию, но я рекомендую установить более старую версию — 3.20. Если вы установили версию 3.23, установите также пакет **mod_auth_mysql**. Данный пакет обеспечивает базовую аутентификацию для сервера Apache с использованием таблиц MySQL. Напомню, что сам сервер MySQL устанавливается командой:

```
rpm -Uh mysql*
```

Запустите сервер. Можно также добавить вызов сервера MySQL в сценарии автозагрузки. При добавлении сервера MySQL в сценарии загрузки (`/etc/rc.d/`)

обратите внимание на то, что сервер MySQL должен быть запущен ПЕРЕД сервером Apache. Представьте такую ситуацию: сначала запускается сервер Apache и сразу же получает запрос от клиента к базе данных MySQL, а сервер баз данных еще не запущен. Даже если запускать сервер MySQL сразу после Apache, то все равно ему понадобится еще некоторое дополнительное время для инициализации. Не забудьте установить пароль для пользователя root сервера MySQL:

```
mysql -u root -e "update user set
password=password('new_password') where user='root'" mysql
```

Перезагрузите сервер:

```
mysqladmin reload
```

Теперь можно проверить работу всей связки Apache + PHP + MySQL. С этой целью создайте небольшой тестовый файл `mysql_test.php` в каталоге `/var/www/html`.

```
<?
// Используется имя пользователя root и пароль passwd
if(!mysql_connect("localhost","root","password"))
{
echo "Не могу соединиться с сервером\n";
echo mysql_error();
exit;
}
echo "Работает!"
?>
```

Как вы уже успели догадаться, если в окне браузера вы увидите слово «Работает!», значит, вы все сделали правильно.

16.3.2. Второй способ: из исходных текстов

Как я уже говорил, это более сложный способ, но у него есть свои преимущества. Во-первых, у вас появится возможность использовать самые последние версии серверов Apache, MySQL и интерпретатора PHP, которых нет в составе даже самого нового дистрибутива Linux. Во-вторых, вы сами сможете контролировать процесс сборки и включать поддержку необходимых вам функций, исключив такую ситуацию, когда, например, разработчики пакетов RPM при сборке интерпретатора PHP забыли включить поддержку сервера MySQL. Мне попадался такой дистрибутив php: функции **mysql_connect()** в нем просто не было.

Итак приступим к настройке. Но перед этим скачайте из Интернет последние версии Apache, MySQL и PHP. Предварительно удалите из системы старые версии, если такие были установлены. После загрузки распакуйте исходные тексты в каталог `/src`. Далее, сначала установите сервер MySQL. С этой целью перейдите в каталог с исходными текстами MySQL и введите следующие команды (первая команда включает поддержку по умолчанию кодировки koi8-r):

```
./configure -with-charset=koi8_ru
make
make install
```

Затем аналогично установите Apache, перейдя в соответствующий каталог:

```
./configure
make
make install
```

Для получения информации обо всех опциях команды **configure** введите команду **configure —help**. После этого распакуйте PHP и перейдите в каталог с исходными текстами PHP. Введите команды:

```
./configure —with-mysql —with-apache=../apache_1.3.20 —with-mod_charset
make
make install
```

Первая команда конфигурирует интерпретатор PHP для работы с сервером баз данных MySQL и Web-сервером Apache. Естественно, вы должны правильно указать путь к исходным текстам Apache с помощью параметра **-with-apache**.

Затем вернитесь в каталог, содержащий исходные тексты Apache, и введите команду:

```
./configure —activate-module=src/modules/php4/libphp4.a
```

Перед этим нужно убедиться в существовании файла `libphp4.a`. Этот файл должен существовать, если php собрался нормально. Если конфигуратор **configure** успешно завершил свою работу, введите команды:

```
make
make install
```

Этими командами вы собираете сервер Apache с подключенным модулем **libphp**. Проверить подключился ли нужный модуль вы можете после установки сервера (выполнения команды make install) с помощью команды:

```
httpd -1
```

В списке модулей должен быть модуль Iibphp4.c, а также модуль **mod_charset.c** — его вы подключили при первой сборке. После этого можно отредактировать файл `/etc/php.ini` и установить пароль для пользователя root сервера MySQL (не путайте пользователя root всей системы с пользователем root сервера MySQL!). Обе операции уже были описаны в п. 16.3.1. Теперь только остается добавить запуск серверов в сценарии автозагрузки системы. Напомню, что сервер MySQL должен запускаться до сервера Apache.

17

Практические примеры. Обратный звонок

Возможно, материал этой главы будет пересекаться с уже имеющимся в этой книге, но при написании главы я ставил цель объяснить все «с нуля». Практически любой читатель сможет настроить шлюз и сервер входящих звонков, не читая предыдущих глав. Конечно, этот читатель должен обладать некоторыми навыками работы в Linux.

17.1. Настройка шлюза

Сначала определим функции, которые должен выполнять шлюз:

1. Поддержка связи с провайдером.
2. Маршрутизация IP-пакетов между локальной сетью и сетью Интернет для выхода пользователей локальной сети в Интернет.
3. Обеспечение IP-сервиса.
4. Защита локальной сети от несанкционированного доступа из Интернет.

Конфигурирование шлюза в операционной системе Linux состоит из следующих этапов:

1. Настройка ядра.
2. Настройка сети.
3. Конфигурирование IpChains.
4. Настройка DNS.
5. Настройка Squid.

Для определенности будет использоваться два сетевых интерфейса – eth0, идущий к провайдеру, и eth1 — во внутренней сети. Пусть интерфейсу ethO назначен IP-адрес 111.111.111.111, а ethl — 192.168.1.1

17.1.1. Настройка ядра

Скорее всего, вам придется перекомпилировать ядро. При этом должны быть активизированы следующие опции:

```
Networking support (CONFIG_NET) [y]
TCP/IP networking (CONFIG_INET) [y]
IP forwarding/gatewaying (CONFIG_IP_FORWARD) [y]
```

```
IP multicasting (CONFIG_IP_MULTICAST) [y]
IP firewalling (CONFIG_IP_FIREWALL) [y]
IP accounting (CONFIG_IP_ACCT) [y]
```

Можно также поэкспериментировать с набором опций **Advanced Router,** если данные функции есть в вашем ядре. Более подробно о процессе компилирования ядра вы можете прочитать в следующей главе.

17.1.2. Настройка сети

После перекомпилирования ядра нужно включить **IP-forwarding.** Сделайте это при помощи следующей команды:

```
# echo "1" > /proc/net/ip_forward
```

Настройку сетевых карт произведите с помощью программы **netconf.** О том, как это сделать, было рассказано в гл. 8.

17.1.3. Конфигурирование IpChains

Теперь приступим к настройке **IPChains.** Создайте цепочку, через которую пойдет весь трафик от провайдера:

```
ipchains -N prov
ipchains -A input -i ethO -j prov
Запретите ip-spoofing:
ipchains -A prov -s 192.168.1.1/16 -1 -j DENY
ipchains -A prov -s 127.0.0.1/8 -1 -j DENY
Запретите Telnet снаружи:
ipchains -A prov -p tcp -destination-port 23 -j REJECT
```

Если вы не хотите, чтобы **samba** «светилась» наружу, запретите порты 137-139:

```
ipchains -A prov -p tcp –destination-port 137 -j REJECT
ipchains -A prov -p udp -destination-port 137 -j REJECT
```

То же самое проделайте для портов 138 и 129.

О настройке **samba** вы можете прочитать в Samba-HOWTO.

Создайте цепочку для подсчета трафика:

```
ipchains -N trafin
ipchains -I input -i ethO -s ! 123.123.123.0/24 -p all -j
trafin
ipchains -A trafin -d 123.123.123.123
```

Для того, чтобы ваши правила были постоянными (при перезагрузке машины правила IpChains теряются), используйте скрипты **ipchains-save и ipchains-restore.** Настройте свои правила, а затем выполните команду:

```
# ipchains-save > /etc/ipchains.rules
```

Далее создайте скрипт, подобный тому, что приведен в листинге 17.1.

Листинг 17.1. Скрипт управления пакетной фильтрацией

```
#! /bin/sh
# Скрипт управления пакетной фильтрацией.
# Если правил нет, то ничего не делать.
```

```
#! /bin/sh
# Скрипт управления пакетной фильтрацией.
# Если правил нет, то ничего не делать.
[-f /etc/ipchains.rules ] | | exit 0
case "$1" in
  start)
    echo -n "Включение пакетной фильтрации:"
    /sbin/ipchains-restore < /etc/ipchains.rules | | exit 1
    echo 1 > /proc/sys/net/ipv4/ip_forward
    echo " . " ;;
  stop)
    echo -n "Отключение пакетной фильтрации:"
    echo 0 > /proc/sys/net/ipv4/ip_forward
    /sbin/ipchains -X
    /sbin/ipchains -F
    /sbin/ipchains -P input ACCEPT
    /sbin/ipchains -P output ACCEPT
    /sbin/ipchains -P forward ACCEPT
    echo "." ;;
  *)
    echo "Использование: /etc/init.d/packetfilter {start|stop}"
    exit 1 ;;
esac
exit 0
```

Этот скрипт добавьте в сценарии загрузки системы.

17.1.4. Настройка DNS

Напомню, что основной задачей сервера доменных имен (Domain Name System) является преобразование мнемонических имен машин в IP-адреса и обратно. Обычно сервер DNS устанавливается на шлюзе, который используется для выхода в Интернет.

Прежде чем приступить к настройке сервера, нужно определить, запущен ли он:

```
# ps -ax | grep named
```

Если он запущен, его нужно остановить (с помощью **ndc),** а если он вообще не установлен, то придется установить пакет **bind.** Для работы сервера должен быть активизирован сервис **network.**

Теперь приступим к непосредственной настройке сервера. Основная информация о параметрах сервера содержится в файле `/etc/named.conf` (см. листинг 17.2).

Листинг 17.2. Файл named.conf

```
logging {
   category cname {null; } ;
   };
options {
   directory "/var/named";
   };
```

```
zone "." {
     type hint;
     file "named.ca";
     };
zone "dhsilabs.com" {
     type master;
     file "dhsilabs.com";
     notify no;
     };
zone "0.0.127.in-addr.arpa" {
     type master;
     file "named.local";
     };
zone "1.168.192.in-addr.arpa" {
     type master;
     file "192.168.1";
     notify yes;
     };
```

Основной каталог сервера — /var/named. В нем сервер будет искать файлы dhsilabs.com, named.local, 192.168.1, named.ca (см. листинги 17.3, 17.4, 17.5). Обслуживаемая нашим сервером зона (домен) — dhsilabs.com (см. листинг 17.3). Файл named.ca -- корневой кэш — содержит информацию о корневых серверах DNS. Позже займемся его обновлением.

Листинг 17.3. Файл dhsilabs.com (для преобразования имен в IP-адреса)

```
@       IN      SOA den.dhsilabs com. hostmaster.dhsilabs.com. (
                  93011120          ; серийный номер
                  10800             ; обновление каждые 3 часа
                  3600              ; повтор каждый час
                  3600000           ; хранить информацию 1000 часов
                  86400   )         ; TTL записи — 24 часа
        IN      NS  den.dhsilabs.com.
        IN      A   192.168.1.1
        IN      MX 150  den.dhsilabs.com.
den     IN      A   192.168.1.1
        IN      HINFO   INTEL CELERON (LINUX)
        IN      MX 100  den
        IN      MX 150  evg.dhsilabs.com.
ns      IN      CNAME  den.dhsilabs.com.
www     IN      CNAME  den.dhsilabs.com.
ftp     IN      CNAME  den.dhsilabs.com.
mail    IN      CNAME  den.dhsilabs.com.
evg     IN      A   192.168.1.2
        IN      MX 100  den.dhsilabs.com.
localhost IN    A   127.0.0.1
```

где: NS ..... обозначает name server;
A ..... IP-адрес;

MX.....почтовик <приоритет>. Чем ниже значение, тем выше приоритет;

HINFO.....сведения об аппаратном обеспечении (заполнять не рекомендую);

TXT.....прочие сведения;

CNAME.....каноническое имя, т.е. если вы в окне браузера введете http://www.dhsilabs.com, то обращение будет произведено к den.dhsilabs.com. Обратите внимание на точку в конце:

```
@   IN  SOA den.dhsilabs.com. hostmaster.dhsilabs.com.
```

(Если точка не указана, то к имени будет добавлено имя домена (т.е. dhsilabs.com)).

Листинг 17.4. Файл named.local

```
@     IN   SOA        dhsilabs.com. root.dhsilabs.com. (
           199609203          ; серийный номер
           28800              ; обновление каждые 8 часов
           7200               ; повтор каждые 2 часа
           604800             ; хранить информацию 168 часов (7 дней)
           86400)             ; TTL записи -- 24 часа
           NS         dhsilabs.com.
1          PTR        localhost.
```

Листинг 17.5. Файл 192.168.1 или файл обратного соответствия

```
@     IN   SOA        den.dhsilabs.com. hostmaster.dhsilabs.com. (
           93011120           ; серийный номер
           10800              ; обновление каждые 3 часа
           3600               ; повтор каждый час
           3600000            ; хранить информацию 1000 часов
           86400 )            ; TTL записи -- 24 часа
@     IN   NS         den.dhsilabs.com
1     IN   PTR        den.dhsilabs.com
2.1.168.192 IN PTR evg.dhsilabs.com
```

Запись **PTR** используется для преобразования IP-адреса в имя. Если указан не весь IP:

```
1 . IN  PTR den.dhsilabs.com
```

то к нему будет добавлен адрес подсети 1.168.192. Обратите внимание: IP-адреса указываются в обратном порядке!

17.1.5. Настройка Squid

Установите пакет **squid.** Осталось настроить и запустить его. Для этого нужно отредактировать файл конфигурации /etc/squid/squid.conf. Сначала укажите адрес прокси провайдера:

```
cach_peer proxy.your_isp.com
```

Задайте объем ОЗУ, который будет использовать прокси-сервером:

```
cache_mem
```

В том случае, если вы планируете использовать этот компьютер еще и для других целей, кроме как в качестве прокси-сервера, то не рекомендуется устанавливать здесь более трети физического объема ОЗУ.

Далее укажите, где будет располагаться кэш (первое число — это количество Мб для кэша):

```
cache_dir /usr/local/squid 2048 16 256
```

Затем укажите хосты, из которых разрешен доступ к прокси-серверу:

```
acl allowed_hosts src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
```

После этого пропишите пользователей, которым разрешено пользоваться **squid** (в приведенном примере это den, admin, developer):

```
ident_lookup on
acl allowed_users user den admin developer
http_access allow allowed_users
http_access deny all
```

Тэги **maxium_object_size** и **maxium_object** устанавливают ограничения на размер передаваемых объектов.

В заключении хочу дать один хороший совет: из соображений безопасности отредактируйте свои /etc/services и /etc/inetd.conf и отключите неиспользуемые сервисы -- это уменьшит вероятность взлома вашей системы. Вот, в общем-то, и все.

77.2. Настройка Dial-In сервера

17.2.1. Установка программного обеспечения

В качестве операционной системы, естественно, будем использовать ОС Linux. Метод настройки, рассмотренный в этой главе, подойдет для любого дистрибутива. Также вам потребуются пакет **ppp** версии 2.3.x (желательно самая новая версия) и пакет **mgetty-1.1.x.** Пакет **mgetty** должен быть собран с опцией -DAUTO_PPP. Если это не так, то его нужно пересобрать.

Я использую **ppp-2.4.0-3mdk.i586.rpm** и **mgetty-1.1.22-2mdk.i586.rpm**, ОС Linux Mandrake 7.2. Если вы используете RedHat/Mandrake, установить **ppp** и **mgetty** можно с помощью команд:

```
# mount -t iso9660 /dev/hdd /mnty/cdrom
#cd /mnt/cdrom/Mandrake/RPMS
#rpm -Uvh mgetty*
#rpm -Uvh ppp*
```

Некоторые замечания:

1. CDROM является устройством /dev/hdd (или Secondary Slave).
2. Используется Linux Mandrake. При использовании Red Hat пакеты находятся В /mnt/cdrom/RedHat/RPMS.
3. Не используется **supermount**. Если у вас **supermount** активен, первую команду вводить не нужно.
4. Третья и четвертая команды устанавливают все семейство **mgetty** и **ppp.** При использовании такого подхода устанавливаются все файлы — и никакой заботы! Но при этом вы можете установить и только то, что вам нужно.

17.2.2. Настройка mgetty

При корректной сборке или установке пакетов **mgetty** и **ppp** у вас должны быть следующие файлы:

Каталог /etc/mgetty+sendfax:

```
dialin.config
login.config
mgetty.config
```

Каталог /etc/ppp:

```
auth-up
auth-down
chap-secrets
ip-up
ip-down
options
pap-secrets
```

Если их нет, вам нужно самостоятельно найти, где они находятся. При самосборке смотрите, что и куда проинсталлировалось. В крайнем случае, необходимые файлы придется создать вручную.

Файл /etc/mgetty+sendfax/dialin.config **обычно пустой — в нем все закомментировано. Файл** /etc/mgetty+sendfax/login.config **должен содержать строчку:**

```
/AutoPPP/-  a_ppp  /etc/ppp/ppplogin
```

Убедитесь, что эта строчка не закомментирована. Если вы хотите, чтобы имена пользователей записывались в журналы (log-файлы), отредактируйте эту строку следующим образом:

```
/AutoPPP/-        /etc/ppp/ppplogin
```

Затем создайте файл /etc/ppp/ppplogin (см. листинг 17.6).

Листинг 17.6. Файл /etc/ppp/ppplogin

```
mesg n
tty -echo
/usr/sbin/pppd silent auth -chap +pap login
```

В некоторых версиях **ppp** вместо **-chap** нужно писать **refuse-chap,** а вместо **+pap** писать **require-pap**. Продолжая настройку, сделайте /etc/ppp/ppplogin исполняемым:

```
# chmod +x /etc/ppp/ppplogin
```

В приведенном примере используется PAP-аутентификация **с использованием пароля из файла** /etc/passwd **(см. ниже). Файл** /etc/mgetty+sendfax /mgetty.config **должен быть примерно такой, как в листинге 17.7.**

Листинг 17.7. Файл mgetty.config

```
# For US Robotics Sportster 28.8 with speaker off
port ttyS0
speed 28800
data-only y
debug 3
```

```
init-chat "" ATZ OK AT&F1M0E1Q0S0=0 OK
answer-chat "" ATA CONNECT \c \r
# For Practical Peripheral 14.4 with fax disabled and prolonged
# carrier wait time (90 sec)
port ttyS1
speed 14400
data-only y
debug 3
init-chat "" ATZ OK AT&F1M0E1Q0S0=0S7=90+FCLASS=0 OK
answer-chat "" ATA CONNECT \c \r
# For USRobotics V.Everything
port ttyS2
speed 57600
data-only y
debug 3
init-chat "" AT OK ATS7=50S0=1+S62=3+S64=2S39=10 OK
```

Для ZyXEL U336E можно использовать такие параметры:

```
init-chat "" ATZ OK AT&F1M0E1Q0S0=0S OK
answer-chat "" ATA CONNECT \c \r
```

В нем определены параметры для трех модемов: US Robotics Sportster 28.8, Practical Peripheral 14.4, USRobotics V.Everything. Для **ZyXEL** U336E можно использовать такие параметры:

```
init-chat "" ATZ OK AT&F1M0E1Q0S0=0S OK
answer-chat "" ATA CONNECT \c \r
```

Данные параметры вы можете узнать из документации по вашему модему. Очень рекомендую прочитать ее перед установкой сервера входящих звонков.

Теперь нужно изменить файл /etc/inittab, как это показано в листинге 17.8.

Листинг 17.8. ***Фрагмент файла inittab***

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
#эти строки нужно добавить
S0:2345:respawn:/sbin/mgetty -x 3 ttySO
S1:2345:respawn:/sbin/mgetty -x 3 ttyS1
S2:2345:respawn:/sbin/mgetty /dev/ttyS2
```

SO, S1, S2 -- это просто идентификаторы. Вы можете использовать вместо них любое имя. Нужно только назначить отдельное имя для каждого порта. Имена S0...S2 я использовал для наглядности.

Теперь запустите **mgetty** (перед выполнением этой команды следует включить модемы):

```
# init q
```

Если при выполнении этой команды модемы не подключены или выключены, в файле /var/log/mesages вы получите сообщение об этом. Если на модеме загорелась лампочка TR, то все настройки выполнены правильно и **mgetty** «подхватил» модем.

17.2.3. Настройка ppp

Обычно для каждого порта в каталоге /etc/ppp создается файл options . ttySx, где x — номер порта (см. листинг 17.9).

Листинг 17.9. Файл options

```
lock
login
auth
netmask 255.255.255.0
modem
crtscts
refuse-chap
require-pap
mtu 576
mru 576
proxyarp
myhost:ppp01
ms-dns CCC.CCC.CCC.CCC
```

Общие настройки для всех портов можно вынести в файл /etc/ppp/options. Имя myhost замените на реальное имя вашего сервера входящих звонков.

ppp01 — это произвольно выбранное имя виртуального узла абонента. Вы можете использовать другие имена, например, igor, denis и тому подобное.

Имена узлов должны быть уникальными, т.е. если вы используете pppO в options . ttySO, то в options . ttyS1 нужно использовать ppp01 и так далее.

Опция ms-dns определяет сервер DNS для клиентов Microsoft. Укажите здесь IP-адрес сервера DNS вашей сети.

Используйте опцию proxyarp, так как IP-адреса будут назначаться внутри broadcast ваших сетевых карт локальной сети. При этом демон **pppd** будет делать вид, что виртуальный хост находится внутри вашего сегмента ethernet. Небольшая деталь: вместо опции refuse-chap можно писать -chap, а вместо require-pap писать +pap.

Теперь отредактируйте файл /etc/ppp/pap-secrets (см. листинг 17.10).

Листинг 17.10. Файл /etc/ppp/pap-secrets

```
# Secrets for authentication using PAP
# client    server  secret  IP addresses
*           *       ""      192.168.0.11
*           *       ""      192.168.0.12
*           *       ""      192.168.0.13
```

В приведенном примере используются три модема для входящих звонков, поэтому нужно сделать три записи. Пароли пользователей находятся в

файле /etc/passwd (или /etc/shadow). Соответственно внесите изменения в свой /etc/hosts (см. листинг 17.11).

Листинг 17.11. Файл /etc/hosts

```
192.168.0.11 ppp01 ppp01.mydomain.com
192.168.0.12 ppp02 ppp02.mydomain.com
192.168.0.13 ppp03 ppp03.mydomain.com
```

Имя mydomain.com замените на реальное имя домена. По большому счету эти записи не мешало бы также внести и в локальную зону DNS. Теперь осталось установить нужные права доступа для /usr/sbin/pppd:

```
# chmod u+s /usr/sbin/pppd
```

17.2.4. Включение IP Forwarding

Разрешение пересылки IP устанавливается в файле /etc/sysconfig /network примерно так: FORWARD_IPV4=yes. При этом ваше ядро должно быть скомпилировано для поддержки IP_FORWARD.

Для включения IP Forwarding введите команду:

```
# echo "1" > /proc/net/ip_forward
```

В некоторых дистрибутивах IP Forwarding включается несколько иначе. Если в вашем дистрибутиве есть программа netconf, используйте ее для включения IP Forwarding, если ее нет, изучите документацию по вашему дистрибутиву.

Теперь вы готовы к работе!

17.2.5. Второй вариант настройки

Этот вариант может оказаться даже более простым, чем первый. Настройки файлов /etc/options и /etc/options.ttySx остаются прежними, но строку **myhost:pp01** нужно заменить на строку вида:

```
Server_IP:Client_IP
```

например,

```
192.168.0.1:192.168.0.11
```

Теперь нужно изменить содержание файла /etc/ppp/pap-secrets (листинг 17.12).

Листинг 17.12. Фрагмент файла /etc/ppp/pap-secrets

```
#
user1   сервер.домен  ""    *
user2   сервер.домен  ""    *
#
```

где: **userl** - это имя пользователя, зарегистрированного в системе;
сервер.домен - это имя сервера входящих звонков;
"" — пароли брать из /etc/passwd (/etc/shadow);
* - абонент может аутентифицироваться с любого IP.

При желании можно назначить другой пароль. В этом случае, если данный сервер используется также и в качестве почтовика, то для входящих звонков и сервиса POP будут использоваться различные пароли.

Внимание!

Пароли в /etc/ppp/pap-secrets содержатся в открытом виде и не кодируются с помощью алгоритма MD5 (или DES), как в файле /etc/shadow (/etc/passwd). Файл /etc/hosts править не нужно.

Вот, собственно, и все.

17.2.6. Если что-то не работает...

Лучший совет в этом случае — смотрите файл `/var/log/messages`. В нем много всего интересного. Если у вас появляются сообщения вида:

```
modprobe: can't locate module char-major-24
```

то надо прописать в файле `/etc/conf.modules` **следующие строки:**

```
alias ppp-compress-21 bsd_comp
alias ppp-compress-24 ppp_deflate
alias ppp-compress-26 ppp_deflate
```

17.2.7. Настройка Windows-клиентов

Рассмотрим наиболее распространенную ситуацию, когда для подключения к нашему серверу используется обыкновенное удаленное соединение. IP-адрес клиента и адрес сервера DNS назначается провайдером, то есть нашим сервером. При этом в свойствах соединения нужно указать следующие данные (см. рис. 17.1):

- Тип сервера удаленного доступа: PPP.
- Дополнительные параметры: только «Программное сжатие данных».
- Допустимые протоколы: только «TCP/IP».

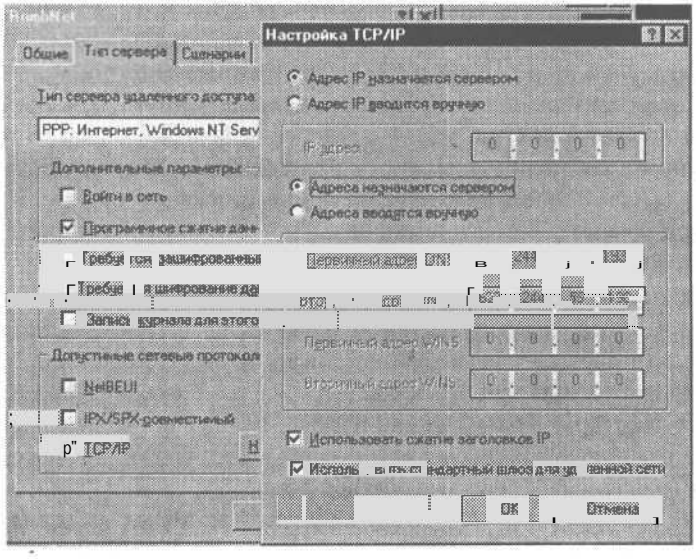


Рис. 17.1. *Параметры PPP*

17.2.8. Дополнительная литература

Из дополнительной литературы рекомендую эту книгу, если вы ее еще не прочитали, а также руководства по протоколу PPP, системе доменных имен DNS, и руководство NET3-HOWTO. Все эти руководства доступны на моей домашней страничке — http://dkws.narod.ru, а также на прилагаемом компакт-диске.

17.3. Обратный звонок

17.3.1. Что такое callback?

Первоначально обратный звонок был предназначен для снижения стоимости международных телефонных переговоров. Стоимость разговора определяется так: отсчет начинается с момента, когда вызываемый абонент поднял трубку или после пятого гудка, если абонент не отвечает, интервал тарификации -- 1 минута, то есть каждая неполная минута будет оплачиваться как полная. Стоимость самого разговора зависит от страны, из которой мы звоним. Например, в Украине 1 минута связи с США вам обойдется в 2,5...3,3 доллара США в зависимости от типа линии, которую вы используете: обыкновенную или Utel. Звонок из США в Украину вам обойдется 1...2 доллара. На этом и основана идея **callback** (callback — обратный звонок).

Рассмотрим небольшой пример: звонок из Украины в США:

1. Украинский абонент набирает выделенный ему номер в США и после первого вызова кладет трубку. Этим звонком он активирует специальное оборудование системы **callback.** Естественно, этот звонок не оплачивается, потому что оплата начинается с момента поднятия трубки вызываемым абонентом или после пятого вызова. Украинский абонент должен быть подключен к линии с тональным набором.
2. Через 5...20 секунд система **callback** перезванивает абоненту Украины и приглашает его набрать номер, по которому он хочет позвонить. Система **callback** набирает этот номер. При этом соединение устанавливается из США, а не из Украины, что в конечном итоге ведет к снижению стоимости всего звонка. При этом интервал тарификации не одна минута, как при звонке из Украины, а всего 6 секунд. ,

В случае с доступом в какую-нибудь сеть система **callback** работает почти аналогично. Только сейчас нашей целью является не снижение стоимости (хотя это тоже не исключено), а повышение безопасности сети и дополнительная ее защита от несанкционированного доступа.

Работает **callback** примерно так: клиент сети как обычно устанавливает соединение с сервером и проходит аутентификацию. Если аутентификация прошла успешно, сервер обрывает соединение (кладет трубку). Естественно, если аутентификация не прошла, сервер кладет трубку и не предпринимает никаких дальнейших действий. Кроме логина пароля, клиент также передает некоторое «волшебное» слово. Если сервер получил это слово, то через определенное время (обычно 25...30 секунд) сервер обратного звонка перезвонит по заранее запрограммированному номеру клиента и установит соедине-

ние. После этого можно работать в сети как обычно. Как видите, если раньше для доступа в сеть достаточно было знать имя пользователя и пароль, то сейчас нужно, чтобы компьютер, который пытается войти в сеть, был подключен к телефонной линии с **заранее запрограммированным номером.**

Например, ваш сосед купил пакет неограниченного доступа к Интернет, а вы каким-то образом узнали его имя пользователя и пароль. При обычном соединении (не **callback)** вы можете работать в Интернет так же, как и сосед, но, естественно, в разное время. При использовании технологии **callback** подобное уже не пройдет, потому что система **callback** перезвонит не к вам домой, а к вашему соседу. Таким образом, система **callback** является дополнительным барьером для нежелательных гостей нашей сети.

В этой главе будут рассмотрены два варианта настройки системы **callback.** Второй вариант более простой и удобный для клиентов: для установления соединения не нужно никаких скриптов, но этот способ почему-то не всегда корректно работает. Первый требует специальной настройки клиентов, но работает в большинстве случаев. Я рекомендую вам поступить следующим образом: ознакомиться с обоими способами, а начать настройку сначала со второго способа и, если он у вас не будет работать, перейти к настройке первого.

17.3.2. Настройка сервера. Способ 1

Приступим к настройке по первому способу. Как я уже отмечал, этот способ предполагает дополнительную настройку клиентов, что не очень удобно. Например, если вы являетесь администратором компании-провайдера Интернет, и у вас несколько сотен пользователей, настраивать все машины, даже если эта работа оплачивается, не очень хочется. Можно, конечно, распечатать подробные инструкции для клиентов, но ведь существуют и такие пользователи, которые и обычный доступ настроить не в состоянии.

Я предполагаю, что программы **mgetty и pppd** у вас уже настроены и сам сервер удаленного доступа нормально функционирует. Напомню, что установка и настройка программы **mgetty** была подробно рассмотрена в предыдущем п. 17.2.

При сборке программы **mgetty** нужно включить опцию автоматического распознания **pap-авторизации.** Если вы этого не сделали, сейчас самое время это сделать. Если вы установили **mgetty** из пакетов RPM, то можете пропустить инструкции по включению этой функции. Хотя если у вас все-таки функция DAUTO_PPP работать не будет, даже при установке из пакетов RPM, вам придется пересобрать программу самостоятельно. Исходные тексты программы **mgetty** доступны по адресу **http://alpha.greenie.net/mgetty/.** После распаковки копируем файл `policy.h-Dist` в файл `policy.h`. В этом файле нужно сделать определенные изменения, которые подходят для вашей системы. В большинстве случаев нужно изменить расположение каталогов или же вообще ничего не изменять.

Теперь нужно включить автоматическое распознавание pap-авторизации. Для этого в файле Makefile найдите строку:

```
CFLAGS=-O2 -Wall -pipe
```

и измените ее на:

```
CFLAGS=-02 -Wall -pipe -DOFIDO -DOAUTO_PPP
```

В этом же файле можно изменить установочные каталоги, но я не рекомендую этого делать. Оставьте все как есть: **SBINDIR=/sbin, BINDIR=/bin**

После этого выполните две команды, которые откомпилируют и установят **mgetty**:

```
make all
make install
```

Теперь перейдите в каталог **callback** каталога, который содержит исходные тексты **mgetty.** Скопируйте программу **callback** в каталог /usr/sbin, а файл callback.config — в каталог /etc/mgetty+sendfax. Отредактируем файл /etc/mgetty+sendfax/login.conf. Раньше (при условии, что вы настраивали сервер удаленного доступа) он у вас содержал примерно такую строку:

```
/AutoPPP/- a_ppp /etc/ppp/ppplogin
```

Сейчас эту строку нужно изменить на следующую:

```
/AutoPPP/ - a_ppp /usr/sbin/pppd noauth -chap +pap -detach
```

При обнаружении входящего звонка **mgetty** передаст управление демону **pppd**.

Добавим в файл mgetty.config описание порта, к которому подключен модем (см. листинг 17.13).

Листинг 17.13. Файл mgetty.config

```
port ttySO
# режим для приема только данных
# (отключает факсимильные возможности)
dataonly y
# Строка инициализации модема
init-chat "" AT OK  # Здесь можно задать строку инициализации модема
# Реинициализация модема
force-init-chat "" AT OK
```

Теперь нужно отредактировать файл /etc/ppp/options.ttyS0 (см. листинг 17.14). В этом файле вы укажите параметры порта ttySO.

Листинг 17.14. Файл /etc/ppp/options.ttyS0

```
# Максимальная скорость передачи данных
38400
# Аппаратный контроль передачи
crtscts
# Блокировка устройства ttySO
lock
# Режим коммутируемой линии
modem
# IP-адреса сервера и клиента соответственно
192.168.1.1:192.168.1.207
# IP-адрес сервера DNS. Этот параметр обязателен для Windows-клиентов
ms-dns 192.168.1.1
```

```
# Интервал отправления LCP-пакетов
lcp-echo-interval 20
# Количество не принятых LCP-пакетов
lcp-echo-failure 6
# Размеры пакетов
mtu 576
mru 576
```

Интервал отправления LCP-пакетов и количество не принятых LCP-пакетов нужны серверу для определения функционирования клиента, то есть с помощью этих параметров **pppd** определяет «жив» или нет клиент. Если на протяжении последних 120 секунд (20*6) клиент не прислал подтверждения, то сервер разорвет связь.

Запустите **mgetty,** как обычно — через файл `/etc/inittab`:

```
S0:35:respawn:/sbin/mgetty -D -n 1 /dev/ttySO 38400
```

Напомню, что SO — это просто идентификатор, и вы можете использовать любое другое значение. 3,5 — это уровни запуска. Параметр -D программы **mgetty** указывает ей отключить все факсимильные возможности модема, а параметр -п 1 заставляет **mgetty** подымать трубку после первого звонка.

Теперь нужно запустить (перезапустить) **mgetty.** Для этого выполните команду **init q.** Также можно воспользоваться командой **killall -1 init.**

После этого на вашем модеме должна загореться лампочка DTR. Если этого не произошло, смотрите файл `/var/log/messages` — вы что-то сделали неправильно.

Помните, в самом начале я упомянул некоторое «волшебное» слово. Допустим, что этим «волшебным» словом будет слово «пожалуйста» (please). Вот его-то и нужно записать в файл `login.config`:

```
please -  · /usr/sbin/callback -s 38400
```

Я позволил себе немного пошутить относительно выбора этого самого слова. Разницы, конечно, нет никакой — вы можете в качестве этого слова установить все, что вам угодно — даже свое имя. Обычно используются слова «callback» или «cbuser» (callback user). После этого снова следует перезапустить **mgetty.**

Вот, собственно, в чем заключается первый способ настройки. Сейчас рассмотрим второй способ, а в пункте «Настройка клиентов» будем настраивать Windows-клиенты.

17.3.3. Настройка сервера. Способ 2

Второй способ, как я уже говорил, обладает неоспоримыми преимуществами. Во-первых, вам не нужно будет писать никакие сценарии для Windows-клиентов. Во-вторых, вы сможете самостоятельно определить, какие пользователи будут использовать функцию **callback,** а какие — нет.

Но и этот способ имеет свои недостатки. Хорошо, что существенный недостаток только один — невозможна авторизация, основанная на сценариях, как в первом способе, а иногда она бывает очень даже полезной. И еще один небольшой недостаток: нужно установить дополнительный патч к

демону **pppd,** а так как патчи не всегда успевают за версиями **pppd,** то приходится использовать более старую версию **pppd.**

Сначала нужно выкачать патч к **pppd,** который реализует поддержку **callback.** Он доступен по адресу: **http://www.pbko.sk/~bobovsky/archiv/pppd-cbcpS-callback/ine-contrib/ppp-2.x.n.CBCP.patch.** Числа x и n — это номера версии демона **pppd.** Выкачивайте самую последнюю версию. Если у вас версии **pppd** старше, чем версия патча, вам придется установить более старую версию **pppd.** Различные версии **pppd** доступны по адресу **ftp://ftp.linuxcare.com.au/pub/ppp/**

Для обновления **pppd** (установки патча) используйте команду:

```
patch -p1 < ppp-2.3.10.CBCP.patch
```

Данная команда обновляет исходные тексты **pppd** (предварительно их нужно выкачать и установить). Эта же команда создает файлы:

```
/etc/ppp/callback-users
/etc/ppp/callback-client
/etc/ppp/callback-server
```

В первом из них нужно будет прописать всех пользователей, которым будет доступна функция обратного звонка. Второй нужен для работы у Linux-клиента функции **callback.** А третий управляет сервером обратного звонка.

Затем перейдите в каталог с исходными текстами **pppd** и введите три команды:

```
./configure
make
make install
```

После установки **pppd** нужно настроить **mgetty.** Напомню, что программа **mgetty** должна быть собрана с поддержкой функции **-DAUTO_PPP** (автоматическая ppp-авторизация).

Далее отредактируйте файл /etc/mgetty+sendfax/login.conf. Он должен содержать одну строку:

```
/AutoPPP/ — a_ppp /usr/sbin/pppd auth -chap +pap login callback-server
```

После этого пропишите своих пользователей в файле callback-users (см. листинг 17.15).

Листинг 17.15. Файл /etc/ppp/callback-users

```
# User list for callback
# Username option
# option – no callback
# option * or empty user definied
# option other admin definied: this number
# in username * and ? wildcars valid, callback uses the best fit
# Examples:
# zotyo 67435 # user zotyo admin definied, number 67453
# gates -- # gates not called back may *
cbuser *
user 320779
* -
```

Первый пользователь — cbuser. Согласно опции * — это пустое определение пользователя — для тестирования. Второй пользователь user — это реальное определение пользователя, телефон для обратного звонка — 320779. Все остальные пользователи не будут использовать функцию callback — опция «-».

С помощью команды **chmod** сделайте сценарии **callback-server** и **callback-client** исполнимыми. После этого необходимо немного отредактировать скрипт **callback-server** (см. листинг 17.16).

Листинг 17.16. Файл /etc/ppp/callback-users

```
#!/bin/sh
# Script callback-server
# Script parameters: delay time in seconds, callback number
DELAY="$1"
NUMBER="$2"
/usr/sbin/chat -v -t 2 "" ATHO
sleep $DELAY
/usr/sbin/chat -v "" AT OK ATS39=5DT$NUMBER CONNECT
```

Данная конфигурация уже должна работать, но иногда модем не успевает инициализироваться, поэтому после команды **sleep $DELAY** следует добавить еще одну команду **sleep,** например, **sleep** *25.* Обратите внимание: используется тональный набор (AT-команда **DT).** Напомню, что для импульсного набора используется команда **DP** (ATDP).

Вот и все, осталось только проверить корректность работы сервера.

17.3.4. Настройка клиентов. Способ 1

Создайте новое соединение и приступите к его конфигурации. При этом, на вкладке «Тип сервера» выключите все параметры, кроме программного сжатия данных. Единственный допустимый сетевой протокол — TCP/IP (см. рис. 17.2).

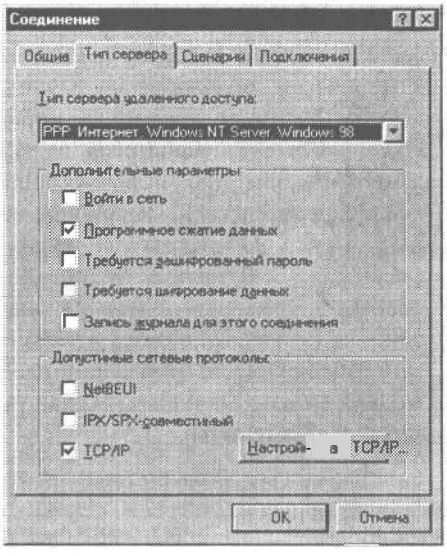


Рис. 17.2. *Свойства соединения*

Затем создайте в любом текстовом редакторе, например, в **Блокноте,** сценарий для обратного звонка (листинг 17.17).

Листинг 17.17. Сценарий для callback

```
proc main
   delay 1
 # это ваше «волшебное» слово
   transmit "please^M"
 # Ожидание запроса номера
 # телефона
   waitfor "phone"
 # Передача номера
   transmit "123456^M"
 # Ожидание вызова
    waitfor "RING"
```

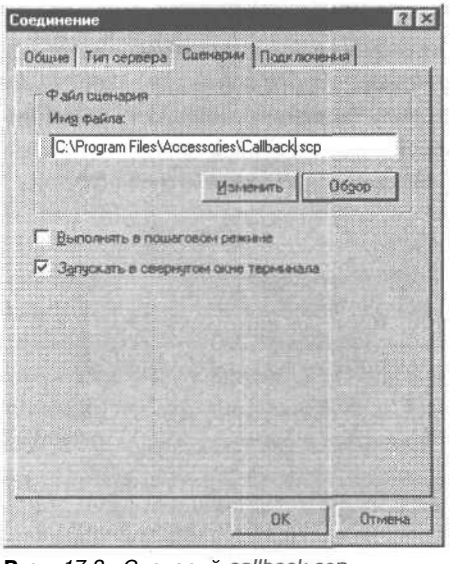


Рис. *17.3. Сценарий callback.scp*

```
# Ожидание соединения модемов
    wainfor "CONNECT"
endproc
```

Сохраните свое творение как файл `callback.scp`. В операционной системе Windows NT данный файл нужно записать в каталог `\WINNT\SYSTEM32\RAS`.

После этого перейдите на вкладку **Сценарии** и выберите только что созданный сценарий (см. рис. 17.3).

Затем перейдите на вкладку Общие и нажмите на кнопку «Настройка». В появившемся окне перейдите на вкладку **Подключение** и нажмите на кнопку «Дополнительно». Далее, в строке инициализации модема необходимо ввести AT&C1SO=1 (см. рис. 17.4). Команда &C1 устанавливает сигнал CD - - без него этот способ работать не будет. Команда SO устанавливает количество звонков, после которых модем клиента будет снимать трубку (1 звонок). Для модема Motorola Premier 33.6 установите такую строку инициализации:

```
AT&F&C0S0=1Q0V1&D3\V4
```

Более серьезный пример файла сценария **callback** вы найдете на прилагаемом компакт-диске в каталоге `/mnt/cdrom/doc/callback`.

В операционной системе Windows NT обратный вызов настраивается несколько иначе. С этой целью откройте окно запуска удаленного доступа: **Пуск → Программы → Стандартные → Удаленный доступ** (Start → Programs → Accessories → Remote access). Нажмите на кнопку «Другое» («More»). Выберите пункт меню «Параметры пользователя» («Users preferences») и перейдите на вкладку **Ответный вызов** («Callback»). Отметьте пункт «Да, требуется ответный вызов по указанным номерам». Номер телефонной линии, на которой установлен модем, можно изменить, нажав на кнопку «Изменить».

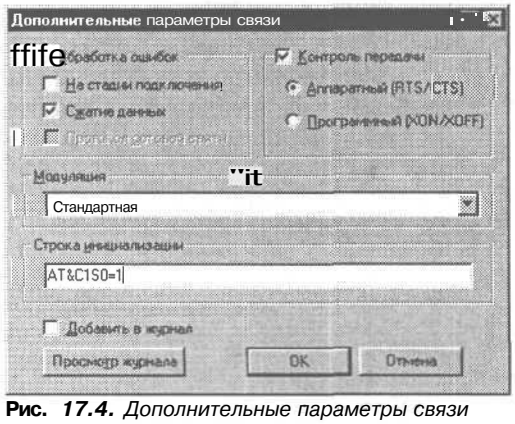


Рис. ***17.4.*** *Дополнительные параметры связи*

17.3.5. Настройка клиентов. Способ 2

Как я уже говорил, для второго способа не нужно создавать никаких сценариев для Windows-клиентов. И, как правило, никаких проблем с настройкой Windows здесь не возникает — нужно просто использовать обыкновенное соединение. Базовая настройка соединения производится так же, как и в первом случае (см. рис. 17.2).

Однако при использовании второго способа могут возникнуть проблемы с настройкой Linux-клиентов. На Linux-клиентах должна быть установлена та же версия **pppd,** что и на серверах. И так же, как и на сервере, ее необходимо пропатчить. После обновления демона **pppd** нужно настроить файл `/etc/ppp/callback-client` (см. листинг 17.18).

Листинг 17.18. Файл /etc/ppp/callback-client

```
#!/bin/sh
# Script callback-client
# Script parameters: delay time in seconds
DELAY="$1"
# Кладем трубку
/usr/sbin/chat -v -t 2 "" \d+++\d\c OK ATH0 OK
# Вместо параметра $DELAY установите значение, которое подходит
# для вашего модема
# Подойдет delay 25 или даже delay 30
sleep $DELAY
# Ожидание callback
/usr/sbin/chat -v "" ATZ OK "" RING ATA CONNECT
```

В файле `ppp-on` нужно вызывать демон **pppd,** что можно сделать следующим способом:

```
/usr/sbin/pppd auth -chap +pap login callback
```

18 Компилирование ядра

В этой главе будет рассмотрены все этапы компилирования ядра, а также приведены рекомендации по повышению производительности системы.

В показательных целях мною в примерах использовано ядро 2.2.17, но ниженаписанное верно также и для более поздних версий ядер (2.3.x , 2.4.x).

18.1. Параметры ядра

Во время загрузки ядру ОС Linux могут быть переданы различные параметры. В этой главе будут рассмотрены не все параметры ядра (полное их описание занимает достаточно много места). За более подробным их описанием вам следует обратиться к BootPrompt-HOWTO. Передача параметров может быть осуществлена либо с помощью загрузчика LILO, либо с помощью любого другого загрузчика Linux (например, bootlin, bootact). В том случае, если вы решили использовать LILO, то в ответ на приглашение нужно ввести:

```
linux строка_параметров.
```

где: **linux** — метка, указанная в файле /etc/lilo.conf.

Вторым способом указания ядру параметров является команда **append,** используемая в файле конфигурации LILO — /etc/lilo/conf. Параметры при этом следует указывать в следующем виде:

```
параметр[=значение1][,значение2]...[,значениеN]
```

Значения разделяются запятой без пробелов. Если нужно указать несколько параметров, используйте пробел для их разделения.

Пример строки параметров:

```
// правильное объявление параметров
root=/dev/hdal ether=9,0x300,0xd0000,0xd4000,eth0
// неправильное объявление параметров
root=/dev/hdal ether=9, 0x300, 0xd0000, 0xd4000, ethO
```

18.1.1. Параметры корневой файловой системы

Итак, начнем описание параметров, с параметров корневой файловой системы:

root=устройство

Устанавливает корневую файловую систему. Например, root=/dev/hda1. В качестве устройства допустимыми являются:

1. /dev/hdaN .. /dev/hddN — для IDE-дисков;
2. /dev/sdaN .. /dev/sdeN — для SCSI-дисков;
3. /dev/xdaN .. /dev/xdbN — для XT-совместимых дисков;
4. /dev/fdN — дисковод для дискет. N=0 — диск A, N=1 — диск B;
5. /dev/nfs — не является устройством, но указывает ядру, что нужно произвести загрузку по NFS.

ro

Этот параметр указывает монтирование корневой файловой системы в режиме «только чтение». Используется по умолчанию.

rw

Задает монтирование корневой файловой системы в режиме «чтение/запись». При использовании этого параметра нельзя запускать программы типа **fsck.** Перед запуском программы **fsck** нужно перемонтировать корневую файловую систему в режиме **ro**.

18.1.2. Управление RAMDISK

При создании загрузочных дискет для ОС Linux необходимо, чтобы на эти дискеты было помещено нужное программное обеспечение и чтобы для этого программного обеспечения хватило места. Обычно поступают следующим образом: создают сжатый архив всего необходимого программного обеспечения и помещают его на загрузочный диск. При загрузке системы в памяти создается «электронный» диск, на который это программное обеспечение и записывается. Этот «электронный» диск называется **RAM-диском.** Описываемые далее параметры задают режимы работы с RAM-диском.

ramdisk_start=<смещение>

Разрешает ядру находиться на гибком диске вместе со сжатым образом RAM-диска.

Ядро не может быть включено в сжатый образ файловой системы RAM-диска, так как оно должно быть записано начиная с нулевого сектора, чтобы BIOS могло загрузить загрузочный сектор и ядро могло бы продолжить загрузку.

Если вы используете несжатый образ RAM-диска, то ядро может быть частью образа файловой системы. Такая дискета может быть загружена с помощью LILO.

В том случае, если вы для загрузки используете две дискеты (первая содержит ядро — **boot,** на второй находится образ файловой системы — **root),** образ файловой системы должен начинаться на нулевом секторе и смещение = 0.

load_ramdisk=

Этот аргумент заставляет ядро использовать RAM-диск. Значение load_ramdisk= 1 сообщает ядру, что нужно загрузить дискету в RAM-диск. Значение по умолчанию 0 (ядро не использует RAM-диск).

prompt_ramdisk=

Сообщает ядру, что нужно запросить дискету, которая содержит образ файловой системы (пример: promt_ramdisk=1).

ramdisk_size=

Устанавливает размер RAM-диска в Кб.

ramdisk=

Определяет размер (в Кб) устройства RAM-диска. Например, для загрузочной дискеты 1.44 Мб нужно указать ramdisk=1440. Этот аргумент поддерживается ядрами, начиная с версии 1.3.47.

18.1.3. Управление памятью

Управление памятью осуществляется с помощью параметра mem:

mem=

Определяет объем памяти, установленной в компьютере.

Например: mem=l6384K или mem=16M.

Иногда нужно указать объем ОЗУ, отличный от того, который имеется на самом деле. Например, у вас чипсет Intel 810 с интегрированной видеоплатой, тогда вам нужно указать объем ОЗУ на 1 Мб меньше (а иногда даже на 2 Мб). Это связано с аппаратной особенностью чипсета. Более подробно об этом вы можете узнать на сайте компании Intel (http://www.intel.com).

18.1.4. Другие параметры ядра

debug

Сообщения ядра (важные и не очень) передаются через функцию **printk().** Если сообщение очень важно, то его копия будет передана на консоль, а также функции **klogd()** для его регистрации на жестком диске.

Сообщения передаются на консоль, потому что иногда невозможно запротоколировать сообщение на жестком диске (например, отказ самого диска). Предел того, что будет отображаться на консоли, задается переменной **console_loglevel.** По умолчанию на консоли отображается все, что выше уровня DEBUG (7). Список уровней можно найти в файле `kernel.h`.

init=

По умолчанию ядро пытается запустить программу /sbin/init, которая продолжит загрузку согласно стартовым сценариям (rc). Если программа **init** повреждена, вы можете использовать параметр **init=/bin/sh.** В оболочке вы сможете заменить поврежденную программу.

no-hit

Процессоры 386 (и выше) имеют инструкцию **hlt,** которая сообщает процессору не производить никаких действий. При этом обычно процессор

переводится в режим пониженного потребления энергии и ожидает прерывания от устройства. Параметр **no-hit** отключает использование инструкции **hlt**. Существование этого параметра обусловлено тем, что некоторые чипы 486DX-100 имеют проблемы с этой инструкцией. Кроме того, параметр no-hit позволяет использовать Linux на бракованных процессорах.

no387

Отключает использование математического сопроцессора.

no-scroll

Отключает функцию прокрутки экрана во время загрузки.

reboot=

Параметр, задающий режим перезагрузки. Возможные значения: **cold и warm,** то есть «холодная» или «горячая» перезагрузка. Поддерживается ядрами версии 2.0 и выше.

single

Устанавливает однопользовательский режим для администрирования системы, например, в случае отказа.

18.2. Конфигурирование ядра

Итак, немного разобравшись в параметрах ядра, приступим к его конфигурированию. Однако перед тем как приступить, убедитесь, что у вас установлены исходники ядра и пакет заголовков:

```
kernel-2.2.17-21mdk.i586.rpm
kernel-headers-2.2.17-21mdk.i586.rpm
```

Затем перейдите в каталог, который содержит исходные тексты ядра. Обычно это `/usr/src/linux` или `/usr/src/linux-2.2.17`. По сути **linux** — это ссылка на каталог `linux-2.2.17`. Все действия нужно выполнять от имени суперпользователя:

```
# cd /usr/src/linux
```

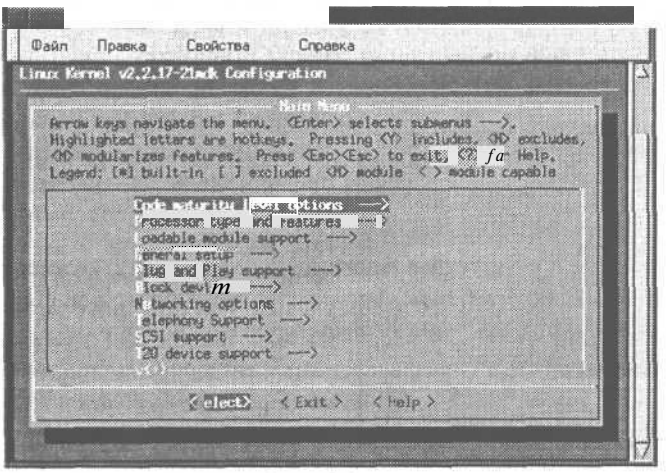


Рис. 18.1. Программа Menuconfig

Затем введите одну из следующих команд:

```
# make config
# make menuconfig
# make xconfig
```

В первом случае вам будет задан ряд вопросов, кстати, очень длинный, на который вам предстоит ответить. Я рекомендую команду **make menuconfig** — это намного удобнее (рис. 18.1). В этом случае вы можете редактировать конфигурацию

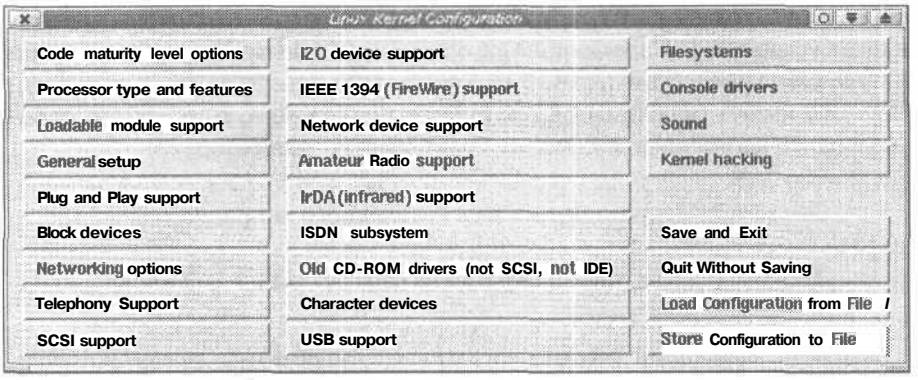


Рис. 18.2. *Программа xconfig*

ядра с помощью меню. Параметр **xconfig** аналогичен **menuconfig**, только предназначен для запуска из-под системы X-Window (см. рис. 18.2).

Перед внесением изменений в файл конфигурации ядра, сохраните его под другим именем — **Save Configuration to an Alternative File.** Во время конфигурирования ядра вы можете включать или исключать некоторые функции из состава ядра или же сделать нужную вам функцию модулем, то есть в состав ядра данная функция включена не будет, но она будет использоваться при необходимости. Например, если вы добавите в систему устройство, то будет подключен нужный модуль, при условии, что вы его откомпилировали. Главная задача — повышение **производительности** системы, этого можно достичь, если точно сконфигурировать ядро и исключить из его состава ненужный код.

18.2.1. Processor type and features

Здесь можно указать тип процессора и его функции, например, поддержка памяти более 1 Гб, MTRR, эмулирование математического сопроцессора.

Очень важно правильно указать тип процессора: после того, как я правильно указал тип своего процессора, производительность системы повысилась примерно в 1,5 раза, особенно это стало ощутимо при загрузке системы. Данная функция используется для оптимизации работы процессора. Если вы укажете тип процессора, например, 486, 586, Pentium, PPro, ядро не обязательно будет запускаться на более ранней архитектуре. Например, если вы укажете Pentium, ядро будет работать на PPro (хотя и медленнее), но нет никакой гарантии, что оно запустится на 486. В табл. 18.1 приведены типы процессоров, которые рекомендуются для получения наибольшей производительности.

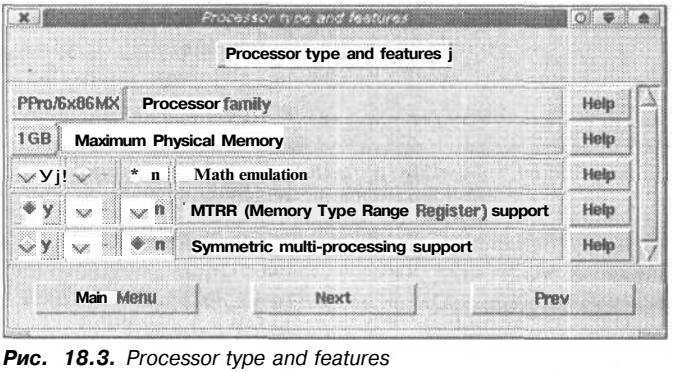


Рис. 18.3. *Processor type and features*

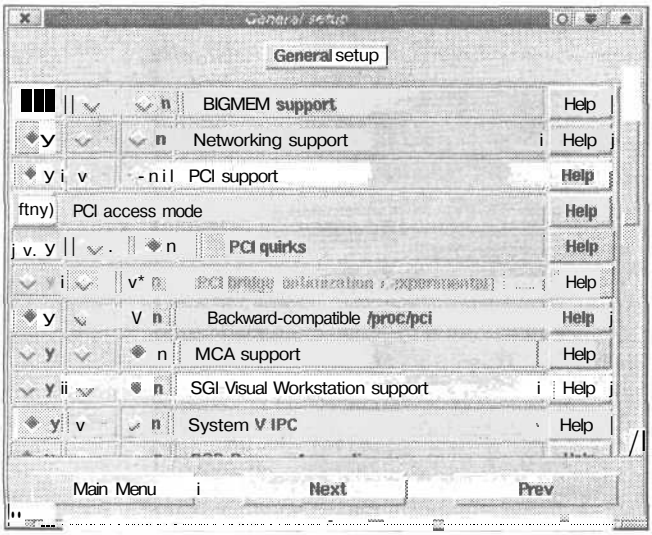


Рис. 18.4. *General setup*

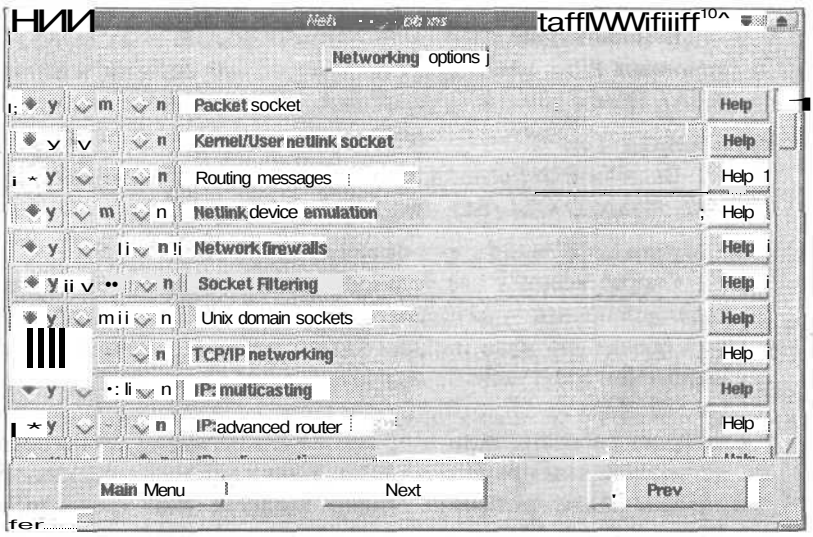


Рис. ***18.5.*** *Networking options*

Типы процессоров *Таблица 18.1*

Тип	Процессоры
386	Процессоры производства AMD/Cyrix/Intel 386DX/DXL/SL/SLC/SX, Cyrix 486DLC/DLC2, UMC 486SX-S
486/Cx486	AMD/Cyrix/Intel/IBM DX4, 486DX/DX2/SL/SX/SX2 AMD/Cyrix 5x86 NexGen Nx586, UMC U5D или U5S
586/K5/5x86/6x86	Обычные (самые первые) процессоры Pentium, AMD K5
Pentium/K6/TSC	Intel Pentium/Pentium MMX, AMD K6,K6-3D
PPro/6x86MX	Intel Pentium II/Pro, Cyrix/IBM 6x86MX, MII

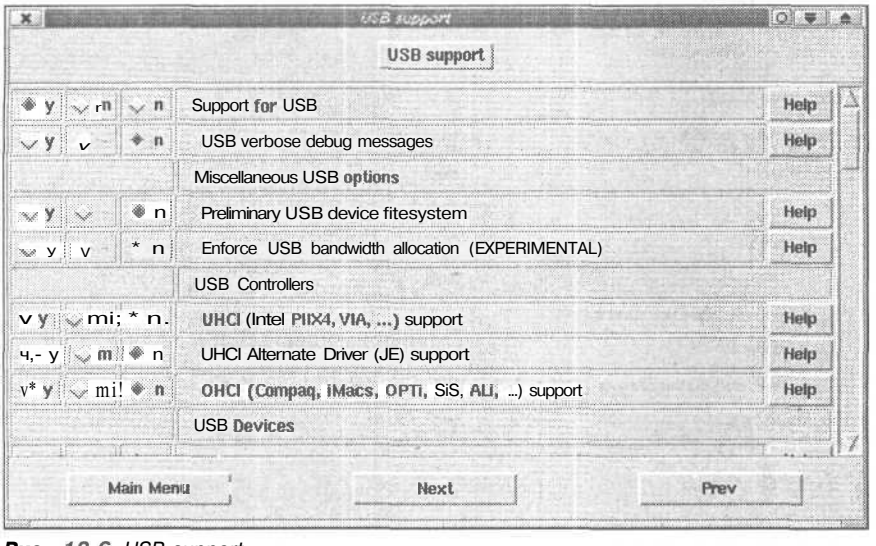


Рис. 18.6. *USB support*

В моем случае ядро было оптимизировано под 586/K5. После того, как я установил PPro, Linux заработал быстрее (для справки: я использую Intel Celeron 433A). Объем памяти — установите 1 Гб, если, конечно, у вас менее 1 Гб.

Math emulation

Включите эту опцию, если вы используете один из следующих процессоров: 386SX/DX/SL/SLC без 80387, 486SL/SX/SX2.

SMP (Symmetric multi-processing support)

Скорее всего, у вас установлен один процессор и эту опцию вам нужно будет отключить — зачем включать лишний код в ядро? Если же вы счастливый обладатель мультипроцессорной машины, включите данную опцию. При включении SMP укажите правильный тип процессора. Вы должны указать хотя бы 586. Ядро не запустится, если у вас выбран тип процессора 486. Также ядро не будет работать, если ваш компьютер оснащен процессором Pentium, а вы установили тип процессора PPro. Если у вас мультипроцессорная машина, вы должны также включить опцию **Enhanced Real Time Clock Support.** Опция **Advanced Power** Managament у вас будет отключена при использовании SMP.

MTRR

В семействе процессоров Intel P6 (Pentium Pro, Pentium II и выше) используются специальные регистры — **Memory Type Range Registers (MTRR).** Эти регистры используются для управления доступом процессора к различным диапазонам памяти. Включение этой опции может существенно повысить производительность системы, особенно если вы используете видеокарту PCI или AGP. Данную возможность поддерживают процессоры и посторонних производителей: Cyrix 6x86, 6x86MX, MII, AMD K6-2 (stepping 8 и выше), K6-3, Centaur C6. Некоторые BIOS устанавливают MTRR для первого процессора, но отключают для второго. Активизация данной опции также

решает и эту проблему. Если вы не уверены, поддерживает ли ваш процессор MTRR, все равно включите данную опцию. Поддержка MTRR увеличит объем ядра всего лишь на 3 Кб.

18.2.2. Loadable module support

Если вы планируете использовать загружаемые модули, включите все функции. Можно создать компактную версию ядра, которая вообще не использует модули, а поддержка всех необходимых устройств будет включена непосредственно в ядро. В этом случае можно отключить все функции в этой секции.

18.2.3. General setup

BIGMEM

Поддержка памяти более 1GB.

Networkingsupport

Включите эту опцию, даже если вы не планируете работу в сети. Функции печати в Linux требуют сетевой поддержки.

PCI support

Поддержка шины PCI.

PCI quirks

Эту опцию нужно использовать, если у вас неисправна BIOS. Некоторые BIOS содержат ошибки, которые могут привести к сбоям при работе с PCI. Данная опция должна исправить эту ошибку. Если вы неуверенны, включите ее. Позже можно будет поэкспериментировать. Если же BIOS исправна, эту функцию можно спокойно отключить и тем самым внести вклад в повышение производительности системы.

PCI bridge optimization (experimental)

Оптимизация моста PCI — для любителей экспериментов. Система может работать нестабильно. Попробовать можно, но я бы не стал жертвовать надежностью ради производительности.

Backward-compatible /proc/pci

Старые версии ядра поддерживали файл /proc/pci, который содержит перечень всех PCI-устройств. Некоторые программы используют этой файл, например, для сбора информации о системе. В новых ядрах используется файл /proc/bus/pci. Для поддержки обратной совместимости рекомендуется включить эту опцию. Если вы ее отключите, то у вас будет только один (новый) интерфейс /proc/bus/pci.

MCAsupport

Данная опция устанавливает поддержку шины MCA. MCA -- шина передачи данных, разработанная IBM, - - использовалась в системах PS1/PS2. Шина MCA снята с производства и не используется.

System V IPC

Просто включите эту опцию. Более подробно о ней вы можете прочитать на сайте **metalab** (ftp://metalab.unc.edu/pub/Linux/docs/LDP/programmers-guide).

BSD Process accounting

При включении этой опции программы пользовательского уровня будут информировать ядро о времени своего создания, владельце, использовании памяти и терминалов. Данную опцию рекомендуется включить.

`Sysctl` support

Включает поддержку Sysctl. Sysctl позволяет изменять параметры ядра без перекомпилирования во время загрузки. Поддержка sysctl увеличивает размер ядра на 8 Кб. Если ядро, которое вы компилируете, не предназначено для дисков загрузки/восстановления, включите эту опцию.

Kernel support for `a.out/ELF/MISC/JAVA` binaries

Linux-программы используют ELF-формат. Поэтому его нужно включить в состав ядра, а остальные использовать в качестве модулей.

Parallel port support

Поддержка параллельного порта.

PC-style hardware

Вы должны включить эту опцию (или хотя бы модулизировать ее), если вы используете параллельный порт типа PC. Все компьютеры, совместимые с IBM PC, и некоторые Alpha используют именно этот тип порта.

Support foreign hardware

Включите эту опцию, если вы используете другой (не PC) тип параллельного порта.

Advanced Power Management (APM) BIOS support

Поддержка расширенного управления питанием: ATX, «green»-устройства (например, VESA-мониторы). Если вам нужно отключить эту функцию во время загрузки, введите в качестве параметра ядра **apm=off**.

При возникновении проблем проверьте следующее:

1. Наличие достаточного количества свопа (объема файла подкачки), а также убедитесь, что раздел подкачки включен.
2. Передайте ядру инструкцию no-hit.
3. Попробуйте отключить поддержку сопроцессора (инструкция no387).
4. Передайте ядру инструкцию floppy-nodma.
5. Убедитесь, что процессор не «разогнан».
6. Установите новый вентилятор для процессора.

Support Enable PM at boot time

Включает APM во время загрузки системы. Если эта опция отключена, BIOS не будет управлять питанием устройств, входить в режимы Standby и Suspend, а также не будет производить никаких действий в ответ на вызовы процессора CPU Idle. Если ваш компьютер зависает во время загрузки, выключите эту опцию.

Make CPU `Idle` calls when idle

Во время цикла простоя ядра разрешает вызовы к APM. Включение данной опции может привести к зависанию компьютера во время загрузки! Если компьютер использует несколько процессоров, эта опция игнорируется. Заметьте, сколько процессоров именно *использует компьютер,* а не сколько в

нем установлено. Если у вас два процессора, а вы используете только один и поддержка SMP у вас отключена, данная опция игнорироваться не будет!

Enable console blanking using APM

Включает мерцание консоли при использовании APM. Некоторые лэптопы могут использовать эту опцию для того, чтобы отключить подсветку LCD-экрана, когда активизирован хранитель экрана на одной из виртуальных консолей Linux.

Ignore multiple suspend/resume cycles

Эта опция необходима для ноутбуков Dell Inspiron 3200 и некоторых других для нормальной работы APM. Прежде чем активизировать эту опцию, прочитайте документацию по вашему ноутбуку.

RTC stores time in GMT

Если ваш аппаратный таймер сохраняет время в формате GMT, включите эту опцию, иначе она должна быть отключена. Если опция выключена, сохраняется локальное время. Рекомендуется сохранять время в формате GMT.

Allow interrupts during APM BIOS calls

Обычно прерывания внешних устройств запрещены во время выполнения процедур APM. BIOS некоторых ноутбуков разрешает прерывания внешних устройств, например, IBM ThinkPad. По умолчанию данная опция выключена. Если вы не уверены, не включайте ее.

18.2.4. PnP support

В данной секции задается поддержка Plug and Play.

18.2.5. Block devices

Normal PC floppy disk support

Если вы хотите использовать FDD в Linux, включите эту опцию.

Enhanced **IDE/MFM/RLL disk/cdrom/tape/floppy** support

Выключите эту опцию, если ваша система оснащена только SCSI-устройствами.

Use old disk-only driver on primary interface

Данная опция устанавливает старый драйвер для управления Primary master интерфейсом IDE. Обычно ее нужно отключить, чтобы был использован новый драйвер для всех четырех дисков (Primary master, Primary slave, Secondary Master, Secondary slave). Ее также нужно отключить, если у вас на компьютере используются только SCSI-устройства.

Include **IDE/ATA-2** Disk support

Поддержка IDE/ATA-2 дисков. Опцию можно отключить, только если вы не используете ATA-диски.

Use multi-mode by default

При возникновении ошибки вида:

```
hda: set_multmode: status=0x51
hda: set_multmode: error=0x04
```

включите эту опцию.

Include IDE/ATAPI CDROM support

Поддержка привода CDROM. При отсутствии такового отключите ее для уменьшения размера ядра.

Include IDE/ATAPI TAPE support

Поддержка IDE/ATAPI-стримера.

Include IDE/ATAPI FLOPPY support

Поддержка IDE/ATAPI-флопии. Если вы используете LS-120 или Iomega-ZIP, включите эту опцию.

SCSI emulation support

Позволяет использовать драйвер SCSI для устройств с интерфейсом ATAPI, для которых нет родного драйвера. Все остальные опции в данной секции предназначены для поддержки конкретных чипсетов. Рекомендую оставить поддержку только тех устройств, которые имеются в вашей системе.

18.2.6. Networking options

Packet Socket

Протокол **Packet** используется программами, которые обмениваются данными непосредственно с сетевыми устройствами без промежуточных сетевых протоколов, например, **tcpdump.**

Kernel/User netlink support

Просто включите эту опцию. В данной секции задания сетевых настроек я рекомендую включить опции, указанные в табл. 18.2.

Опции сети *Таблица 18.2*

Опция	Описание
Routing messages	Сообщения маршрутизатора
Netlink device emulation	Опция обратной совместимости. Скоро будет удалена, но сейчас она нужна
Network firewalls	Поддержка firewall
Socket Filtering	Фильтр сокетов
UNIX domain sockets	Поддержка UNIX-сокетов. Не отключайте эту опцию
TCP/IP networking	Поддержка TCP/IP обязательно должна быть включена
IP:firewalling (*)	IpChains
IP:firewall packet (*)	IpChains
IP: transperent proxy support (*)	Прозрачный прокси
IP: masquerading (*)	IP-маскарадинг
IP: ICMP masquerading (*)	ICMP-маскарадинг
IP: masquerading virtual server support (*)	IP-маскарадинг для виртуальных серверов
IP: broadcast GRE over IP (*)	Поддержка broadcasting в WAN
IP: aliasing support	Поддержка псевдонимов
IP: TCP syncookie support	Рекомендуется включить из соображений безопасности. Противодействует SYN-атакам
IP: allow large windows	Позволяет повысить производительность при работе в сети. Не рекомендуется при объеме памяти менее 16 Мб

Опции, отмеченные звездочкой (*), требуются в случае конфигурирования сервера. При создании сервера также потребуется включение ряда дополнительных опций в зависимости от его назначения. Из соображений безопасности включение поддержки firewall и на рабочей станции не будет лишним.

18.2.7. SCSI support

В данной секции можно установить параметры SCSI. При отсутствии в системе SCSI-устройств можно отключить все.

18.2.8. Network device support

Здесь можно указать поддерживаемые протоколы (например, PPP), а также типы поддерживаемых сетевых адаптеров. Отключите все, что не используете. Например, если у вас установлена PCI-сетевая плата, то особого смысла включения поддержки других ISA-сетевых плат я не вижу.

18.2.9. IrDA, USB support

Поддержка соответственно IrDA- и USB-устройств.

18.2.10. Filesystems

Здесь принцип такой: все, что нужно — встраивайте в ядро, остальное — или исключайте, или используйте в качестве модуля. При этом я рекомендую включить в ядро следующие файловые системы:

Second ext fs (ext2), ISO 9660, MS Joliet CDROM extension, VFAT, /proc, /dev/pts

В виде модулей следующие: MS DOS FAT, MINIX.

Также неплохо было бы включить поддержку квотирования, если вы настраиваете сервер.

18.2.11. Sound

Включите поддержку звука, если у вас есть звуковая плата. Во всем остальном следует поступить так, как и в случае с сетевыми платами — включить в ядро только используемую звуковую плату, а все остальные отключить (даже не использовать в качестве модуля). Точнее включить код драйвера для вашей звуковой платы непосредственно в ядро вам, скорее всего, не удастся, но вы его можете использовать в качестве модуля, а все остальные модули даже не компилировать.

78.3. Компилирование ядра

Теперь, когда все устройства сконфигурированы, нужно сохранить файл конфигурации ядра и перейти непосредственно к этапу компилирования ядра.

Введите команду:

```
# make dep
```

После завершения ее работы необходимо ввести команду:

```
# make bzImage
```

Если исходники ядра и компилятор установлены корректно, то примерно минут через 20 (это зависит от версии ядра и от быстродействия вашей системы) вы получите откомпилированное ядро. Обычно оно помещается в каталог /usr/src/linux/arch/i386/boot.

Теперь следует откомпилировать модули, которые будут использоваться ядром:

```
# make modules
```

и установить их:

```
# make modules_install
```

Перед установкой модулей сделайте резервную копию модулей старого ядра (каталог /lib/modules). Теперь можно ввести команду:

```
# make install
```

Однако для установки только что созданного ядра, я не рекомендую этого делать. Сначала нужно протестировать ваше ядро. С этой целью откройте в любом редакторе файл /etc/lilo.conf:

```
# vi /etc/lilo.conf
```

Добавьте в него следующие строки (более подробно файл /etc/lilo.conf обсуждался в гл. 4):

```
image=/usr/src/linux/arch/i386/boot/bzImage
label=my_linux
root=/dev/hda5
append=" mem=128M"
read-only
```

Потом введите команду:

```
# lilo
```

Теперь перезагрузите систему:

```
# reboot
```

Попробуйте загрузить ядро. В случае возникновения ошибок вы всегда сможете загрузить старую версию.

19

Полезные команды и программы. Создание RPM-пакетов

19.1. Общие команды

Цель данной главы — вспомнить «хорошо забытое старое». Из современных пользователей ОС Linux мало кто пользовался или вообще знает о существовании таких программ как **mail,** fastmail, **locate, which.** Эти небольшие программы позволяют сделать многие операции намного быстрее, чем программы, обладающие графическим интерфейсом и предназначенные для работы в графической системе X Window. К тому же эти программы не так чувствительны к системным ресурсам компьютера. Если у вас, например, Intel Pentium 166 и 32 Мб ОЗУ, то после того, как вы раз попробуете поработать с программой **pine,** вам уже никогда не захочется запускать программу **netscape** с параметром «-mail» для чтения почты.

Возможно, некоторые из команд, которые я здесь опишу, у вас работать не будут, потому что не установлены соответствующие им пакеты. Здесь я не буду подробно описывать все параметры программ, а только опишу, для чего предназначена та или иная программа. Чтобы не возникало вопросов, напомню, что с точки зрения Linux нет разницы между понятиями «программа» и «команда».

appres класс библиотека

Программа **appres** выводит список ресурсов X, которые используются приложением.

Например: $ appres `xterm`

arch

Выводит тип аппаратной платформы компьютера.

Пример использования: $ arch.

Пример результата: `i686`.

banner параметры строка

Выводит строку, рисуя буквы с помощью звездочек. Длина строки должна быть не более десяти символов.

Ьс [параметры] [сценарии]

Программа **bc** является калькулятором, который может производить вычисления с произвольной точностью. Профессиональный математический пакет, конечно, программа **bc** не заменит, но с ее помощью можно производить довольно сложные расчеты. Калькулятор работает в интерактивном режиме, однако он также может обрабатывать сценарии. Язык сценариев очень похож на С.

Пример использования:

```
bc
scale=3
sqrt(4)+2
4.000
quit
```

Команда **scale** задает точность вычисления, то есть определяет, сколько знаков должно быть после запятой. Команда **sqrt** вычисляет квадратный корень.

Калькулятором поддерживаются циклы **for**, **while**, операторы **continue, break.** Функция s вычисляет синус аргумента, с — косинус, а — арктангенс, е — экспоненту, 1 — натуральный логарифм.

bdftopcf [параметры] файл.bdf

Программа **bdftopcf** предназначена для преобразования шрифтов в формате BDF в формат PCF. Формат PCF обычно используется сервером X. Программа **bdftopcf** имеет девять параметров, указанных в табл. 19.1.

Параметры программы bdftopcf *Таблица 19.1*

Параметр	Описание
-l	Не вычисляет метрику шрифтов
-l	Устанавливает биты в шрифте: младший бит — первый
-L	Установка порядка байтов: младший байт будет первым
-т	То же, что и -l, но первым будет старший бит
-M	То же, что и -l, но первым будет старший байт
-o file	Записать результат в указанный файл
-pn	Устанавливает размер строки развертки для каждой строки в п байт. Допустимые значения: 1, 2, 4
-t	Преобразование шрифта в терминальный
-un	Определяет положение строки развертки. Допустимые значения: 1, 2, 4

bmtoa [параметры] файл

Программа **bmtoa** преобразует растровый рисунок в текстовый файл. Существует также обратная программа — **atobm.**

cal [параметры] месяц год

Программа **cal** выводит календарь на заданный месяц.

cksum файлы

Программа *cksum* вычисляет контрольную сумму (CRC) для группы файлов. Эту программу можно использовать и для одного файла.

cpio [параметры] файл

Программа **cpio** используется для создания архивов как на магнитной ленте, так и на жестком диске. По умолчанию создается архив на магнитной ленте.

date

Программа **date** используется для отображения даты и времени. Пользователь root может использовать ее для установки времени и даты. Для подробного описания форматов отображения даты и времени, введите команду **man date.** Дата устанавливается в формате MMddhhmmyy — месяц, число, час, минуты, год соответственно. Например, для установки даты 14 января 2002 года и времени 16:35 введите команду: date 0114163502

df [параметры] устройство

Выводит информацию о свободном месте на указанном устройстве. Если устройство не указано, выводится информация о свободном месте для каждой смонтированной файловой системы.

free [параметры]

Команда free выводит информацию об использовании оперативной и виртуальной памяти. По умолчанию информация выводится в килобайтах. Если вы хотите вывести информацию в байтах или в мегабайтах, используйте параметры -Ь и -т соответственно.

dumpkeys

Данная программа выводит информацию о текущей раскладке клавиатуры.

echo [параметры] строка

Команда **echo** выводит текст или значения переменных окружения. Если задан параметр **-n**, команда не будет выводить в конце строки символ новый строки, то есть не будет перевода строки.

fsinfo

Выводит информацию о запущенном сервере шрифтов системы X Windows.

id имя_пользователя

Выводит информацию об указанном пользователе. Если пользователь не указан, выводится информация о текущем пользователе. Выводятся UID, GID, а также все группы, членами которых является пользователь.

info тема

Программа **info** подобна команде **man.** Иногда программа **info** сообщает дополнительные сведения или даже содержит темы, которых нет в справочной системе **man.**

insmod [параметры] модуль

Вставляет загружаемый модуль в ядро. Данную команду имеет право выполнять только пользователь root.

kill [параметры] PID

Отправляет процессу указанный сигнал. Программа **kill** подробно рассмотрена в гл. 5.

killall [параметры] имя

Отправляет указанный сигнал группе процессов с указанным именем.

login пользователь [параметры]

Идентифицирует пользователя. Можно использовать для регистрации под другим именем. Для регистрации как суперпользователь используйте команду **su.**

lsmod

Выводит список загруженных модулей ядра.

md5sum [параметры] файл

Эта программа подсчитывает и проверяет контрольный код с использованием алгоритма MD5. Параметр **-b** позволяет трактовать любой файл как двоичный, а параметр -t как текстовый. С помощью параметра **-c файл** можно проверить целостность файла, а параметр -s **строка** вычисляет хэш-код указанной строки.

minicom [параметры]

Minicom — терминальная многофункциональная программа. Данная программа обладает таким количеством функций, что для их описания можно было бы написать еще одну главу. Обратитесь к справочному руководству по программе **minicom.**

nice [параметры] команда [аргументы]

Устанавливает приоритет запускаемой программы. Команда **nice** уже рассматривалась в гл. 5.

passwd [пользователь] [пароль]

Изменяет пароль пользователя. Пользователь root может изменить пароль любого пользователя системы. Обыкновенный пользователь может изменить только свой пароль. Новый пароль не должен быть словарным словом и содержать не менее шести символов. Суперпользователь имеет право установить любой пароль, то есть новый пароль может быть словарным словом и содержать менее шести символов, например, list.

pathchk

Проверяет корректность указанного полного имени файла.

printenv переменная

Выводит значение переменной окружения. Однако обычно удобнее для этой цели использовать команду **echo $имя_переменной**

ps параметры

Выводит список процессов (см. гл. 5).

renice приоритет процесс [параметры]

Изменяет приоритет заданного процесса. В отличие от команды **nice, renice** задает приоритет для уже запущенного процесса.

reset

Выполняет начальную инициализацию (сброс) терминала.

rmmod [параметры] модуль

Удаляет загружаемый модуль из ядра. Данную команду имеет право выполнять только пользователь root.

rx [параметры] файл

Позволяет получить файл по протоколу **XModem**. Эта программа входит в пакет **minicom**.

rxvt

Программа **rxvt** представляет собой упрощенный вариант программы **xterm**.

rz [параметры] файл

Позволяет получить файл по протоколу **ZModem.** Эта программа входит в пакет **minicom.**

sb [параметры] файл

Позволяет отправить файл по протоколу **YModem**. Эта программа входит в пакет **minicom.**

setterm параметры

Данная программа позволяет установить параметры терминала. Более подробную информацию вы получите в справочной системе.

sleep число

Данная программа приостанавливает выполнение дальнейших операций на указанное время. Число задается в секундах. Обычно используется при написании сценариев.

startx

Запускает систему X Window, если она у вас не запускается автоматически. Программа **startx** обычно является сценарием запуска программы **xinit**.

strace [параметры] программа

Используется для трассировки системных вызовов и сигналов. Очень полезная программа при отладке программного обеспечения.

stty настройки параметры

Программа stty позволяет просматривать и устанавливать параметры терминала. Без параметров она выводит установки для текущего терминала.

sx [параметры] файлы

Позволяет отправить файл по протоколу **XModem**. Эта программа входит в пакет **minicom.**

systat параметры система

Программа **systat** опрашивает указанную систему, используя службу systat. Данная служба должна быть доступной на опрашиваемой машине. Если служба systat недоступна, опрашиваются службы daytime и netstat. При вызове программы **systat** можно использовать параметры, указанные в табл. 19.2.

sz [параметры] файлы

Позволяет отправить файл по протоколу **ZModem.** Эта программа входит в пакет **minicom.**

Параметры команды systat *Таблица 19.2*

Параметр	Описание
-n	Опрос службы netstat
-p порт	Используется указанный порт
-s	Опрос службы systat
-t	Опрос службы daytime

tee [параметры] файлы

Отправляет информацию со своего стандартного ввода в указанные два файла. Если указан только один файл, то перенаправляется вывод в этот файл и на стандартный вывод. Если файлы существуют, программа их перезапишет. Запретить перезапись можно с помощью опции «-a». Эту программу очень удобно использовать в таком контексте:

```
$ some_program | tee file
```

В данном примере вы можете спокойно работать с интерактивной программой some_program (название используется произвольное), все сообщения, которые выведет программа, будут записаны в файл file с помощью программы tee. Вы же увидите эти сообщения на консоли «без потери качества», поскольку программа **tee** перенаправит их на стандартный вывод. В файле, естественно, вы не увидите ту информацию, которую вводили при работе с программой some_program.

tload [параметры]

Выводит график загрузки системы.

top [параметры]

Этой программой вы, скорее всего, уже пользовались. Данная программа выводит список процессов, отсортированный по занимаемому процессорному времени. Можно использовать параметры, представленные в табл. 19.3.

Параметры программы top *Таблица 19.3*

Параметр	Описание
Dсекунды	Задержка между обновлением списка процессов
Q	Обновление без задержек
S	Информация о процессорном времени, занимаемом завершенными процессами-потомками
S	Программа не будет работать в интерактивном режиме
I	Программа будет игнорировать неактивные процессы

true

Программа ничего не делает, но код ее завершения равен нулю. Это означает успешное завершение. Данная программа часто используется в сценариях.

ul [параметры] терминал

Программа **ul** заменяет весь текст, имеющий атрибут подчеркивания, на подчеркнутый. Используется пользователями терминалов, подключенных к ОС Linux.

users

Выводит информацию о пользователях, которые зарегистрированы в данный момент в системе.

w

Выводит различную информацию о системе: список пользователей, которые зарегистрированы в системе, загрузку системы, выполняемые пользователями задачи.

which программа

Сообщает имя каталога, в котором располагается указанная программа.

who

Выводит список пользователей, зарегистрированных в системе.

whoami

Выводит информацию о текущем пользователе. Данная команда является сокращением команды **who** am i.

write пользователь

Читает со стандартного ввода сообщение и выводит его на консоль указанного пользователя. Данный пользователь должен быть зарегистрирован в системе.

xclipboard

Выводит содержимое буфера обмена системы X Window.

xcmap

Выводит текущую цветовую палитру в виде таблицы.

xconsole [параметры]

Выводит в окне X сообщения, которые отправляются на устройство /dev/console. Параметры программы приведены в табл. 19.4.

Параметры программы xconsole *Таблица 19.4*

Параметр	Описание
-daemon	Режим демона (фонового процесса)
-file файл	Использовать указанный файл вместо устройства /dev/console
-notify	Отображение имени приложения, которое отправило сообщение на консоль
-nonotify	Параметр, обратный параметру notify
-verbose	Вывод дополнительной информации
-exitOnFail	Завершение работы при ошибке

xcpustate

Выводит информацию о состоянии процессора.

xdpyinfo [параметры]

Выводит различную информацию о сервере X.

xeyes [параметры]

Рисует два глаза, следящих за указателем мыши.

xf86config

Данная программа создает файл конфигурации сервера **X** — **XF86Config**. Иногда эту программу удобнее использовать, чем другие конфигураторы.

xfd [параметры]

Выводит в окне все символы указанного шрифта в виде таблицы. Указать шрифт можно с помощью параметра **-fa font_name.**

xfontsel

Очень удобная программа для выбора шрифтов. Можно просматривать все шрифты, а можно только те, которые соответствуют определенному шаблону. Шаблон создается пользователем.

xfractint

Небольшой генератор фракталов. В состав оконной среды KDE входит собственный генератор фракталов — **kfract.**

xgc

Данная программа демонстрирует графические возможности системы X Window.

xhost [параметры]

Управляет доступом пользователей различных систем к серверу X. Параметр **+система** разрешает доступ данной системы к серверу X, а параметр **-система** — запрещает. Если не указывать параметры, вы получите текущие установки доступа.

xkill

Программа **xkill** принудительно завершает любую X-программу. Фактически эта программа разрывает соединение между сервером X и выбранным клиентом. Приложения для системы X Window (X-программы) являются клиентами сервера X. Для завершения работы той или иной программы достаточно щелкнуть на ее окне после запуска программы **xkill.** Очень полезная программа!

xlock

Программа **xlock** блокирует сервер X. Для снятия блокировки нужно ввести пароль пользователя. Программа **xlock** аналогична программе **vlock,** которая «запирает» консоль.

xlogo

Выводит логотип сервера X.

xman

Графический аналог программы **man.**

xmessage [параметры] сообщение

Выводит диалог с указанным сообщением. Внешний вид диалога можно задать с помощью параметров программы (см. табл. 19.5). Данная программа

Параметры программы xmessage *Таблица 19.5*

Параметр	Описание
-buttons кнопки	Определяет кнопки, которые должны быть отображены в окне диалога. Список состоит из пар имя_кнопки:код_возврата, разделенных запятыми
-default имя_кнопки	Имя кнопки по умолчанию
-file файл	Вывод содержимого указанного файла
-print	Вывод имени нажатой кнопки на стандартный вывод

аналогична программе echo, но предназначена для работы с сервером X. Обычно эту программу удобно использовать в сценариях.

`xon [параметры] программа`

Позволяет запускать указанную программу на удаленном сервере X. За более подробным описанием обратитесь к справочной системе.

xpaint

Простейший графический редактор. Поддерживаются форматы TIFF, PPM, XBM.

xrefresh

Обновляет (перерисовывает) весь экран.

xsetroot [параметры]

Программа xsetroot изменяет настройки главного окна X, то есть рабочего стола. Более подробную информацию вы можете получить в справочной системе.

xterm

Запускает самый обычный терминал, предназначенный для работы в системе X Window.

xv [параметры] файл

Программа xv позволяет просматривать и редактировать изображения в форматах GIF, JPEG, TIFF, PGM, PPM, PBM, BMP, PCX, IRIS RGB и во многих других. Эту программу удобно использовать для просмотра изображений. Для редактирования лучше использовать программу gimp.

xvidtune[параметры]

Программа предназначена для более точной настройки сервера X. Можно настроить видеорежимы, время отключения мониторов и другие параметры.

xwd [параметры]

Программа xwd предназначена для захвата изображения окна или его части и сохранения этого изображения в файле.

yes строка

Данная команда непрерывно выводит указанную строку. Если строка не указана, выводится символ «y». Обычно используется для написания сценариев.

19.2. Команды для работы с файлами

Некоторые команды из этой группы уже были подробно рассмотрены в гл. 4, поэтому здесь они рассматриваться не будут.

basename файл расширение

Удаляет из имени файла путь и расширение, если оно задано. Используется при написании сценариев.

chgrp [параметры] группа файлы/каталоги

Команда chgrp предназначена для изменения группы, которой принадлежат указанные файлы или каталоги. Использует несколько параметров, позволяющих автоматизировать операции по изменению группы (см. табл. 19.6).

Параметры команды chgrp *Таблица 19.6*

Параметр	Описание
-R	Рекурсивный режим
-v	Вывод более подробной информации
-f	Не выводит сообщения о файлах и каталогах, группа которых не может быть изменена
-c	Вывод сообщений о сделанных изменениях

chown [параметры] владелец файлы

Команда **chown** предназначена для изменения владельца указанных файлов или каталогов. Для того, чтобы сменить владельца файла, вы должны сами быть владельцем этого файла или пользователем root. Параметры этой команды аналогичны параметрам команды **chgrp** (см. табл. 19.6).

chroot каталог

Данная команда изменяет корневой каталог системы на указанный. Эту команду может выполнять только пользователь root.

file [параметр] файл

Определяет тип указанного файла.

find путь условия

Программа **find** производит поиск файла. При этом программа **find** намного функциональнее, чем все известные мне программы поиска. Эта программа может производить как простейший поиск, как это делает большинство программ поиска, так и поиск, удовлетворяющий множеству условий. Возможные условия поиска представлены в табл. 19.7.

Условия поиска программы find *Таблица 19.7*

Параметр	Описание
-amin выражение	Поиск файлов, доступ к которым производился в зависимости от выражения: +т — более т минут назад; m — минут назад; -т — менее т минут назад
-anewer файл	Поиск файлов, доступ к которым производился после даты последнего изменения указанного файла
-atime выражение	Поиск файлов, доступ к которым производился в зависимости от выражения: +d — более d дней назад; d — дней назад; -d — менее d дней назад
-cmin выражение	Поиск файлов, изменение которых произошло: +т — более т минут назад; m — минут назад; -т — менее т минут назад
-cnewer файл	Поиск файлов, которые были изменены после даты последнего изменения указанного файла
-ctime выражение	Поиск файлов, которые были изменены: +d — более d дней назад; d — дней назад; -d — менее d дней назад
-daystart	Считать время от начала дня, а не от текущего момента
-empty	Поиск пустых файлов и каталогов
-exec команда	Выполнение указанной команды для каждого из найденных файлов
-follow	Следовать ссылкам (как на файл, так и на каталог)
-fstype тип	Поиск файлов только на файловых системах указанного типа

Продолжение табл. 19.7

Параметр	Описание
-gid номер	Поиск файлов, принадлежащих указанной группе
-iname файл	Поиск указанного файла. В выражении «файл» могут использоваться символы подстановки (маски файла), регистр букв игнорируется
-inum номер	Поиск файла с указанным номером информационного узла (i-node)
-ipath путь	Поиск файла по указанному пути, регистр букв игнорируется
-links n	Поиск файлов, имеющих n ссылок
-lnameфайл	Поиск символических ссылок на указанный файл
-maxdepth уровень	Поиск в дереве каталогов не глубже указанного уровня
-mindepth уровень	Поиск в дереве каталогов на указанном уровне и глубже
-name имя	Поиск файла с указанным именем
-nogroup	Поиск файлов, которые не принадлежат ни одной из групп, указанных в файле /etc/group
-nouser	Поиск файлов, владельцы которых не перечислены в файле /etc/passwd, то есть не являются пользователями системы
-ok команда	Аналогично параметру -exec, но перед выполнением команды запрашивается подтверждение у пользователя
-path путь	Поиск файлов по указанному пути
-perm режим_доступа	Поиск файлов с указанным режимом доступа
-size n	Поиск файлов размером n блоков
-type тип_файла	Поиск файла указанного типа: b — блочное устройство, c — символьное устройство, d — каталог, f — обычный файл, l — ссылка, p — поток
-user пользователь	Поиск файлов, принадлежащих указанному пользователю

gzip [параметры] файлы

Архивирует указанные файлы. После сжатия исходный файл удаляется, а полученный файл будет иметь то же имя, что и исходный, но с суффиксом **.gz.** Параметры программы **gzip** представлены в табл. 19.8.

gzexe [параметры] программа

Программа **gzexe** сжимает указанную программу. При запуске сжатого файла программа распаковывается и запускается. Это увеличивает время запуска, но позволяет сэкономить место на жестком диске. По сути, программа **gzexe** является самораспаковывающимся архивом (SFX). Опция -d позволяет разархивировать сжатую программу.

locate [параметры] шаблон

Производит поиск файла в базе данных файлов. Поиск производится по шаблону.

Параметры программы gzip *Таблица 19.8*

Параметр	Описание
-c	Архивация производится на стандартный вывод, исходный файл не удаляется
-d	Распаковка (то же, что и gunzip)
-f	Архивация производится даже в тех случаях, когда исходный файл уже является сжатым, имя создаваемого файла совпадает с уже существующим файлом, на исходный файл установлено несколько ссылок
-l	Сжатие файла уже сжатого другими программами (pack, deflate, lzh, compress)
-q	Быстрый режим работы: не выводятся информационные сообщения
-r	Работа в рекурсивном режиме
-t	Проверка целостности архива после сжатия
-v	Вывод дополнительных сообщений

mkfifo [параметры] имя

Создает именованный поток FIFO (First In First Out). Права доступа по умолчанию 0666 минус значение, выдаваемое командой **umask**. С помощью параметра **-m права_доступа** можно задать права доступа для создаваемого потока.

mkfontdir каталог

Используется для создания списка шрифтов, находящихся в указанном каталоге. Список (или индекс) шрифтов будет помещен в файл fonts . dir в этом же каталоге. Список шрифтов необходим для сервера X.

mknod [параметры] имя тип номер дополнительный_номер

Программа **mknod** используется для создания потоков FIFO, блочных, символьных устройств. Права доступа, как и в случае с программой **mkfifo,** равны 0666 минус значение **umask.** Тип создаваемого устройства задается при помощи параметра **тип,** указанного в командной строке (см. табл. 19.9).

Типы специальных файлов *Таблица 19.9*

Тип	Описание
P	Поток (FIFO)
B	Блочное устройство
C	Символьное устройство
U	Символьное устройство

Можно также использовать параметр -т **права_доступа**. Этот параметр определяет права доступа для только что созданного устройства.

pwd

Выводит имя текущего каталога.

size [параметры] программа

Программа size выводит размеры сегментов и общий размер программы. Можно использовать параметры, представленные в табл. 19.10.

sq исходный_файл файл_результата

Программа sq сжимает отсортированный по алфавиту список слов. Эта команда обычно используется для сжатия больших справочников. Для распаковки сжатого справочника используйте команду **unsq**. Данная программа имеет те же параметры, что и sq.

umask маска

Устанавливает маску для создания файлов. Каждый новый файл или каталог будет иметь маску прав доступа, равную 0777 минус значение **umask.** Значение **umask** по умолчанию равно 022.

Параметры программы size *Таблица 19.10*

Параметр	Описание
-A	Вывод в формате программы size UNIX SysV
-B	Вывод в формате программы size BSD
-d	Вывод в десятичной системе счисления
-o	Вывод в восьмеричной системе счисления
-x	Вывод в шестнадцатеричной системе счисления
--target имя	Использование другого формата объектных файлов

updatedb[параметры]

Программа **updatedb** обновляет базу данных, которую использует программа **locate** для поиска файлов. Эта база данных содержит список всех файлов, которые содержатся в указанных в конфигурационном файле каталогах. С помощью параметров программы можно изменить настройки базы данных (см. табл. 19.11).

Параметры программы updatedb *Таблица 19.11*

Параметр	Описание
--localpaths=dir1 dir2...	Помещает в базу данных сведения о файлах, которые находятся в указанных каталогах
--netpaths=dir1 dir2 ...	Помещает в базу данных сведения о файлах, которые находятся в указанных каталогах на удаленной системе
--netuser=username	От имени этого пользователя будет производиться поиск файлов на удаленной машине
--old-format	Создание базы данных будет произведено в старом формате
--output=файл	Использовать указанный файл в качестве базы данных
--prunepaths=dir1 dir2	Не помещать в базу данных сведения о файлах в указанных каталогах

xfilemanager или **xfm**

Запускает графический менеджер файлов. Настоятельно советую использовать вместо него программу Midnight Commander (**mc**), которая обладает большими возможностями и более удобна.

zcat архив

Распаковывает архив на стандартный вывод.

znew [параметры] архив.Z архив.gz

Программа **znew** преобразует архивы, созданные программой **compress,** в новый формат — **grip.** Можно использовать параметры, представленные в табл. 19.12.

Параметры программы znew *Таблица 19.12*

Параметр	Описание
-9	Наибольшее сжатие
-f	Разрешение перезаписи существующих файлов
-K	Формат архива будет изменен, если это приведет к уменьшению его размера
-P	Использование потока для передачи файла
-t	Проверка целостности нового архива перед удалением старого
-v	Вывод дополнительных сведений

zoo [параметры] архив

Еще одна программа-архиватор. Для более подробного описания обратитесь к справочной системе.

19.3. Команды для работы с Интернет

audiocompose файл

Программа **audiocompose** записывает звуковое сообщение, которое вы потом сможете отправить по электронной почте. Для работы этой программы должна быть правильно установлена звуковая плата.

audiosend адрес

Программа **audiosend** используется как для записи, так и для отправления звуковых сообщений по электронной почте.

biff

Данная программа уведомляет вас о приходе новой почты. Для работы этой программы нужна программа **sendmail.**

chfn [параметры]

Данная программа изменяет информацию о пользователе, которую можно получить с помощью программы **finger.**

dnshostname [параметры] система

Выводит текущее имя системы. Пользователь root может использовать ее для изменения имени системы.

dnsquery [параметры]

Опрашивает серверы DNS. Установить параметры запроса можно с помощью параметров программы (см. табл. 19.13).

Параметры программы dnsquery *Таблица 19.13*

Параметр	Описание
-n сервер	Использовать указанный сервер DNS
-t тип	Установка типа записи: A, ANY, HINFO, NS, CNAME, PTR, SOA, WKS, MX и другие
-c класс	Установка классов записи: IN, CHAOS, HS, ANY.
-p п	Установка таймаута
-r п	Установка количества попыток, если сервер не отвечает
-s	Использование потока

elm [параметры]

Программа elm является интерактивным почтовым клиентом. Данная программа имеет больше возможностей, чем программа **mail,** однако она менее функциональна, чем программа **pine.**

fastmail [параметры] файл список_адресов

Данная программа позволяет отправить указанный файл большой группе пользователей. Эту программу очень удобно использовать для сопровождения списка рассылки. Параметры программы указаны в табл. 19.14.

faucet [параметры] порт

С помощью программы **faucet** можно организовать сетевой поток через сеть. Обычно **faucet** используется в качестве сервера, а **hose** (см. ниже) в качестве клиента.

Параметры программы fastmail *Таблица 19.14*

Параметры	Описание
-b список	Отправка слепых копий по указанному списку адресов
-c список	Отправка копий по указанному списку адресов
-C комментарий	Установка поля Comments заголовка сообщения
-d	Режим отладки
-f from	Установка поля from (от) заголовка сообщения
-i ID	Установка поля Message-ID (идентификатор сообщения) заголовка сообщения
-r reply-to	Установка поля reply-to (ответ) заголовка сообщения
-R описание	Установка описания для данного сообщения
-s subject	Тема сообщения (поле subject)

finger [параметры] пользователь

Программа **finger** получает информацию о пользователе, которую можно изменить с помощью программы **chfn**. Для работы службы **finger** необходимо, чтобы было разрешено использование порта 79 в файле `/etc/services` на указанной системе. В противном случае вы получите сообщение Connection refused. Обычно порт 79 запрещают только на маршрутизаторе, чтобы никто извне не мог получить доступ к этой службе. В локальной сети, как правило, эта служба используется.

frm

Выводит список тем и отправителей всех сообщений в вашем почтовом ящике.

ftp [параметры] система

Программа **ftp** представляет собой обычный клиент FTP, который присутствует практически в каждой операционной системе. Параметры программы описаны в табл. 19.15, а команды ftp — в табл. 19.16.

Параметры программы ftp *Таблица 19.15*

Параметр	Описание
-d	Режим отладки
-д	Не разрешает использовать символы подстановки в именах файлов
-i	Запрет интерактивного режима
-n	Не регистрироваться на удаленной системе после установления соединения. Для регистрации вручную, введите команду user
-v	Отображает всю информацию, полученную с удаленной системы

Команды для работы с клиентом FTP *Таблица 19.16*

Команда	Описание
account пароль	Устанавливает пароль для входа в систему
append файл1 файл2	Присоединить файл «файл1» на локальной машине к файлу «файл2» на удаленной
Ascii	Включение режима ASCII для передачи файлов. Данный режим включен по умолчанию
Bell	Подача звукового сигнала по окончании передачи файлов
Binary	Включение двоичного режима передачи файлов
Bye	Закрытие текущего соединения и завершение работы с ftp
Case	Изменяет все прописные буквы в названиях принимаемых файлов на строчные
cd каталог	Изменяет текущий каталог на удаленной машине
Cdup	Переходит в родительский каталог

Продолжение табл. 19.16

Команда	Описание
Chmod	Изменяет права доступа к файлам на удаленной машине. Вы должны иметь соответствующие права на выполнение этой операции
Close	Закрытие текущего соединения
Cr	Фильтрация символов возврата каретки в текстовых файлах
delete файл	Удаляет файл на удаленной машине
dir каталог файл	Вывод содержимого заданного каталога на удаленной машине в указанный файл. Если файл не указан, то на стандартный вывод
Disconnect	Аналогична команде close
get файл1 файл2	Копирует файл «файл1», который находится на удаленной машине, в файл «файл2» на локальной. Если параметр «файл2» не указан, то исходное имя файла не изменяется
Glob	Разрешение символом подстановки в командах mget, mdelete, mput
Hash	Выводит индикатор передачи файлов. Индикатор состоит из меток #. Каждая метка выводится после окончания передачи каждого блока файла
help команда	Вывод справки по указанной команде
idle n	Установка времени простоя на удаленной системе в n секунд
Image	Установка двоичного режима передачи и приема файлов
lcd каталог	Изменяет каталог на локальной машине
ls каталог файл	Аналогична команде dir
macdef файл	Определяет макрос. Определение должно заканчиваться пустой строкой и будет сохранено в указанном файле
mdelete файлы	Удаление нескольких файлов на удаленной машине
mdir шаблон	Вывод списка файлов в текущем каталоге, содержащих шаблон
mget файлы	Копирует несколько файлов с удаленной машины
mkdir каталог	Создает каталог на удаленной машине
mls каталог файл	Аналогична команде dir
mode режим	Установка режима. Режим по умолчанию — stream (поток)
modtime файл	Выводит дату и время последней модификации файла
mput файлы	Копирует указанные файлы на удаленную машину
newer файл	Копирует файл с удаленной системы, если он новее, чем соответствующий файл на локальной машине
open система [порт]	Устанавливает соединение с указанной машиной. Если параметр «порт» не задан, используется порт по умолчанию
Prompt	Запрещает или разрешает интерактивные приглашения
proxy команда	Выполнение команды на другом соединении
put файл1 файл2	Копирует файл «файл1» с локальной машины в файл «файл2» на удаленную машину. Если второй параметр не задан, исходное имя файла не изменяется
Pwd	Выводит текущий каталог на удаленной машине
Quit	Аналогична команде bye
recv файл1 файл2	Копирует файл «файл1», который находится на удаленной машине, в файл «файл2» на локальной. Если параметр «файл2» не указан, то исходное имя файла не изменяется
reget файл1 файл2	Копирует файл «файл1», который находится на удаленной машине, в файл «файл2» на локальной. Если параметр «файл2» не указан, то исходное имя файла не изменяется. Возобновляет копирование, если во время предыдущего копирования произошел сбой, но часть файла уже скопирована на локальную машину
remotestatus файл	Запрос у удаленной системы состояния указанного файла. Если файл не указан, запрашивается состояние удаленной системы
rename файл1 файл2	Переименование файла на удаленной машине
Reset	Сброс буфера передачи
restart позиция	Повтор передачи, начиная с заданной позиции. Позиция определяет байтовое смещение

Продолжение табл. 19.16

Команда	Описание
rmdir каталог	Удаление каталога на удаленной системе. Вы должны иметь соответствующие права доступа
Runique	Включает режим уникальных имен. Если вы копируете файл, который уже существует, его имени будет назначен суффикс .1, .2 и так далее
send файл1 файл2	Копирует файл «файл1» с локальной машины в файл «файл2» на удаленную машину. Если второй параметр не задан, исходное имя файла не изменяется
size файл	Вывод информации о размере указанного файла
Status	Вывод информации о текущем соединении
struct имя	Изменение структуры передачи файлов. По умолчанию используется stream
sunique	Включение режима уникальных имен на удаленной машине
system	Вывод имени операционной системы удаленной машины
trace	Трассировка пакетов
type режим	Установка режима передачи файлов. По умолчанию используется текстовый режим
umask маска	Установка маски создания файлов
user имя пароль номер	Установка имени и пароля пользователя, а также номера подключения
verbose	Вывод подробных информационных сообщений
! команда параметры	Выполняет указанную команду на локальной машине с передачей ей указанных параметров
$ макрос аргументы	Запуск макроса на локальной машине
? команда	Вывод справки

fuser [параметры] файлы

Выводит список процессов, которые используют данный файл.

getpeername[параметры]

Выводит информацию о соединении, использующем гнезда.

hose [параметры]

С помощью программы **hose** можно организовать сетевой поток через сеть. Обычно **faucet** используется в качестве сервера, а **hose** (см. ниже) в качестве клиента. Для более подробной информации обратитесь к справочной системе.

host [параметры] система [сервер]

Программа **host** используется для преобразования имени системы в IP-адрес. Вы также можете ввести IP-адрес для его преобразования в имя системы. Параметр «сервер» задает сервер DNS, который вы хотите опросить. Данная программа имеет много параметров, описание которых вы найдете в справочной системе.

hostname [параметры] система

Выводит текущее имя системы. Пользователь root может использовать ее для изменения имени системы.

lynx [параметры] URL

Программа **lynx** является текстовым браузером. Эта программа не выводит рисунков, различных шрифтов. Благодаря этому значительно повышается скорость доступа к Web. Кроме самого URL-адреса, можно указать и некоторые параметры (см. табл. 19.17). Ограничения, используемые в работе **lynx,** приведены в табл. 19.18.

Параметры программы lynx *Таблица 19.17*

Параметр	Описание
-anonymous	Использовать анонимную регистрацию
-ascii	Не переводить символы японского алфавита в латинские символы, если включена поддержка японского языка
-auth=имя пароль	Использовать указанную информацию для аутентификации
-book	Загрузить страницу с избранными ссылками
-cache n	Установить количество кэшируемых документов
-case	Учитывать регистр букв при поиске
-cfg=файл	Использовать указанный файл в качестве файла конфигурации
-crawl	Выводить каждую страницу в файл, если указан параметр -traversal, или на экран, если указан параметр -dump
-display=сервер	Использовать указанный сервер X для запуска программы rexeced
-dump	Преобразовать страницу в текстовый вид и отправить результат на стандартный вывод
-editor=программа	Использовать указанную программу в качестве редактора
-emacskeys	Раскладка клавиатуры как в редакторе emacs
-force_html	Считает первый документ документом HTML
-ftp	Запрещает соединения FTP
-get_data	Использование метода GET для отправления данных
-head	Отправляет запрос HEAD для заголовков MIME
-homepage=URL	Устанавливает домашнюю страницу
-image_links	Включает ссылки на все изображения
-index=URL	Устанавливает индексный файл
-localhost	Запрещает доступ к локальной машине
-loexec	Запрещает выполнение программ на локальной машине, исполнимые файлы которых находятся на удаленной машине
-mime_header	Вывод заголовка MIME полученного документа при просмотре исходного текста документов
-nobrowse	Не просматривать каталоги
-noexec	Не выполнять программы на локальной машине
-nolist	Не выводить список ссылок при указании параметра -dump
-nolog	Не отправлять сообщения об ошибках авторам документов
-noprint	Запрет функции печати
-noredir	Запрещает автоматическое перенаправление и выводит сообщение с указанием нового URL
-nostatus	Не выводить информацию о процессе получения документа
-numbers_links	Нумерация ссылок
-post_data	Отправка данных с помощью метода POST
-realm	Запрещает указывать URL при запуске
-reload	Сброс кэша прокси-сервера
-restrictions= ограничения	Устанавливает определенные ограничения (см. табл. 19.18)
-rlogin	Не распознавать команды rlogin
-selective	Просматривать только каталоги, которые содержат файл .www_browsable
-show_cursor	Вывод курсора в начале выбранной ссылки
-source	Отображает страницы Web в виде исходного текста HTML
-telnet	Не распознавать команды TELNET
-term=терминал	Устанавливает тип терминала
-trace	Режим трассировки
-traversal	Обход всех ссылок на стартовой странице
-underscore	Переключает формат вывода подчеркивания при использовании параметра -dump
-validate	Разрешает использовать URL только с указанием протокола, например, http://www.linux.ru
-vikeys	Раскладка клавиатуры как в редакторе vi

Ограничения *Таблица 19.18*

Ограничение	Описание
All	Используются все ограничения, перечисленные ниже
Bookmark	Запрещает изменение расположения файла избранных ссылок
bookmark_exec	Запрещает переходы по ссылкам из файла избранных ссылок
change_exec_perms	Запрещает изменение режима доступа для файлов
Default	Запрещает использовать службы по умолчанию для анонимных пользователей
dired_support	Запрещает управление файлами на локальной системе
disk_save	Запрещает сохранение двоичных файлов на диске
Download	Запрещает загрузку файлов
Editor	Запрещает режим редактирования
Exec	Запрещает выполнение сценариев
exec_frozen	Не разрешает редактировать сценарии на локальной системе
file_url	Запрещает использование URL вида file://
goto	Запрещает команду goto
inside_ftp	Запрещает использование ftp для систем вне домена
unside_news	Запрещает отправление новостей Usenet для систем внутри домена
inside_rlogin	Запрещает использование rlogin для систем вне домена
inside_telnet	Запрещает использование telnet для систем внутри домена
jump	Запрещает команду jump
mail	Запрещает исходящую почту
options_save	Запрещает сохранение настроек в файле .lynxrc
outside_ftp	Запрещает использование ftp для систем вне домена
outside_news	Запрещает отправление новостей Usenet для систем вне домена
outside_rlogin	Запрещает использование rlogin для систем вне домена
outside_telnet	Запрещает использование telnet для систем вне домена
print	Запрещает печать
shell	Запрещает выполнение интерпретатора командной строки
suspend	Запрещает переходить в фоновый режим при нажатии Ctrl+Z
telnet_port	Запрещает указывать номер порта для соединения telnet

mail [параметры] пользователи

Программа **mail** используется для чтения, создания, отправления электронной почты. Данная программа является одной из первых почтовых клиентов. Интерфейс этой программы не очень удобен, но она присутствует практически в каждой системе.

metamail [параметры]

Программа **metamail** позволяет работать с мультимедийными расширениями электронной почты. Обычно данная программа вызывается другими программами для обработки нетекстовой информации.

mimeencode [параметры]

Программа **mimeencode** предназначена для кодировки формата MIME. Системы на базе DOS/Windows/Windows NT не поддерживают кодирование UU (Unix to Unix), для обмена информацией с этими системами используется кодирование MIME. Поддержку кодирования UU в системах DOS/Windows можно организовать *с* помощью программ сторонних разработчиков. При добавлении какого-нибудь файла в качестве вложения электронного письма предпочтительнее использовать кодирование MIME. Параметры программы представлены в табл. 19.19.

Параметры программа mimeencode *Таблица 19.19*

Параметр	Описание
-b	Использование кодирования Base64 (по умолчанию)
-q	Использование кодирования quotes-printable
-u	Декодирование
-p	Замена символов CR/LF на символ новой строки
-o файл	Запись результата в файл

Иногда вместо программы **mimeencode** используется программа **mmencode** с аналогичными параметрами.

pine [параметры] адрес

Один из самых удобных почтовых клиентов. Программа обладает достаточно удобным интерфейсом и поддерживает MIME-кодирование.

ping система

Программа **ping** позволяет «пропинговать» указанную систему. Данная программа отправляет пакеты ICMP ECHO_REQUEST (код 3) на указанную систему. Эта программа используется для определения пропускной способности сети. Для завершения работы программы нажмите Ctrl+C.

popclient [параметры] система

Позволяет получать почту, используя протоколы POP2 или POP3. Обычно данная программа практически не используется, так как все почтовые клиенты имеют свои POP-клиенты.

rusers [параметры] система

Данная программа выводит список пользователей, подключенных к указанной системе или ко всем системам сети. Параметр -l выводит более подробную информацию.

showaudio файлы

Программа **showaudio** позволяет прослушать сообщения, записанные с помощью программы **audiocompose**.

sliplogin имя

Позволяет установить соединение, используя протокол SLIP (Serial Line Internet Protocol). Нужная информация хранится в файле /etc/slip.hosts. Скорее всего, у вас не будет данной программы, так как протокол SLIP считается устаревшим и вместо него используется протокол PPP.

talk пользователь [терминал]

Позволяет двум пользователям вести диалог. Если вы хотите пообщаться с пользователем другой системы, используйте следующую форму указания имени пользователя: имя_пользователя@имя_системы.

telnet [параметры] система [порт]

Устанавливает соединение с помощью протокола Telnet.

tftp система

Программа **tftp** представляет собой тривиальный клиент FTP (отсюда буква «t» в ее названии). Обычно данная программа предназначена для

обмена данными по протоколу TFTP. Данный протокол используется для обмена информацией с интеллектуальными маршрутизаторами и не используется никакой аутентификации. Программа **tftp** поддерживает следующие команды: ascii, **binary, connect, get, put, quit, status, timeout, verbose.**

uudecode файл

Программа **uudecode** предназначена для UU-декодирования указанного файла.

uuencode файл

Программа **uuencode** предназначена для UU-кодирования указанного файла.

xbiff

Программа **xbiff** является графическим аналогом программы **biff,** но предназначена для работы в системе X Window.

19.4. Обработка текста

bpe файл

Программа **bpe** — это редактор двоичных файлов. Данная программа позволяет редактировать файл в двоичном или шестнадцатеричном режиме. После запуска программы вы можете использовать множество команд редактирования. Некоторые из них представлены в табл. 19.20.

Команды программы bpe *Таблица 19.20*

Команда	Описание
+	Прокрутка вперед на две строки
-	Прокрутка назад на две строки
/	Поиск строки, начиная с текущей позиции
9	Справочная информация
D	Вывод одной страницы
E	Редактирование в текстовом режиме
E	Редактирование в шестнадцатеричном режиме
F	Поиск строки, начиная с текущей позиции
H	Поиск последовательности байтов, начиная с текущей позиции
N	Переход к следующему сектору
P	Переход к предыдущему сектору
Q	Выход без сохранения
S	Установить позицию в файле
W	Запись изменений на диск

cat [параметр] файл

В этой книге команда cat неоднократно использовалась, но нигде я не упомянул о ее параметрах. Все параметры команды cat рассмотрены в табл. 19.21.

cmp [параметры] файл1 файл2

Программа стр сравнивает два файла и выводит различия. Если файлы не отличаются, программа ничего не выводит. Если найдено различие, то выводится номер строки и номер символа в строке. Эту программу можно использовать как для текстовых, так и для двоичных файлов. Параметры программы представлены в табл. 19.22.

Параметры команды cat *Таблица 19.21*

Параметр	Описание
-A	Вывод всех непечатаемых символов. Не выводятся символы перевода строки и символы табуляции. В конце каждой строки выводится символ «$». Символ табуляции заменяется на последовательность символов «^I»
-e	В конце каждой строки выводится символ «$»
-n	Вывод нумерации строк
-s	Не выводит пустые строки
-t	Заменяет символ табуляции на последовательность символов «^I». Символы протяжки страницы заменяются на «^L»
-T	Заменяет символ табуляции на последовательность символов «^I»
-v	Вывод всех непечатаемых символов. Не выводятся символы перевода строки и символы табуляции

Параметры программы cmp *Таблица 19.22*

Параметр	Описание
-c	Программа выведет отличающиеся символы
-i n	Программа будет игнорировать первые n байтов обоих файлов
-l	Выведет позиции всех различий
-s	Программа не будет выводить информацию на экран. Код возврата программы cmp будет равен: 0 — если файлы не отличаются; 1 — если файлы отличаются; 2 — если произошла ошибка

column [параметры] файл

Программа **column** форматирует текст файла, разбивая его на колонки. Исходный текст может быть введен со стандартного ввода. Параметры программы приведены в табл. 19.23.

Параметры программы column *Таблица 19.23*

Параметр	Описание
-c n	Задает число колонок
-s символ	Указанный символ будет использоваться в качестве разделителя колонок. Этот параметр используется вместе с параметром -t
-t	Форматирование выполняется в виде таблицы. Разделителем по умолчанию является символ пробела. Другой разделитель задается параметром -s
-x	Заполняет сначала колонки, а потом строки

csplit [параметры] файл образец

Данная программа позволяет разбить текстовый файл на несколько частей. Части файла определяются требуемым размером или с помощью указанного образца. Исходный файл при этом не изменяется. Имена частей файла начинаются с символов **xx.** Первая часть будет иметь имя **xxOO,** вторая — **xx01** и так далее. Программа **csplit** может разбить файл максимум на сто частей. За более подробной информацией обратитесь к справочной системе.

diff [параметры] [расширенные_параметры] файл1 файл2

Программа **diff** предназначена для сравнения файлов. Она выводит отличающиеся строки. Строка файла «файл1» помечается символом «<», а строка файла «файл2» помечается символом «>». С помощью тире разделяется

содержимое файлов. Описание основных параметров данной программы сведено в табл. 19.24. Программу **diff** можно использовать для сравнения файлов из разных каталогов. Для этого используются расширенные параметры (см. табл. 19.25).

Основные параметры программы diff *Таблица 19.24*

Параметр	Описание
-a	Сравнение всех файлов, в том числе и двоичных
-b	Игнорируются символы пробела в конце строки
-B	Игнорируются пустые строки в файлах
-c	Вывод контекста для каждого найденного различия
-d	Игнорируются области со многими изменениями
-e	Создается сценарий редактора ed, с помощью которого можно превратить файл «файл1» в файл «файл2»
-H	Поиск только небольших изменений
-I	Игнорировать регистр букв
-I шаблон	Игнорировать строки, содержащие указанный шаблон
-n	Установить формат вывода RCS
-N	Несуществующие файлы считать пустыми
-t	Замена символов табуляции на восемь пробелов
-T	В начало выводимых строк вставить символ табуляции
-u	Вывод новой и старой версий файла в одной строке
-w	Игнорировать пробелы
-y	Вывод в две колонки

Расширенные параметры программы diff *Таблица 19.25*

Параметр	Описание
-l	Постраничный вывод
-r	Рекурсивный режим. Сравниваются файлы во всех подкаталогах
-s	Выводит имена совпадающих файлов
-S файл	Начать сравнение с указанного файла
-x шаблон	Игнорировать файлы, имена которых содержат указанный шаблон
-X шаблон	Аналогичен параметру -x

diff3 [параметры] файл1 файл2 файл3

Сравнение трех файлов. Программа **diff3** не выводит отличающиеся строки. Вместо этого она выводит следующие сообщения:

1. = = = =, если все три файла отличаются;
2. = = = 1, если первый файл отличается от второго и третьего;
3. = = = 2, если второй файл отличается от первого и третьего;
4. = = = 3, если третий файл отличается от первого и второго.

emacs [параметры] файл *

Программа **emacs** — это мощный и довольно сложный текстовый редактор. Этот редактор входит в состав большинства дистрибутивов ОС Linux. Полное описание редактора вы сможете найти в документации, поставляемой вместе с самим редактором.

expand [параметр] файл

Программа **expand** заменяет символы табуляции на определенное количество пробелов. Обычно один символ табуляции заменяется на восемь пробелов. Вы можете использовать параметр -i для замены символов табуляции только в начале строки.

fmt [параметры] файл

Программа fmt форматирует текст в файле, выравнивая его по правому краю. При этом удаляются символы новой строки. Параметры программы рассмотрены в табл. 19.26.

Параметры программы fmt *Таблица 19.26*

Параметр	Описание
-c	Две первые строки не будут форматироваться
-p образец	Будут форматироваться строки, которые начинаются с указанного образца
-s	Строки не будут объединяться
-t	Делать отступ в начале нового абзаца
-и	Установить один пробел между словами и два -- между предложениями
-w n	Устанавливает длину строки. По умолчанию длина равна 72 символам

fold [параметры] файл

Данная программа выравнивает текст по правому краю. Ширина строки по умолчанию — 80 символов. Необходимую ширину строки вы можете задать с помощью параметра **-w** n.

ghostview [параметры] файлы

Программа **ghostview** позволяет просматривать файлы в формате PostScript. Для своей работы эта программа использует интерпретатор **ghostscript.**

grep [параметры] образец файлы

Программа **grep** ищет заданный образец в указанных файлах. Возможен ввод файла со стандартного ввода. Если образец содержит пробелы, его необходимо заключить в кавычки, а если образец начинается со знака «-», то его нужно указывать с помощью параметра **-e**. Параметры программы приведены в табл. 19.27.

Параметры программы grep *Таблица 19.27*

Параметр	Описание
-A n	Выводит n строк после строки, которая содержит образец
-B n	Выводит n строк перед строкой, которая содержит образец
-b	Выводит для каждой строки, которая содержит образец, ее номер в файле
-c	Выводит только количество совпадений
-C	Выводит две строки перед строкой, содержащей образец, и две строки после нее
-e образец	Определяет образец
-f файл	Поиск образцов, которые находятся в файле
-n	Выводит строки, содержащие образец, но не выводит имена файлов
-i	Игнорируется регистр букв
-l	Выводит только имена файлов, строки которых содержат образец
-L	Выводит имена файлов, строки которых не содержат образец
-n	Вывод строк, которые содержат образец, и их номеров в файле

Продолжение табл. 19.27

Параметр	Описание
-s	Не выводит сообщение об ошибке, если один из указанных файлов невозможно открыть
-v	Вывод строк, которые не содержат образец
-w	Поиск совпадения целого слова с образцом
-x	Поиск совпадения целой строки с образцом
-число	Выводит указанное число строк до и после строки с образцом

grodvi [параметры] файл

Данная программа преобразует файл в формате **groff** в файл в формате DVI.

groff [параметры] файл

Данная программа обеспечивает работу пользователя и других программ с документами в формате **groff.** Для более подробного описания обратитесь к справочной системе.

head [параметры] файлы

Программа **head** выводит начало файла. По умолчанию выводятся первые десять строк файла. Однако число строк можно задать с помощью параметра **-n число.**

ispell [параметры] файлы

Программа **ispell** выполняет проверку правописания в указанных файлах. Обычно данная программа используется другими программами, например, текстовыми редакторами или почтовыми клиентами, для проверки правописания.

join [параметры] файл1 файл2

Программа **join** позволяет объединить два файла по общему полю. Описание программы и ее параметров вы найдете в справочной системе.

less [параметры] файлы

Данная программа предназначена для просмотра файлов. Эта программа была создана как альтернатива старой программе **more.** Программа **more** позволяет просматривать файлы только по одной странице вперед. В отличие от программы **more,** программа **less** позволяет просматривать текст в обоих направлениях и имеет множество параметров, управляющих процессом просмотра (см. табл. 19.28).

При просмотре файла можно использовать пробел для прокрутки вперед на один экран, клавишу «Enter» для прокрутки вперед на одну строку, клавишу «b» для возврата назад на один экран, а клавишу «/» для поиска образца. За более подробным описанием комбинаций клавиш обратитесь к документации.

Параметры программы less *Таблица 19.28*

Параметр	Описание
-help	Выводит краткое описание параметров и другую справочную информацию
-a	Начинать поиск с первой строки. По умолчанию поиск начинается со второй строки
-bчисло	Определяет число буферов для каждого файла. Размер буфера равен 1024 байт
-B	Не выделять буферы при чтении данных из потока
-c	Перерисовка экрана сверху вниз

Продолжение табл. 19.28

Параметр	Описание
-C	То же, что и параметр -c, но перед перерисовкой будет выполнена очистка экрана
-d	Не выводить предупреждения при недостаточных возможностях терминала
-e	Завершение работы при втором достижении конца файла
-E	Завершение работы при достижении конца файла
-f	Может использоваться для открытия файлов или файлов устройств
-m	Запуск в режиме программы more — внизу экрана будет отображаться процентное положение в файле
-M	То же, что и -т, но внизу экрана будет отображаться номер строки в файле
-п	Не выводить номера строк в файле
-N	Выводить номера строк в файле
-Офайл	Копирование информации в файл при чтении из потока. Если файл существует, он будет перезаписан без предупреждения
-q	Запрет звуковых сигналов
-r	Вывод всех символов. По умолчанию управляющие символы выводятся как «^»
-s	Замена нескольких пустых строк на одну пустую строку
-u	Символы табуляции и возврата каретки будут считаться печатаемыми
-U	Символы табуляции и возврата каретки будут считаться управляющими
-V	Вывод версии
-хшаг	Установка шага табуляции
-учисло	Устанавливает максимальную прокрутку вперед

look [параметры] образец файл

Программа **look** производит поиск указанного образца в файлах. Если файл не указан, поиск производится в системном словаре. Обычно это файл `/usr/dict/words`.

lpq [параметры] пользователь

Проверяет очередь печати указанного пользователя. Данная программа используется вместе с демоном печати **lpd**. Если пользователь не задан, то используется имя пользователя, запустившего программу. С помощью параметра **-Рпринтер** можно указать принтер.

lpr [параметры] файлы

Данная программа предназначена для постановки в очередь печати указанных файлов. Можно использовать параметр **-Рпринтер** для указания принтера. Параметр -s позволяет сэкономить дисковое пространство, так как с помощью этого параметра можно создать символическую ссылку вместо копирования файла в каталог для спула. Однако при этом вы не должны изменять файл до окончания печати. Описание других параметров вы найдете в справочной системе.

lprm [параметры] номер_задания пользователь

Данная программа позволяет удалить указанное задание из очереди печати. Вы не можете удалить задание другого пользователя, а другие пользователи не могут удалить ваше задание. Это ограничение, конечно, не распространяется на пользователя root. Номер задания можно узнать, используя программу **lpq**. Как обычно, имя принтера можно задать с помощью параметра **-Рпринтер.** Если использовать параметр «-», то можно удалить все задания.

more [параметры] файлы

Как и программа **less,** программа **more** управляет просмотром файла. Вместо нее предпочтительнее использовать программу **less,** обладающую большими возможностями. Единственное преимущество данной программы состоит в том, что она присутствует в составе даже самых старых дистрибутивов UNIX и Linux.

paste [параметры] файлы

Программа **paste** объединяет строки двух файлов.

pico [параметры] файлы

Программа **pico** является текстовым редактором. Данная программа входит в состав пакета **pine** и вызывается программой **pine** для редактирования сообщений. Параметры редактора **pico** представлены в табл. 19.29.

Параметры программы pico *Таблица 19.29*

Параметр	Описание
+число	Позиционирование на указанном номере строки
-d	Клавиша «Delete» будет использоваться для удаления символа под курсором
-e	Автоматическое завершение имен файлов
-g	Выводит курсор перед текущим выделенным блоком
-k	Удаление от курсора до конца строки вместо удаления всей строки
-m	Поддержка мыши. Только при работе в системе X Window
-nсекунды	Уведомляет о прибытии новой почты. Наличие новой почты проверяется через указанное время
-o каталог	Устанавливает рабочей каталог
-rчисло	Определяет положение правой границы
-t	Обычно используется при вызове редактора из других программ, например, из программы pine
-v	Просмотр файла
-w	Не переносить строки
-x	Не выводить строку подсказок в нижней части экрана
-z	Разрешает перевод редактора в фоновый режим при нажатии Ctrl+Z

pr [параметры] файл

Программа **pr** предназначена для подготовки файла для печати. Эта программа не печатает файл. Распечатать вы его должны самостоятельно. Для подготовки файла к печати можно использовать параметры, приведенные в табл. 19.30.

Параметры программы pr *Таблица 19.30*

Параметр	Описание
+страница	Начало печати с указанной страницы
-колонки	Вывод в несколько колонок
-a	Печать колонок «поперек»
-b	Выравнивание колонок на последней странице
-c	Вывод управляющих символов при помощи символа «^»
-d	Установить двойной интервал между строками
-e n	Замена символов табуляции на n пробелов. По умолчанию параметр n равен 1
-F	Использовать символы протяжки страницы вместо символов новой строки для разделения страниц

Продолжение табл. 19.30

Параметр	Описание
-h заголовок	Установить заголовок
-l	Заменить последовательность пробелов на символы табуляции
-l длина	Установить длину строки (по умолчанию — 66 символов)
-т	Вывести несколько файлов одновременно. Вывод каждого файла будет производиться в отдельной колонке
-n символ число	Перед каждой строкой будет выведен ее номер, состоящий из указанного числа цифр. Если задан символ, то он будет использоваться для отделения номера от строки
-o ширина	Устанавливает ширину левого поля
-r	Не выводить сообщения об ошибке, если невозможно открыть файл
-s символ	Устанавливает разделитель для колонок
-t	Не выводить верхний и нижний колонтитул
-v	Вместо непечатаемых символов выводить их номер в восьмеричной системе
-w ширина	Устанавливает ширину страницы в символах

printf параметр

Выводит строку, используя синтаксис языка С.

sed [параметры] файл

Программа sed модифицирует файл согласно списку команд. Результат отправляется на стандартный вывод, исходный файл не изменяется. Список команд можно задать с помощью параметра **-fимя_файла**. Более подробное описание вы найдете в справочной системе.

sort [параметры] файл

Программа **sort** используется для сортировки, объединения и сравнения текстовых файлов. Выбрать режим работы вы можете с помощью параметров программы (см. табл. 19.31).

Параметры программы sort *Таблица 19.31*

Параметр	Описание
+число1[-число2]	Устанавливает поля сортировки. Сортировка выполняется от позиции строки «число1» до позиции «число2». Если аргумент «число2» не задан, то до конца строки
-b	Пробелы в начале строки игнорируются
-c	Если файлы не отсортированные, выводит сообщение об ошибке
-d	При сортировке игнорируются все символы, кроме букв, цифр и пробелов
-f	Преобразовать при сортировке строчные буквы в прописные
-l	Символы, не входящие в таблицу ASCII, будет проигнорированы
-M	Сокращенные англоязычные названия месяцев (Jan, Feb и так далее) преобразуются в сокращения, написанные прописными буквами, и сортируются в календарном порядке
-n	Сортировка по числовым значениям
-o файл	Определяет файл результата
-r	Обратный порядок сортировки
-tсимвол	Использовать указанный символ в качестве разделителя полей

split [параметры] файл1 файл2

Используется для разбиения файла на две или более частей. Установить размер каждой части можно с помощью параметра -С. Если вам нужно разбить файл на определенное количество строк, используйте параметр **-число.**

tac [параметры] файл

Выводит содержимое файла в обратном порядке — от последней строки до первой. Данная программа противоположна программе **cat.**

tail [параметры] файл

Данная программа противоположна программе **head.** Другими словами, программа **tail** выводит последние 10 строк файла. Указать другое число можно с помощью параметра **-1 число.**

tr [параметры] строка1 строка2

Программа **tr** заменяет строку *строка1* на строку *строка2.* Обычно используется для поиска и замены символов в указанных строках. За более подробной информацией обратитесь к справочной системе.

unexpand [параметры] файл

Программа **unexpand** заменяет в текстовом файле последовательность пробелов на символы табуляции. С помощью параметра -a можно заменить все последовательные пробелы на символы табуляции. По умолчанию замена производится только в начале строки. Параметр -t n определяет, сколько пробелов будут заменены на один символ табуляции. По умолчанию аргумент n равен 8.

uniq [параметр] файл1 файл2

Программа **uniq** удаляет повторяющиеся строки из файла *файл1* и записывает в файл *файл2.* Если параметр *файл2* не указан, то программа выводит результат на стандартный вывод. Параметры данной программы представлены в табл. 19.32.

Параметры программы un *Таблица 19.32*

Параметр	Описание
-c	Подсчитывает количество повторяющихся строк
-d	Выводит только повторяющиеся строки
-f n	Пропускает указанное число полей строки
-sчисло	Пропускает указанное число символов поля
-u	Выводит только неповторяющиеся строки
-w n	Сравнивает первые n символов строк
-число	Пропуск указанного числа полей в строке
+число	Пропуск указанного числа символов в строке
-check-chars=n	Сравнивает первые n символов строк
-skip-chars=n	Пропускает первые n символов в строке
-skip-fields=n	Пропускает первые n полей в строке. Поля разделяются пробелами или символами табуляции

vi [параметры] файл

Программа vi — это текстовый редактор. Более подробную информацию вы сможете получить в справочной системе.

wc [параметры] файл

Эта программа подсчитывает количество строк, символов или слов в текстовом файле. Если файл не указан, используется стандартный ввод. Данную программу удобно использовать при написании сценариев обработки текста. Параметры программы рассмотрены в табл. 19.33.

Параметры программы wc *Таблица 19.33*

Параметр	Описание
-bytes	Подсчет количества байтов
-c	Подсчет количества символов
--chars	Подсчет количества символов
-l	Подсчет количества строк
-lines	Подсчет количества строк
-w	Подсчет количества слов. Данный параметр используется по умолчанию
--words	Подсчет количества слов. Данный параметр используется по умолчанию

xedit файл

Программа **xedit** является простейшим текстовым редактором, предназначенным для работы в системе X Window. Вместо него обычно удобнее использовать редакторы **kedit** или **gedit.** Первый из них является стандартным редактором оконной среды KDE, а второй — оконной среды Gnome.

zdiff файлы

Данная программа распаковывает архивы **gzip** и выполняет программу diff.

79.5. *Создание RPM-пакетов*

Программа **RPM** предназначена для произведения всех видов операций с программным обеспечением, в том числе и для создания пакетов для установки (RPM-пакетов).

Прежде, чем описать много сухих фактов, взятых из документации, рассмотрим простой пример создания небольшого RPM-пакета. Я создал этот пакет для своей программки, которая контролирует состояние указанного последовательного порта.

Будем считать, что программа уже откомпилирована и все файлы, необходимые для ее работы, уже подготовлены. При этом понадобятся следующие файлы:

port.....откомпилированный бинарный файл.

README.....файл, который будет помещен в каталог /usr/doc/port-1.0-99.

port.1.....файл для справочной системы man.

Все эти файлы я поместил в каталог /root/port. Конечно, это не совсем корректно, но об этом будет сказано немного позже.

Для создания пакета нужно создать файл спецификаций. В файле спецификаций указывается вся информация о создаваемом пакете: название, версия, файлы программ, файлы документации, действия, выполняемые при установке пакета и при его удалении. Мой файл спецификаций для программы **port** представлен в листинге 19.1

Листинг 19.1. Файл спецификации для программы port

```
Summary: Program to control your serial device
Name: port
Version: 1.0
Release: 101
Group: Monitoring
License:GPL
```

```
Packager: Denis Kolisnichenko [dhsilabs@mail.ru]
URL:http://dkws.narod.ru
%description
Программа port предназначена для мониторинга состояния последовательного
порта. При получении сигнала (1) на какой-нибудь контакт указанного порта,
port отправляет сообщение запустившему ее пользователю на указанный email
%files
%doc /root/port/README
/root/port/port
/root/port/port.1
```

Для построения пакета нужно ввести команду:

```
# rpm -bb /root/port/port.spec
```

Если вы не допустили никаких ошибок при создании файла спецификаций, на экране вы увидите примерно такое сообщение:

```
Executing(%install): /bin/sh -e /var/tmp/rpm-tmp.33439
Proces'sing files: port-1.0-99
Finding Provides: (using /usr/lib/rpm/find-provides)...
Finding Requires: (using /usr/lib/rpm/find-requires)...
Requires: ld-linux.so.2 libc.so.6 libc.so.6(GLIBC_2.0)
Записан: /usr/src/RPM/RPMS/i686/port-1.0-99.i686.rpm
```

При этом будет создан пакет **port-1.0-99.i686.rpm**. Этот пакет будет помещен в каталог `/usr/src/RPM/RPMS/i686`.

При удалении такого пакета он будет удален из базы RPM, но удаления самих файлов не произойдет. Действия, которые нужно выполнить до и после удаления пакета из базы RPM, вы можете определить в макрокомандах **%preun** и **%postun** соответственно. Например,

```
%preun
rm -f /usr/bin/port
rm -f /usr/man/man1/port.1
```

Такой подход — самый простой выход из положения, однако он является не очень корректным. Решение этой проблемы оставляю вам в качестве домашнего задания.

А сейчас проведем небольшой эксперимент. Запустите Midnight Commander (**mc**), перейдите в каталог `/usr/src/RPM/RPMS/i686/` и «войдите» в пакет **port-l.0-99.i686.rpm** как в обычный каталог. В нем будет «подкаталог» INFO, в котором и содержится вся информация о пакете.

Что ж, вы успешно разобрались с построением простого пакета, но для создания реальных пакетов установки ваших знаний все еще не хватает. Теперь настала очередь той сухой теории, о которой я упомянул в начале этого пункта. Традиционно, процедура создания RPM-пакетов состоит из следующих этапов:

1. Извлечения исходных текстов программы из архива.
"2. Компилирование программы из исходных текстов.
3. Создание RPM-пакета.

Первые два этапа можно пропустить, что мы и сделали при создании пакета. Такое можно сделать только в случае, если программа уже откомпилирована из исходных текстов.

Программа RPM использует файл конфигурации **rpmrc**. Поиск этого файла производится в каталогах /usr/lib/rpm, /etc, $HOME. Просмотреть этот файл можно с помощью команды:

```
# rpm –showrc
```

Запись **topdir** файла конфигурации **rpmrc** содержит название каталога, в котором находится дерево подкаталогов, которое используется менеджером RPM для построения пакетов. Введите команду:

```
# rpm –showrc | grep topdir
-14: _builddir  %{_topdir}/BUILD
-14: _rpmdir    %{_topdir}/RPMS
-14: _sourcedir %{_topdir}/SOURCES
-14: _specdir   %{_topdir}/SPECS
-14: _srcrpmdir %{_topdir}/SRPMS
-14: _topdir    %{_usrsrc}/RPM
```

У меня эти подкаталоги находятся в каталоге /usr/src/RPM. Как вы видите, в этом каталоге находятся подкаталоги BUILD, RPMS, SOURCES, SPECS, SRPMS.

В каталоге BUILD создается RPM-пакет. В каталоге SOURCES находятся сжатые исходные тексты программы. В каталог RPMS помещаются созданные пакеты. Точнее, они помещаются в один из его подкаталогов, в какой именно — это зависит от архитектуры. В каталог SRPMS помещаются пакеты, содержащие исходные тексты программы. В каталоге SPECS находятся файлы спецификаций. Обычно файл спецификации называется **название_программы-версия-релиз.spec**.

Например, если у вас есть исходный текст программы в архиве, из которого вы хотите создать пакет RPM, скопируйте его в каталог SOURCES:

```
# cp source_code-1.0.tar.gz /usr/src/RPM/SOURCES.
```

По умолчанию менеджер RPM работает с пакетами, расположенными в каталоге с именем, совпадающим с названием пакета и его версией. Для нашего пакета **port** это будет каталог port-1.0-99. Менеджер пакетов будет компилировать наш пакет в каталог /usr/src/RPM/port-1.0-99.

Думаю, уже достаточно информации о каталогах RPM. Теперь перейдем к файлу спецификаций. Файл спецификаций состоит из четырех сегментов: заголовка, подготовительного, файлового, установочного. В заголовке указывается общая информация о пакете. В листинге 19.1 к сегменту заголовка относятся тэги **Summary, Name, Version, Release, Group и License.** На них мы останавливаться не будем, так как их назначение понятно из листинга 19.1.

Есть еще очень полезный тэг: **BuildRoot.** Он изменяет расположение дерева BUILD. Значением по умолчанию является /usr/src/RPM или другой каталог, задаваемый переменной окружения $RPM_BUILD_ROOT. В целях экономии дискового пространства полезно после установки удалить дерево %RPM_BUILD_ROOT. Но это дерево по умолчанию может

содержать другие файлы, относящиеся к другим пакетам. Поэтому сначала с помощью тэга **BuildRoot** нужно задать какой-нибудь временный каталог, а после установки удалить его.

В каждом сегменте находятся макрокоманды. С некоторыми мы уже знакомы — это **%description, %files, %doc, %install**. В табл. 19.34 приведено полное описание макрокоманд.

Макрокоманды *Таблица 19.34*

Макрокоманда	Описание
%description	Полное описание пакета
%prep	Подготовка архива. Здесь задаются команды для извлечения исходного текста программы и его распаковки, дополнительная подготовка исходного текста. После макрокоманды %prep задаются обычные команды shell
%setup	Макрокоманда извлечения файлов из архивов. Опция -n возволяет указать каталог, в котором будет создаваться новый пакет. Обычно распаковывается архив, расположенный в каталоге SOURCES, в каталог BUILD
%build	Макрокоманда компилирования. Обычно здесь запускается программа make с необходимыми параметрами
%files	Задает список файлов, входящих в состав пакета. При указании имен файлов должен быть указан полный, а не относительный путь. Для указания лолного пути можно использовать переменную окружения $RPM_BUILD_ROOT. Необходимые файлы уже должны быть помещены в каталог BUILD. Этого можно достичь с помощью макрокоманды %setup или с помощью макрокоманды %pre (см. ниже)
%config список	Задает список файлов, которые будут помещены в каталог /etc
%doc список	Задает список файлов, которые будут помещены в каталог /usr/doc/[package]-[ver]-[release]
%install	Этап установки программного обеспечения. Здесь нужно записать команды, которые будут устанавливать файлы, входящие в состав пакета. Удобнее использовать команду install, которую я использовал в листинге 19.1
%pre	Действия, которые будут выполнены до инсталляции пакета
%post	Действия, которые будут выполнены после инсталляции пакета
%preun	Действия, которые будут выполнены перед удалением пакета
%postun	Действия, которые будут выполнены после удаления пакета
%clean	Удаление дерева BUILD. Используется вместо опции —clean программы rpm. Обычно содержит одну команду: rm -rf $RPM_BUILD_ROOT

Нужно сделать небольшое замечание относительно макрокоманд **%config** и **%doc**. В этом случае список задается не так, как в макрокоманде **%files**. Если после макрокоманды **%files** можно было просто указать по одному файлу в каждой строке, то в макрокоманде **%doc** каждому файлу (или каждому списку) должна предшествовать команда **%doc**. Например:

```
%doc README TODO Changes
%doc Install
а не
%doc
README
TODO
Changes
Install
```

Еще раз отмечу, что наличие всех макрокоманд в файле спецификаций не является обязательным.

При создании пакета мы использовали опцию -bb программы **rpm**. При указании этой опции создается только двоичный RPM-пакет, если вы хотите создать также пакет, содержащий исходный текст программы, используйте опцию **-ba**. Созданный пакет помещается в каталог SRPMS и будет иметь имя `port-1.0-99.src.`rpm. То есть вместо названия архитектуры будет указано, что данный пакет содержит исходный текст профаммы. Для создания такого пакета в каталоге SOURCES должны находиться исходные тексты программы.

Для полноты картины осталось рассмотреть опции менеджера rpm, которые используются для создания пакетов (см. табл. 19.35).

Опции менеджера пакетов rpm *Таблица 19.35*

Опция	Описание
-ba	Создаются два пакета: двоичный и содержащий исходный текст. При этом не пропускается ни один этап установки, указанный в файле спецификаций
-bb	Создается только двоичный пакет. Не пропускается ни один этап установки, указанный в файле спецификаций
-bc	Выполняются этапы %pre и %build. При этом пакет распаковывается и компилируется
-bi	Выполняются этапы %pre, %build, %install
-bl	Выполняется проверка списка файлов, указанных в макрокоманде %files
-bp	Выполняется только этап %pre, то есть распаковывается архив
--recompile package.src.rpm	Указанный пакет, содержащий исходные тексты, сначала устанавливается, а потом компилируется
--rebuild package.src.rpm	Устанавливается и компилируется пакет исходных текстов, а затем создается новый двоичный пакет
--test	Проверка файла спецификаций
--clean	Удаление дерева каталогов BUILD после создания пакета
--showrc	Выводит файл конфигурации

19.6. Использование редактора vi

Согласно традиции, в состав любой Unix-подобной операционной системы входит текстовый редактор vi. Если вам придется работать с другим дистрибутивом, в нем может не оказаться предпочитаемого вами редактора, а вот **vi** есть всегда. Редактор vi помещается на загрузочную дискету, поэтому если вам придется редактировать системные файлы, загрузившись с системной дискеты для восстановления системы, особого выбора у вас не будет — только редактор vi. Поэтому давайте остановимся на нем подробнее.

Несмотря на свое название (vi — visual editor), vi является далеко не самым простым в использовании редактором. При работе в vi вы можете находиться в одном из трех режимов:

1. Командный режим.
2. Режим вставки.
3. Режим последней строки.

Первым режимом является режим командной строки, и именно в этом режиме вы находитесь, когда запускаете vi. В этом режиме вы можете вводить различные команды для работы с текстом.

В режиме вставки вы можете редактировать текст файла. Переход в режим вставки происходит при введении команды i (от insert) в командном

режиме. Для возврата из режима вставки в режим командной строки нажмите клавишу «Esc».

Режим последней строки является расширением командного режима. Команды, которые вы будете вводить в этом режиме, будут отображаться в последней строке экрана. Находясь в этом режиме, вы можете сохранить файл, выйти из **vi** или выйти без сохранения. Например, для сохранения файла используется команда **w**, а для выхода — команда q, если вы хотите выйти с сохранением файла, введите команду **wq**, а без сохранения — команду **q!**. Войти в режим последней строки можно, нажав **":"**.

Запустите редактор **vi** командой:

```
vi some_file.txt
```

Так как файла `some_file.txt` не существует, **vi** создаст новый файл с таким именем. В левой части экрана вы увидите столбец, состоящий из символов тильды «~». Этот столбец сообщает о том, что вы находитесь в конце файла: и это не удивительно, поскольку только что созданный файл пуст. Курсор отображается в верхнем левом углу как символ подчеркивания «_».

Нажмите i для перехода в режим вставки и начинайте печатать. Перейти на новую строку вы можете с помощью клавиши «Enter», корректировать ввод можно клавишами «Backspace» и «Del». Для возврата в командный режим нажмите клавишу «Esc». В командном режиме вы можете использовать стрелки для перемещения по тексту.

Кроме команды i, для вставки текста вы можете использовать команду a. Эта команда вставляет текст после текущего положения курсора. Например, используя команду a, вы можете вставить текст между словами *first* и *second:*

```
first_second third
```

Команда o вставляет текст в строку, которая находится ниже текущей. Для удаления текста в режиме команд используются команды: x, dd, **dw**. Первая удаляет символ перед курсором, вторая удаляет целые строки, а dw — слово, на котором находится курсор.

Для замены текста используется команда R. Это не автоматическая замена одного фрагмента текста на другой, как в других текстовых редакторах. Имеется в виду режим замены, который в обыкновенных редакторах включается с помощью клавиши «Insert».

Редактор **vi** разрабатывался таким образом, чтобы его можно было использовать на алфавитно-цифровых терминалах, на клавиатуре которых были только алфавитно-цифровые клавиши. На таких терминалах для перемещения по тексту используются клавиши h, j, k, и l для перемещения влево, вниз, вверх и вправо соответственно. Команда w переместит курсор на начало следующего слова, а b — на начало предыдущего. Команды 0 (ноль) и $ используются для перемещения в начало и конец текущей строки. Комбинация клавиш Ctrl+F переместит вас на один экран вперед, а Ctrl+B — на один экран назад. Для перемещения на определенную строку используйте номер этой строки и символ G. Например, команда **1G** переместит вас на первую строку файла, а команда G (без номера) — в конец файла.

Для выхода из редактора **vi** без сохранения изменений, перейдите в режим последней строки, введя символ **":"**, а затем введите команду **q!**. Для записи изменений введите команду **w**, для выхода и сохранения — команду **wq**.

Если вы хотите редактировать другой файл без выхода из **vi,** введите команду **e filename** в режиме последней строки. Перед этим нужно сохранить текущий файл или использовать команду **e! filename** для загрузки нового файла без сохранения изменений в текущем файле.

Добавить к редактируемому файлу другой файл позволяет команда г **filename** в режиме последней строки. Выполнить команду интерпретатора команд операционной системы можно с помощью команды **:r! команда,** например, :r! **ls** /etc. Тогда результат выполнения этой команды будет добавлен к текущему файлу.

19.7. Интерпретатор команд bash

Интерпретатор команд — это программа, выполняющая команды пользователя. Стандартным интерпретатором (или оболочкой) является **bash** (Bourne Again Shell). Достаточно распространенными также являются следующие интерпретаторы: **sh, ash, bsh, tcsh, csh, zsh.** Список установленных в вашей системе оболочек находится в файле /etc/shells. Команды оболочки можно вводить в командной строке, а можно оформить в виде сценария. Сценарий — это файл, содержащий команды оболочки. Создайте обыкновенный текстовый файл и сделайте его исполнимым. Система выполнит указанную последовательность команд. Для того, чтобы система узнала, какую оболочку нужно использовать, первая строка сценарий должна содержать полное имя сценария.

Например:

```
#!/bin/bash
```

Между символами # и ! не должно быть пробелов. Для обработки сценария вы можете использовать любую программу (естественно, она должна понимать синтаксис файла), а не только указанную в файле /etc/shells. Например, вы можете написать:

```
#!/usr/bin/my_proga
```

Оболочка при этом запустит программу /usr/bin/my_proga и передает имя файла сценария в качестве параметра. Если вы напишете:

```
#!/usr/bin/my_proga -f
```

то оболочка выполнит команду:

```
#!/usr/bin/my_proga -f <имя_сценария>
```

Создадим небольшой сценарий, который будет очищать экран и выводить на него ваше имя. Имя можно передать как параметр. Назовем наш сценарий **test:**

```
#!/bin/bash
# Это комментарий
clear
echo $1
```

Теперь рассмотрим все по порядку. С первой строкой, я думаю, все ясно. Вторая строка очищает экран. Третья строка выводит информацию, которая передана сценарию как первый (1) параметр. Запустите сценарий следующим образом:

```
./test Vasya Pupkin
```

На экране будет напечатан первый параметр, то есть слово *Vasya.* Вы можете немного изменить сценарий, чтобы он выводил оба параметра:

```
echo $1 $2
```

Если вы хотите передать фамилию, имя и отчество, то нужно использовать следующую команду:

```
echo "$1"
```

При этом не нужно явно указывать три параметра, просто интерпретатор не будет использовать пропуск для разделения параметров и все, что вы передадите ему, будет считаться одним параметром. При этом, если вы запустите сценарий с параметром *Vasya Pupkin,* на экране увидите *Vasya Pupkin.* А если укажете параметр *Ivanov Ivan Ivanovich*, сценарий так и напечатает *Ivanov Ivan Ivanovich.*

19.7.1. Каналы и списки

Материал этого и следующего пункта дополняет гл. 5, в которой рассматривалось перенаправление ввода/вывода. Поэтому я не буду подробно описывать сам механизм этих процессов, а ограничусь лишь несколькими примерами, чтобы напомнить вам гл. 5.

С помощью каналов вы можете перенаправить стандартный вывод одной программы на стандартный ввод другой. Например:

```
cat /var/log/secure | less
ps -ax | grep "$UID" | less
```

В первом случае стандартный вывод команды cat (содержимое файла /var/log/secure) перенаправляется на стандартный ввод программы **less,** которая обеспечивает поэкранный вывод информации. Вторая строка выводит список всех процессов, принадлежащих данному пользователю. Первая команда ps-ax выводит список всех запущенных в системе процесов, вторая (grep **"$UID"**) ищет фрагмент текста, содержащий идентификатор пользователя (UID) и выводит результат на стандартный вывод, то есть на стандартный ввод команды less. **$UID** является переменной окружения, которая содержит идентификатор пользователя. О переменных окружения поговорим немного позже.

Интерпретатор позволяет указывать списки команд в командной строке. Например:

```
lpr file.txt; lpq
```

При этом сначала выполняется команда постановки задания в очередь печати, а потом проверяется состояние принтера. Теперь более сложный пример:

```
ps -ax | head -n 1; ps -ax | grep httpd
```

Сначала выполняется команда **ps** для печати заголовка таблицы, а потом — для вывода информации о демоне **httpd.**

Можно использовать операции конъюнкции и дизъюнкции, например:

```
command1 && comand2
command3 || command4
```

Команда *command2* будет выполнена в случае успешного завершения команды *commandl* (возвратный код равен 0). Команда *command4* будет выполнена, если код возврата команды *command3* не равен 0. Самый простой пример — создание и изменение каталога:

```
mkdir mydir && cd mydir
```

Обычно интерпретатор bash выполняет команды в синхронном режиме, то есть после запуска программы ожидает ее завершения. Однако можно запустить программу и в асинхронном режиме, то есть без ожидания ее завершения. Для этого нужно использовать символ «**&**» в конце команды, например:

```
program &
```

19.7.2. Перенаправление ввода/вывода

Перенаправление ввода/вывода уже рассматривалось в гл. 5, поэтому я лишь напомню общий формат команд:

```
команда > (>>) файл
список > (>>) файл
```

Как вы уже знаете, при использовании одного знака больше файл, в который переназначен вывод, будет перезаписан, а при использовании двойного больше информация будет добавлена в конец файла. При использовании списка команд список нужно взять в фигурные скобки:

```
{date; free; who; } > logfile
```

Перенаправление ввода/вывода может быть использовано и в обратном направлении. Например, для печати списка URL достаточно выполнить команды:

```
lpr << URL
http://www.linux.ru
http://linux.ru.net
http://www.linux.org
URL
```

19.7.3. Подоболочки

При написании сценариев вы можете использовать механизм подоболочек. Если вы знакомы с каким-нибудь языком программирования, то должны знать об области распространения действия переменных. Существуют глобальные и локальные переменные. Первые действуют во всей программе, а вторые только внутри определенного блока, например, функции. Если в локальной функции определена переменная с таким же именем, что и одна из глобальных переменных, в этом локальном блоке будет использоваться значение локальной переменной. Далее приведен простейший пример программы, использующей локальные и глобальные переменные на языке Pascal:

```
Program Test;
Var A : integer; {глобальная переменная}
```

```
procedureGetLocal;
var A : integer; {локальная переменная}
begin
A:=10;
writeln(A);
end;

begin
A:=7;
writeln(A);

GetLocal(A);
writeln(A);
end;
```

При запуске программа выведет на экран:

```
7
10
7
```

B bash наблюдается нечто подобное. Этот блок называется подоболочкой. Если список команд заключен в фигурные скобки, то он выполняется в текущей оболочке, а если в обыкновенные, то в подоболочке. Итак, рассмотрим выполнение почти аналогичных сценариев:

```
#!/bin/bash
# Сценарий 1

NUM="one";   (NUM="two"; echo $NUM;);
echo $NUM

#!/bin/bash
# Сценарий 2
NUM="one";   {NUM="two"; echo $NUM;};
echo $NUM
```

Сценарий 1 выведет на экран следующую информацию:

```
two
one
```

а сценарий 2:

```
two
two
```

С помощью механизма подоболочек вы можете создавать более гибкие сценарии. С его помощью, например, можно временно изменять рабочий каталог:

```
pwd; (cd /tmp; pwd;); pwd
```

Сценарий выведет на экран:

```
/home/user
/tmp
/home/user
```

19.7.4. Переменные и массивы

Пример простейшей переменной мы уже рассмотрели выше. Переменные в **bash** не нуждаются в предварительном описании, как в других языках, например, в том же Pascal. Все переменные в bash — текстовые. Имя переменной должно начинаться с буквы и может состоять из латинских букв, цифр, знака подчеркивания. Нельзя определять функцию и переменную с одинаковыми именами. Чтобы воспользоваться значением переменной, нужно использовать символ доллара перед именем переменной:

```
VAR="my var"
echo $VAR
```

Как я уже говорил, все переменные текстовые. Например, когда вы присваиваете переменной значение **VAR=13**, то это будет не числовое значение, а строка из двух символов. Если присваиваемое значение имеет пробелы, его нужно взять в кавычки:

```
VAR="value 1"
```

Присвоить значение переменной можно с помощью встроенной команды **read:**

```
echo -n "Enter value: "
read x
echo $x
```

Параметр -n команды **echo** не выводит символ новой строки в конце сообщения *«Enter value:»*, то есть не переводит строку. Команда **read** читает значение, введенное пользователем с клавиатуры, и записывает его в переменную x. Последняя команда выводит только что введенное значение. При использовании команды **read** можно указывать несколько переменных:

```
read a b c
```

Пользователь должен ввести значения переменных, разделяя их пробелами. Для окончания ввода нужно нажать «Enter». Если введено меньше значений, чем нужно, оставшимся переменным будет присвоена пустая строка. А если больше, то лишние значения будут проигнорированы.

Интерпретатор **bash** использует следующие метасимволы, имеющие для него особое значение:

```
* ? ; & ( ) | ^ < > <возврат_каретки> <табуляция> <пробел>
```

Для того, чтобы использовать эти символы как они есть, нужно их цитировать с помощью символа \. Например, символ перевода строки можно цитировать как **\n**, символ табуляции как \t, а символ вопроса как \?

Особое значение при присваивании переменным значений имеют кавычки. Все символы, заключенные в одинарные кавычки ' ' представляют самих себя. Между двойными кавычками “ ” выполняются команды и подстановки значений. Символы “\”, “,”, “ ' ”, “ $ “ могут цитироваться с помощью обратной наклонной черты: \\, \$, \’

Для обозначения параметров командной строки используются специальные переменные, указанные в табл. 19.36.

Специальные переменные *Таблица 19.36*

Название переменной	Значение
$0	Имя выполняемой команды. Для сценария — путь, указанный при его вызове. Если вы знакомы с языком программирования Pascal, эта переменная должна вам напоминать вызов функции ParamStr(O)
$п (где п - номер параметра, $1, $2, ...)	Обращение к параметру с номеров п. Для Pascal — это вызов функции ParamStr(n)
$#	Число параметров, которые были указаны при вызове сценария. Аналогична вызову функции ParamCount в Pascal
$*	Все параметры, заключенные в кавычки: "$1 $2 ..."
$@	Все параметры, каждый из которых заключен в кавычки "$1" "$2" ...
$?	Код завершения последней команды
$$	Номер текущего процесса (PID)
$!	Номер (PID) последнего асинхронного процесса (команды, выполненной в фоновом режиме)

Пример.

```
echo "Все параметры: "
echo $*
```

Интерпретатор **bash** поддерживает одномерные массивы с неограниченным числом элементов. В других оболочках существуют определенные ограничения на массивы, например, в ksh максимальное число элементов массива ограничено 1024-мя элементами.

Присвоить значение элементу массива можно с помощью такой конструкции:

```
Имя_массива[индекс]=значение
```

Например:

```
A[1]=23
A[3]=54
A[0]=77
```

Нумерация элементов начинается с ноля. Тип элементов массива, как и тип переменных, текстовый. Присвоить значение элементам массива можно также с помощью инструкции **set.** Например, выражение:

```
set -A array 3 56 77 12
```

аналогично выражениям:

```
array[0]=3
array[1]=56
array[2]=77
array[3]=12
```

Вышеприведенные способы инициализации массивов могут применяться как в **bash,** так и в **ksh,** а также и в других оболочках. Но, тем не менее, • существует еще один способ, который работает только в **bash:**

```
array = ([0]=3 [3]=12 [2]=77 [1]=56)
```

Обратиться к значению элемента можно следующим образом:

```
${имя_массива[индекс]}
```

Например, вывести значение первого (нулевого) элемента массива можно так:

```
echo {$array[0]}
```

Обратиться ко всем элементам массива сразу можно с помощью одного из выражений:

```
${имя_массива[*]} или ${имя_массива[@]}
```

Если значение хотя бы одного элемента массива может содержать пробелы, используйте второе выражение:

```
echo ${array[@]}
```

Особое значение имеют переменные окружения. У каждого процесса есть своя среда — множество доступных ему переменных. Обыкновенные переменные доступны только из локального процесса. Чтобы использовать их значения в порождаемых процессах, нужно использовать встроенную команду **export.** После того, как вы экспортируете переменные, они будут доступны всем порожденным процессам.

Выражение export:

```
export имя
export имя=значение
export имя имя имя ...
export имя=значение имя имя=значение ...
```

Как видите, можно экспортировать уже инициализированную переменную (которой уже присвоено значение), а можно выполнить инициализацию переменной непосредственно при экспорте. Экспортировать можно как одну переменную, так и целый список.

Для деактивизации переменной используется команда **unset:**

```
unset имя
```

Каждому процессу доступны переменные оболочки, приведенные в табл. 19.37.

Переменные оболочки *Таблица 19.37*

Название переменной	Значение
PWD	Текущий каталог
UID	Идентификатор пользователя, запустившего сценарий
REPLY	Последняя строка, введенная с помощью команды read
RANDOM	Случайное число в диапазоне от 0 до 32767
SECONDS	Число секунд, прошедшее с момента запуска оболочки
IFS (Internal Filed Separator)	Внутренний разделитель полей. Используется синтаксическим анализатором и командой read для разделения строчки на слова. По умолчанию его значение равно " \t\n", где: « » — пробел; «\t» — символ табуляции; «\n» — символ новой строки
HOME	Домашний каталог
PATH	Путь вызова
LOGNAME	Имя пользователя, которое использовалось для входа в систему
MAIL	Имя файла, в который поступает электронная почта
SHELL	Имя интерпретатора команд
TERM	Тип терминала пользователя

Пример.

```
echo "$HOME"
```

19.7.5. Подстановка команд и арифметических выражений

В гл. 13 (п. 13.5) мы уже сталкивались с подстановкой команд. Тогда переменной DT присваивался результат выполнения команды **date:**

```
DT='date'
```

Как я уже писал, при подстановке команд нужно использовать обратные одинарные кавычки (они расположены под символом тильды на клавиатуре). Подставлять можно не только одну команду, а целые списки команд:

```
USERS='who I wd -l'
UP='date; uptime'
```

В первом случае мы получим количество пользователей работающих в системе, а во втором — последовательно записанные результаты выполнения команд **date и uptime.**

Подставлять результаты выполнения можно не только в переменные, а и в другие команды, например:

```
grep 'id -un' /etc/passwd
```

Данная команда ищет в файле /etc/passwd вхождение результата выполнения команды **id -un**

Подстановка арифметических выражений осуществляется с помощью конструкции

```
$(( выражение ))
```

Например:

```
A = $(( (10+5)/2 ))
e.cho $A
```

При этом на экране вы увидите число 7, а не 7.5, потому что используется целочисленное вычисление. Пример. Количество часов, прошедшее с момента запуска оболочки:

```
hrs = $(( $SECONDS/3600 ))
```

19.7.6. Управляющие структуры и циклы

К управляющим структурам относятся:

- Конструкция **if-fi.**
- Конструкция **case-esac.**

Конструкция if-fi

Общий синтаксис конструкции **if-fi:**

```
if список1 then
    список2
elif список3 then
    список4
else
    список5
fi
```

Конструкция **if-fi** работает так же, как и в других языках программирования. Если *список1* (условие) истинный, выполняется *список2*, иначе выпол-

няется *список3* и проверяется его истинность и т.д. Допускается неограниченная вложенность операторов **if.** Например:

```
if uuencode myfile myfile > myfile.uu; then
    echo "Успешное конвертирование";
else
    echo "Ошибка";
fi
```

Можно использовать сокращенный вариант:

```
if список1 then
    список2
fi
```

Например:

```
if [ $? -ne 0 ]; then echo "Ошибка. См. файл протокола"; fi;
```

Вместо списка команд удобно использовать команду **test** или выражение [условие]. Например, следующие выражения аналогичны:

```
test -e /etc.passwd
[-e /etc/passwd]
```

И первое, и второе выражение проверяют существование файла /etc/passwd. Другие опции команды **test** представлены в табл. 19.38.

Опции команды test *Таблица 19.38*

Опция	Возвращаемое зничение и описание
-d файл	Истина, если файл существует и является каталогом
-e файл	Истина, если файл существует
-f файл	Истина, если файл существует и является простым файлом
-k файл	Истина, если файл существует и для него установлен бит односторонней операции
-L файл	Истина, если файл существует и является ссылкой
-r файл	Истина, если файл существует и доступен для чтения
-s файл	Истина, если файл существует и его размер больше 0
-x файл	Истина, если файл существует и является исполнимым
-w файл	Истина, если файл существует и доступен для записи
-o файл	Истина, если файл существует и принадлежит данному пользователю
-z строка	Истина, если длина строки равна 0
-n строка	Истина, если длина строки не равна 0

Команда **test,** в случае успешного завершения, возвращает значение истина, то есть 0 — успешное завершение. Если в скобках стоит непустое слово, **test** возвратит тоже 0, например:

```
[ word ]; echo $?
0
[]; echo $?
1
```

В первом случае возвращается истина (true), на экран выводится ноль - код удачного (безошибочного) завершения программы. Во втором случае на экран выводится единица — команда **test** возвратила значение ложь (false).

Сравнение строк осуществляется следующим образом: выражения **str1 = str2** или **strl = = str2** истинны, когда строки strl и str2 равны. Обратите внимание: между двумя символами равно не должно быть пропуска!

Символ ! инвертирует любое условие команды **test,** например, выражение **str1 != str2** будет истинным, когда строки *str1* и *str2* не равны между собой. Символ ! является символом логической операции NOT (отрицание). Кроме этого символа, можно использовать опции команды -o и -a, которые обозначают логические операции ИЛИ (OR) и И (AND). Например:

```
str="word"; export str; ["$str" -a -f  /etc/passwd]; echo $?
0
str=""; export str; ["$str" -a -f  /etc/passwd]; echo $?
1
```

В первом случае непустая строка *str* возвращает истину, опция *—f* возвращает также истину, потому что файл /etc/passwd существует всегда. Результат операции И: *истина* И *истина* = *истина,* поэтому на экране вы увидите 0.

Во втором случае пустая строка *str* возвратит ложь, а опция *—f* возвращает истину. Результат операции И: *ложь* И *истина* = *ложь.* Если вы забыли законы логики, освежите свои знания с помощью табл. 19.39.

Логические операции *Таблица 19.39*

AND	True	False	OR	True	False	XOR	True	False
True	True	False	True	True	True	True	False	True
False	False	False	False	True	False	False	True	False

Операция XOR — это **исключающее ИЛИ.** Данная операция не используется при создании сценариев с помощью интерпретатора **bash.**

Для сравнения целых чисел используются опции команды **test,** приведенные в табл. 19.40.

Сравнение целых чисел *Таблица 19.40*

Опция	Описание
-eq	Равно
-ne	Не равно
-lt	Меньше
-gt	Больше
-le	Меньше или равно
-ge	Больше или равно

Интерпретатор **bash** воспринимает строки, как целые числа. Если нужно обнулить строку, то это достигается таким присваиванием: x=0.

Пример.

```
x=124 ; export x ; [ 111 -lt "$x" ] ; echo $?
0
```

Поскольку 111 меньше, чем 124, на экране вы увидите 0 (истина).

Примечание.

Во всех примерах, вы, наверное, заметили использование команды ***export****. Это необходимо для того, чтобы порожденному процессу (не забывайте:* ***test*** *— это отдельная программа) переменная x была доступна.*

Теперь, когда мы уже знакомы с конструкциями **test** и **if,** рассмотрим небольшой пример, демонстрирующий вложенность операторов if и использование команды **test.** Пример приведен в листинге 19.2.

Листинг 19.2. Пример вложенности операторов

```
echo -n "Какую оценку ты получил сегодня по программированию? "
read x
  if [ $x = 5 ]
  then echo  "Отлично !"
  elif [ $x = 4 ]
     then echo "Хорошо"
     elif [ $x = 3 ]
        then echo  "Удовлетворительно"
        elif [ $x = 2 ]
           then echo  "Надо бы пересдать"
           else echo "Как вообще можно было получить такую оценку???"
     fi
```

Если вы введете 5, сценарий отобразит на экране слово «Отлично», при вводе 4 вы увидите слово «Хорошо» и так далее. Если вы введете О, 1 или число больше пяти, вы увидите на экране последнюю фразу: «Как вообще можно было получить такую оценку???».

Конструкция case-esac

Конструкция выбора (case — выбор) имеет следующий синтаксис:

```
case значение in
 шаблон1) список1 ;;
 ...
 шаблонN) списокN ;;
esac
```

Рассмотрим сценарий (см. листинг 19.3), аналогичный сценарию 19.2, но использующий конструкцию **case** вместо **if.**

Листинг 19.3. Пример использования оператора case

```
echo -n " Какую оценку ты получил сегодня по программированию?
\
      read x
      case $x in
     5) echo  "Отлично !" ;;
     4) echo "Хорошо" ;;
     3) echo "Удовлетворительно" ;;
     2) echo "Надо бы пересдать" ;;
     *} echo  "Как вообще можно было получить такую оценку???"
;;
      esac
```

Работа сценария аналогична первому сценарию: при вводе оценок 2, 3, 4, 5 будут отображены соответствующие сообщения, а во всех остальных случаях — последнее сообщение.

Примечание.

*Структура оператора **case** больше напоминает структуру оператора **case** в языке Pascal, чем в языке C. Последняя строка выбора с шаблоном *) будет выбрана, когда не произойдет ни одного совпадения с ранее указанными шаблонами. Если же произошло совпадение с шаблоном **шаблонN**, то будет выполнен список **списокN**. После выполнения списка команд **списокN** будет произведен выход из структуры **case** — так же как и в Pascal. В языке C наблюдается нечто другое: если будет обнаружено совпадение, скажем с шаблоном 3, то будут выполнены последовательности операторов 3, 4, 5, ... N. Чтобы прервать выполнение блока **case** в языке C нужно использовать оператор **break. В bash** же такого нет.*

Если для одного списка команд нужно описать два или более шаблонов, используется символ | (OR).

```
case num in
    1|2|3) echo "1 or 2 or 3      ;;
    4|5) echo "4 or 5" ;;
    *) echo "other num"  ;;
esac
```

Циклы

Интерпретаторы **bash** и **ksh** поддерживают циклы **for, while, until, select,** а интерпретатор **sh** только **for** и **while.**

Синтаксис цикла for:

```
for имя_переменной in   список1
do
   список2
done
```

Простой пример:

```
for i in 1 2 3 4 5; do echo $i; done
```

На экране вы увидите:

```
1 2 3 4 5
```

Еще раз напомню, что любой список в **bash** должен заканчиваться точкой с запятой. Начинающие «программисты» делают много ошибок, связанных именно с этой особенностью списков. Пример использования: построчно вывести содержимое файла /etc/passwd вы можете с помощью такого цикла:

```
for str in 'cat /etc/passwd'
do
    echo "$str";
done
```

Цикл **for** закончит свою работу, когда будет обработан последний элемент списка, в данном случае, когда на экран будет выведена последняя строка файла /etc/passwd.

Синтаксис цикла **while:**

```
while список1
do
   список2
done
```

Цикл **while** будет выполняться, пока условие, заданное в списке *список1*, будет истинным. Поэтому цикл **while** иногда называют **циклом с истинным условием.** Например,

```
x=1
while [$x -lt 10]
do
 echo $x
 x = $(( $x + 1 ))
done
```

На экране вы увидите: 1 2 3 4 5 6 7 8 9

Когда переменная x примет значение 10, цикл завершит свою работу, так как программа **test** вернет значение *false* (x уже не меньше, а равен 10).

Цикл **until** (до) имеет похожую структуру, но выполняется несколько иначе:

```
until список1
do
    список2
done
```

Цикл **until** прекратит работу, когда условие, указанное в списке *список1*, станет истинным. Другими словами, он будет выполняться пока это условие ложно. Цикл while, наоборот, выполняется пока условие истинно. Лучше всего разница между этими циклами видна на примере (сравните листинги 19.4 и 19.5)

Листинг 19.4. Цикл while

```
x=1;
while ! [$x -де 10]
do
    echo $x
    x = $(( $x + 1 ))
done
```

Листинг 19.5. Цикл until

```
x=1;
until [$x -ge 10]
do
    echo $x
    x = $<( $x + 1 ))
done
```

Циклы, приведенные в листингах 19.4 и 19.5, выведут одинаковую последовательность цифр на экран:

1 2 3 4 5 6 7 8 9 1 0

Рассмотрим еще один полезный цикл **select,** который позволяет создавать нумерованные пункты меню. Его конструкция такова:

```
select имя in список1
do
    список2
done
```

Пример:

```
echo "Выберите файл для просмотра
select file in /home/den/temp/* QUIT
do
 if [-e $file]; then less $file
else
break
done;
```

В моем временном каталоге /home/den/temp находится всего два файла — file.txt, proto.txt, поэтому на экране монитора будет отображено следующее:

```
Выберите файл для просмотра:
/home/den/temp/.
/home/den/temp/..
/home/den/temp/file.txt
/home/den/temp/proto.txt
QUIT
```

Первые два файла — это ссылки на текущий и родительский каталоги. Пункты меню 3 и 4 — это файлы, которые можно выбрать для просмотра. QUIT — это последний элемент списка. При его выборе сработает оператор **break** и цикл завершится.

19.7.7. Подстановка переменных

Мы уже рассмотрели подстановку команд, сейчас рассмотрим подстановку переменных (см. табл. 19.41).

Подстановка переменных *Таблица 19.41*

Конструкция	Описание
${переменная:-значение}	Если переменная определена и не является пустой строкой, подставляется ее значение, иначе подставляется значение, указанное в конструкции. Реальное значение переменной при этом не изменяется
${переменна:=значение}	Значение присваивается переменной, если она не определена или является пустой строкой
${переменная:?сообщение}	Если переменная не определена или является пустой строкой, выводится указанное сообщение
${переменная:+значение}	Если переменная инициализирована (определена), вместо нее используется указанное в конструкции значение. Реальное значение переменной не изменяется
${переменная}	Если переменная определена, то подставляется ее значение. Скобки используются лишь тогда, если после переменной стоит символ, который может «приклеиться» к имени переменной

Пример.

```
${1 :? "Не хватает параметра"}
```

Данное сообщение будет выведено, если сценарий будет запущен без параметров. Если указать хотя бы один параметр, сообщение не будет отображаться на экране.

19.7.8. Функции

Описание функции выглядит так:

```
имя() { список; }
```

Пример:

```
cdir()
{
# изменяем каталог
cd /
}
```

При выполнении функция не создает нового процесса, а выполняется в среде процесса, содержащего эту функцию. Аргументы функции можно передать ей как обыкновенные параметры при вызове сценария. Функции можно описывать в любом месте сценария, но вызов функции должен осуществляться только после ее описания. Возвращаясь к примеру, модифицируйте функцию:

```
#!/bin/bash
# файл fn
echo $$
cdir()
{
# изменяем каталог
echo "X=$X"
X=2
echo "Params $0 $# $1 $2"
echo "PID = $$"
return 0
cd $1
)
X=1
echo "X=$X"
cdir /etc   # вызов функции "cd" с параметрами
echo $?
echo "X=$X"
```

На экране вы увидите примерно следующую информацию:

```
788
X=1
X=1
Params fn 1 /etc
788
0
X=2
```

Проанализируем полученную информацию. Как уже отмечалось, функция не порождает нового процесса, поэтому PID остался равным 788 как до вызова функции, так и во время ее выполнения. Переменная X доступна нашей функции, потому что описана до вызова функции. Функция «видит» значение переменной X, установленное в основном блоке сценария. Затем функция изменяет значение переменной X и передает его в основной блок (X=2). Функции был передан только один параметр — /etc, вместо второго параметра была отображена пустая строка. Имя файла осталось прежним – fn. Обратите внимание на важный момент: функция сообщила нам много полезной информации об устройстве функций в **bash,** но не оправдала своего названия — **cdir** (change dir). Реально изменения каталога не произошло, потому что перед выполнением команды **cd** была выполнена команда **return** с кодом завершения 0, которая прервала выполнение функции.

19.7.9. Обработка сигналов и протоколирование

Возможно, вы хотите обеспечить выполнение вашего сценария после выхода пользователя из интерпретатора или выполнить какие-нибудь действия при отключении удаленного пользователя от системы. «Перехватить» сигнал (прерывание) можно с помощью команды **trap.** Формат команды **trap** следующий:

```
trap имя сигналы
```

где: имя - это имя функции или набор команд, которые должны быть выполнены при получении сигнала;

сигналы -- наиболее часто используется перехват сигналов, описанных в табл. 19.42. Полный список сигналов вы найдете в гл. 5.

Сигналы *Таблица 19.42*

Номер	Название	Описание
01	SIGHUP	Освобождение линии (hangup)
02	SIGINT	Прерывание (interrupt)
03	SIGQUIT	Выход (quit)
09	SIGKILL	Уничтожение процесса (kill). Не перехватывается и не игнорируется
15	SIGTERM	Программный сигнал завершения

Пример. Игнорирование сигналов 1, 2, 3, 15

```
trap : 1 2 3 15
```

: — это пустой оператор, не выполняющий никаких действий.

Рассмотрим, как можно протоколировать работу собственного сценария. Для этого существуют два способа -- с помощью команды tee и команды **script.**

Способ 1:

```
$LOGFILE=my_log
if ["$LOGGING" != "true"]; then export LOGGING="true"; exec $0
| tee $LOGFILE; fi
```

Способ 2:

```
$LOGFILE=my_log
if ["$LOGGING" != "true"]; then export LOGGING="true"; exec
script $0 $LOGFILE; fi
```

В первом случае мы устанавливаем флаг протоколирования LOGGING и заново запускаем наш сценарий. При этом перенаправляем весь стандартный вывод команде tee, которая выполнит протоколирование. Второй способ аналогичен первому за исключением того, что мы не будем самостоятельно запускать сценарий — это за нас выполнит команда **script.** Оба способа можно использовать для протоколирования работы других программ:

```
script program ~/program.log
```

20

Графический интерфейс пользователя. Система X Window

Система X Window является мощной графической средой для UNIX-станций. Данная система была разработана Массачусетским технологическим институтом (MIT) и стала стандартом для всех UNIX-систем. Практически каждая рабочая станция UNIX работает на одном из вариантов системы X Window.

Группа программистов, возглавляемая Дэвидом Вексельблатом (David Wexelblat) создала свободно распространяемую версию MIT X Window для процессоров i80386-Pentium IV и совместимых с ними. Эта версия получила название **XFree86**, поскольку могла выполняться в операционных системах, предназначенных для процессоров, использующих систему команд x86 — Linux, FreeBSD и другие. XFree86 является торговой маркой XFree86 Project, Inc.

Данная глава включена в книгу последней, так как графический интерфейс на сервере — это излишество. Зачем тратить драгоценные системные ресурсы на графический интерфейс, если хороший администратор подходит к серверу очень редко, а иногда вообще один раз — во время первоначальной настройки? Однако, если вы начинающий администратор, графический интерфейс поможет вам быстрее настроить тот или иной сервис. Может случиться такое, что вы установите новую версию дистрибутива и окажется, что расположение некоторых системных файлов будет изменено. Чтобы не тратить время на прочтение документации, вы можете использовать один из конфигураторов, например, netconf или **linuxconf**. Графический конфигуратор уж точно знает, где и что лежит. Но применение графического интерфейса только для конфигураторов не оправдано, потому что упомянутые конфигураторы работают и в консоли.

Другая причина применения GUI (Graphic User Interface — графический интерфейс пользователя) - удобство: браузер Netscape удобнее, чем lynx, хотя и к последнему можно привыкнуть, если в нем часто работать. Обычно, если позволяют системные ресурсы, устанавливается система X Window, но не настраивается для автоматического запуска. Какой бы ни была причина установки вами системы X Window, если вы все-таки установили ее, то эта глава для вас. В этой главе мы рассмотрим конфигурирование X Window (далее X), а также работу в графических средах **KDE** и Gnome.

Двумя предложениями выше я упомянул, что установить X вы можете, если позволяют системные ресурсы. Какие же минимальные системные требования X? Как бывает со многими Linux-приложениями, здесь более критичен объем оперативной памяти, чем частота процессора. Intel Pentium III 600 MHz/32 MB работал при запущенном KDE в два раза медленнее, чем Celeron 433 MHz/64 MB. Для комфортной работы с KDE и многими X-приложениями необходимо 64 или даже 128 Мб оперативной памяти. Желательно иметь в запасе 64...128 Мб своп-пространства. Частота процессора — не менее 200 MHz. Как видите, если для работы сервера, например, шлюза, достаточно процессора Intel Pentium 133 MHz и 32 MB памяти, то при установке X минимальные системные требования возрастают. Я понимаю, что делаю из мухи слона, потому что сейчас минимальная конфигурация компьютера — Celeron 400 MHz /128 MB, но все это сказано для владельцев старых компьютеров. Например, если у вас где-нибудь завалялся старенький IP 166 MMX/32 MB, он еще может принести большую пользу, если использовать его как шлюз для доступа к Интернет целой сети предприятия. И все это будет работать намного быстрее, чем организованный на скорую руку шлюз на платформе Windows 98 + WinGate, даже При использовании Pentium III. Более мощные компьютеры можно загрузить, соответственно, и более ответственными и ресурсоемкими процессами под управлением окон — набором текстов в Word и просмотром MP4. Извините, я немного отвлекся. Я говорил о стареньком компьютере, который может нам еще пригодиться. Так вот, если не хотите испортить приятное впечатление от Linux, не устанавливайте на таком компьютере X Window. Помню свою первую попытку установить Linux + X Window. Тогда у меня был как раз IP 166MMX. Установить-то установил, и X настроил, а вот удовольствия от работы не было никакого. При использовании менее ресурсоемких оконных менеджеров, например, **fvwm** и **fvwm2**, все работало удовлетворительно, а вот KDE тогда еще самой первой версии жутко притормаживал.

20.1. Установка и запуск системы

Удобнее всего устанавливать систему X Window при установке операционной системы. Если вы еще не сделали этого, сейчас рассмотрим установку системы из пакетов. Обычно все необходимые пакеты находятся на первом инсталляционном диске Linux. Перейдите в каталог `/mnt/cdrom/Mandrake/RPMS`, если вы используете Mandrake, или `/mnt/cdrom/RedHat/RPMS`, если вы используете Red Hat Linux. Проще всего установить все пакеты сразу:

```
rpm -in XFree*
```

Для установки всех программ для X Window введите команду:

```
rpm -ih x*
```

Такая команда у вас будет работать, если вы не используете или еще не установили демон **xinetd**. В противном случае менеджер RPM-пакетов сообщит вам, что пакет xinetd уже установлен и установка всех пакетов будет прервана.

Естественно, такой вариант (вариант установки всех программ) вас не устраивает — уж больно он неэкономно относится к дисковому простран-

ству. Попробую перечислить необходимые пакеты для установки X Window и KDE. В любом случае вам нужно установить следующие пакеты:

XFree86-4.0.1-28mdk.i586.rpm — **основной пакет**
XFree86-libs-4.0.1-28mdk.i586.rpm — **библиотеки**
XFree86-server-4.0.1-28mdk.i586.rpm
XFree86-server-common-3.3.6-18mdk.i586.rpm
XFree86-xfs-4.0.1-28mdk.i586.rpm — **сервер шрифтов**
XFree86-glide-module-4.0.1-28mdk.i586.rpm — **модуль glide**
XFree86-FBDev-3.3.6-18mdk.i586.rpm
XFree86-Xvfb-4.0.1-28mdk.i586.rpm — виртуальный буфер кадров для X Window
XFree86-Xnest-4.0.1-28mdk.i586.rpm
XFree86-100dpi-fonts-4.0.1-28mdk.i586.rpm
XFree86-75dpi-fonts-4.0.1-28mdk.i586.rpm
XFree86-cyrillic-fonts-4.0.1-28mdk.i586.rpm
xinitrc-2.4.4-32mdk.noarch.rpm
XFree86-SVGA-3.3.6-18mdk.i586.rpm

Последний пакет является сервером для вашей видеоплаты. Данный сервер может работать с большинством видеоадаптеров SVGA. Если у вас другая видеоплата, например, Mach 8, 32, 64; AGX, S3, VIRGE, IBM 8514, установите соответствующий сервер. Этим вы обеспечите корректную работу всей системы. X Window. После установки X перейдем к установке KDE. Если ввести команду:

```
rpm -ih kde*
```

система установит много ненужных пакетов с документацией. Все пакеты **kde-il8n*** содержат одну и ту же документацию, но на разных языках — от бразильского до украинского. Данные пакеты также выполняют локализацию KDE. В любом случае вам нужно установить только один пакет из всех этих — kde-i18n-Russian-2.0-1mdk.i586.rpm. Устанавливать его нужно после остальных пакетов среды KDE.

Вот какие пакеты вам нужно установить:

1. kdebase-2.0-7mdk.i586.rpm — базовый пакет
2. kdelibs-2.0-5mdk.i586.rpm — библиотеки
3. kdelibs-sound-2.0-5mdk.i586.rpm — поддержка звука (необязательно)
4. kdeutils-2.0-3mdk.i586.rpm — всевозможные утилиты для KDE
5. kdesupport-2.0-lmdk.i586.rpm -- вспомогательные библиотеки (желательно установить)
6. kdepim-2.0-1mdk.i586.rpm
7. kdenetwork-2.0-1mdk.i586.rpm — сетевые приложения (например, kppp)
8. kdemultimedia-2.0-4mdk.i586.rpm — программы для работы с мультимедиа (необязательно)
9. kdegraphics-2.0-4mdk.i586.rpm — программы для работы с графикой (необязательно)
10. kdeadmin-2.0-2mdk.i586.rpm
11. kdeaddutils-2.0-3mdk.i586.rpm — вспомогательные программы
12. kde1-compat-1.1.2-7mdk.i586.rpm — библиотеки для совместимости с KDE версии 1 (необязательно)
13. kdegames-2.0-lmdk.i586.rpm — игрушки (необязательно)
14. kdetoys-2.0-1mdk.i586.rpm — приколы (необязательно)

15. kdesdk-2.0-1mdk.i586.rpm — KDE SDK (для разработчика)
16. kdelibs-devel-2.0-5mdk.i586.rpm — исходники библиотек KDE (необязательно)
17. koffice-2.0-2mdk.i586.rpm — офисный пакет K-Office

Если вы хотите установить также и оконную среду Gnome, просто введите команду:

```
rpm -in gnome*
```

Ничего лишнего (за исключением пары небольших пакетов) установлено не будет. Пока вы не установили ни одного графического средства конфигурирования X Window. Сейчас целесообразно ввести команду **xf86config**. Отвечая на вопросы программы, будьте предельно внимательны: неправильная установка некоторых параметров (например, частоты горизонтальной или вертикальной развертки) может сжечь ваш монитор. Вы можете установить конфигуратор **DrakConf** — более безопасное средство настройки X Window. Настройка системы с его помощью будет рассмотрена в пункте 20.3.

Установите символическую ссылку на сервер видеоплаты:

```
ln -sf /usr/XllR6/bin/XF86S3 /etc/X11/X
```

В этом примере я использовал сервер XF86S3. При использовании конфигуратора, например, **xf86config** или **Xconfigurator**, данная ссылка уже должна быть установлена, но иногда конфигуратор почему-то «забывает» установить ее.

Теперь перейдем непосредственно к настройке X. Как я уже отмечал, система X Window может запускаться автоматически при запуске Linux. Вопрос о режиме запуска X задается при установке системы — сразу после выбора графического режима. В любом случае, даже если вы пропустили этап настройки X во время установки системы, а саму систему X Window установили, то она уже должна быть настроена для работы. В этом случае устанавливается режим 640x480 и 8-битный цвет (256 цветов).

Если вы не установили автоматический запуск системы, запустить X можно из консоли, введя команду **startx.** После запуска X перейти в нужную консоль вы можете, нажав комбинацию клавиш Ctrl+Alt+Fn, где *n* — это номер нужной вам консоли. Перейти из консоли в X (если система X Window запущена) можно с помощью комбинации **Alt+F7**. Напомню, что для переключения между консолями используется комбинация Alt+Fn. Для «аварийного» выхода из X Window используйте комбинацию клавиш Ctrl+Alt+BackSpace.

После запуска системы с помощью команды **startx** вы можете выбрать оконную среду, в которой хотите работать. При автоматическом запуске системы регистрация пользователя будет происходить в графическом режиме. Если вы хотите зарегистрироваться в консоли, нажмите комбинацию клавиш Ctrl+Alt+F1.

На самом деле команда **startx** — это обыкновенный сценарий, который запускает программу **xinit.** Именно эта программа запускает систему X Window. По большому счету можно сразу запускать **xinit,** но это будет не очень корректно. Сценарий **startx** делает это правильно, установив должным образом параметры запуска. Вот что представляет собой сценарий **startx** (листинг 20.1).

Листинг 20.1. Сценарий startx

```
#!/bin/sh
# $XConsortium: startx.cpp,v 1.4 91/08/22 11:41:29 rws Exp $
# $XFree86: xc/programs/xinit/startx.cpp,v 3.2 1998/12/20 11:58:22 dawes Exp $
#
# (c) 1999 Red Hat Software, Inc.
userclientrc=$HOME/.xinitrc
userserverrc=$HOME/.xserverrc
sysclientrc=/etc/X11/xinit/xinitrc
sysserverrc=/etc/X11/xinit/xserverrc
clientargs=""
serverargs=""
if [ -f $userclientrc ]; then
    clientargs=$userclientrc
else if [ -f $sysclientrc ]; then
    clientargs=$sysclientrc
fi
fi
if [ -f $userserverrc ]; then
    serverargs=$userserverrc
else if [ -f $sysserverrc ]; then
    serverargs=$sysserverrc
fi
fi
display=:0
whoseargs="client"
while [ "x$1" != "x" ]; do
    case "$1" in
        /''*|\.*)       if [ "$whoseargs" = "client" ] ; then
                        if [ "x$clientargs" = x ]; then
                            clientargs="$1"
                        else
                            clientargs="$clientargs $1"
                        fi
                    else
                        serverargs="$serverargs $1"
                    fi;;
        -)          whoseargs="server" ;;
        *)          if [ "$whoseargs" = "client" ]; then
                        clientargs="$clientargs $1"
                    else
                        case "$1" in
                            :[0-9]) display="$1"
                            ;;
                            *) serverargs="$serverargs $1"
                            ;;
                        esac
                    fi ;;
    esac
    shift
```

```
done
# set up default Xauth info for this machine
mcookie=`mcookie`
serverargs="$serverargs -auth $HOME/.Xauthority"
xauth add $display . $mcookie
xauth add `hostname -f`$display . $mcookie
exec xinit $clientargs -- $display $serverargs
# various machines need special cleaning up,
# which should be done here
```

20.2. Конфигурационный файл XF86Config

Как и любая другая программа, система X Window имеет свой конфигурационный файл. Согласно традиции, конфигурационные файлы хранятся в каталоге /etc. Главный конфигурационный файл называется **XF86Config** и находится в каталоге /etc/X11. В этом файле указываются настройки всех устройств, необходимых для реализации графического интерфейса: видеоадаптера, монитора, мыши, клавиатуры, а также настройки шрифтов, которые использует система X Window. Пример моего файла приведен в листинге 20.2.

Внимание!

Не нужно использовать этот пример — у вас он может работать некорректно. Вы можете использовать его только в том случае, если конфигурации наших видеосистем совпадают: Riva TNT2 Vanta, Samsung SyncMaster 550s.

Листинг 20.2. Пример файла XF86Config

```
Section "Files"
    RgbPath "/usr/X11R6/lib/X11/rgb"
    FontPath   "/usr/X11R6/lib/X11/fonts/cyrillic"
    FontPath   "unix/:-1"
EndSection

Section "ServerFlags"
EndSection

Section "Keyboard"
    Protocol     "Standard"
    AutoRepeat   250 30
    LeftAlt          Meta
    RightAlt         Meta
    ScrollLock       Compose
    RightCtl         Control
    XkbDisable
    XkbKeycodes      "xfree86"
    XkbTypes         "default"
    XkbCompat        "default"
    XkbSymbols       "us(pc105)"
    XkbGeometry      "pc"
    XkbRules         "xfree86"
    XkbModel         "pc105"
```

```
    XkbLayout          ""
EndSection

Section  "Pointer"
    Protocol      "PS/2"
    Device        "/dev/psaux"
    Emulate3Buttons
    Emulate3Timeout     50
EndSection

Section  "Monitor"
    Identifier "My Monitor"
    VendorName "Samsung"
    ModelName  "SyncMaster 550s"
    HorizSync   30-61
    VertRefresh 50-120

ModeLine "1024x768" 75.00 1024 1048 1184 1328  768  771  777  806 -hsync -vsync
ModeLine "1024x768"   65.00 1024 1048 1184 1344  768  771  777  806 -hsync -vsync
# 1024x768, 70.0Hz; hfreq=56.476002, vfreq=70.069000
ModeLine "1024x768"   75.00 1024 1048 1184 1328  768  771  777  806 -hsync -vsync
# 1024x768, 75.0Hz; hfreq=60.022999, vfreq=75.028999
ModeLine "1024x768"   78.75 1024 1040 1136 1312  768  769  772  800 +hsync +vsync
# 1024x768, 75.0Hz; hfreq=60.022999, vfreq=75.028999
ModeLine "1024x768"   78.75 1024 1040 1136 1312  768  769  772  800 +hsync +vsync
# 1280x1024, 75.0Hz; hfreq=79.975998, vfreq=75.025002
ModeLine "1280x1024"  135.00 1280 1296 1440 1688 1024 1025 1028 1066 +hsync +vsync
# Далее следует очень длинный список различных режимов работы
# монитора — ModeLine
# Я не привел его в листинге
EndSection

Section  "Device"
    Identifier  "Generic  VGA"
    Chipset     "generic"
EndSection

Section  "Device"
    Identifier   "RIVA TNT2"
    VendorName   "Unknown"
    BoardName    "Unknown"
#     Chipset      "RIVATNT2"
#     VideoRam     8192
    Option       "power_saver"
EndSection

Section  "Screen"
    Driver "svga"
    Device       "RIVA  TNT2"
    Monitor      "  My Monitor  "
    DefaultColorDepth 32
    Subsection  "Display"
        Depth         8
        Modes         "1024x768"  "800x600"  "640x400"
        Viewport      0 0
```

```
    EndSubsection
    Subsection "Display" ,
        Depth          16
        Modes          "1024x768" "800x600" "640x480"
        ViewPort       O O
    EndSubsection
    Subsection  "Display"
        Depth          24
        Modes          "1024x768"  "800x600"  "640x480"
        ViewPort       O O
    EndSubsection
    Subsection  "Display"
        Depth          32
        Modes          "1024x768"  "800x600"  "640x480"
        ViewPort       O O
    EndSubsection
EndSection

Section "Screen"
    Driver "vga16"
    Device         "Generic VGA"
    Monitor       " My Monitor "
    Subsection  "Display"
        Modes          "640x480"  "800x600"
        ViewPort       O O
    EndSubsection
EndSection

Section "Screen"
    Driver "vga2"
    Device         "Generic VGA"
    Monitor       " My Monitor "
    Subsection  "Display"
        Modes          "640x480"  "800x600"
        ViewPort       O O
    EndSubsection
EndSection

Section  "Screen"
    Driver "accel"
    Device         "RIVA TNT2"
    Monitor       " My Monitor "
    DefaultColorDepth  32
    Subsection "Display"
        Depth          8
        Modes          "1024x768"  "800x600"  "640x400"
        ViewPort     . O O
    EndSubsection
    Subsection  "Display"
        Depth          16
        Modes          "1024x768"  "800x600"  "640x480"
        ViewPort       O O
    EndSubsection
```

```
    Subsection "Display"
        Depth       24
        Modes       "1024x768" "800x600" "640x480"
        Viewport    0 0
    EndSubsection
    Subsection "Display"
        Depth       32
        Modes       "1024x768" "800x600" "640x480"
        Viewport    0 0
    EndSubsection
EndSection

Section. "Screen"
    Driver "fbdev"
    Device      "RIVA TNT2"
    Monitor     " My Monitor "
    DefaultColorDepth 32
    Subsection "Display"
        Depth       8
        Modes       "default"
        Viewport    0 0
    EndSubsection
    Subsection "Display"
        Depth       16
        Modes       "default"
        Viewport    0 0
    EndSubsection
    Subsection "Display"
        Depth       24
        Modes       "default"
        Viewport    0 0
    EndSubsection
    Subsection "Display"
        Depth       32
        Modes       "default"
        Viewport    0 0
    EndSubsection
EndSection
```

В секции **Files** указаны основные каталоги, необходимые для работы X Window. Обратите внимание на строку:

```
FontPath    "/usr/X11R6/lib/X11/fonts/cyrillic"
```

После установки русских шрифтов эту строку в файл конфигурации нужно добавить самостоятельно, после чего перезапустить сервер X — нажать комбинацию Ctrl+Alt+Backspace и заново запустить сервер с помощью команды System. Если ваша система настроена на автоматический запуск, завершите сеанс в вашем оконном менеджере и в окне регистрации в системе нажмите на кнопку «System». После чего выберите **Restart X** Server и нажмите на кнопку «Ок».

Следующая секция — **ServerFlags**. В ней определяются глобальные переменные сервера. Часто эта секция пуста (все закомментировано).

В секции **Keyboard** конфигурируется клавиатура, а секция **Pointer** – мышь. В секции **Monitor** описываются параметры монитора. Один из них идентификатор — *My Monitor.* Значение этого параметра потом указывается в секции **Screen.** В секции **Screen** делаются ссылки на используемую видеоплату (секция Device) и на монитор, а также на используемые режимы работы монитора. Здесь же устанавливается глубина цвета:

```
Depth       32
```

20.3. *Настройка X Window*

Теперь перейдем к практической настройке X Window. Если вы используете Red Hat Linux, запустите утилиту setup. Вы также можете использовать **XF86Setup** (нужно установить отдельный пакет) или **Xconfigurator**. Этапы настройки системы X аналогичны во всех дистрибутивах: выбирается монитор и видеоадаптер, а потом устанавливается разрешение монитора. При использовании некоторых средств настройки X (например, **xf86config**), которые позволяют указать частоту монитора, не перестарайтесь, вы можете вывести монитор из строя, указав недопустимую частоту! Перед тем как использовать такие средства, прочитайте руководство пользователя монитора. Программа **xf86config** позволяет более точно настроить X, но вы должны знать, что делаете.

Дальше все рисунки будут соответствовать дистрибутиву Linux Mandrake. Сначала запускаем программу **XFdrake**. Вы также можете запустить программу Xconfigurator — это просто ссылка на XFdrake (см. рис. 20.1).

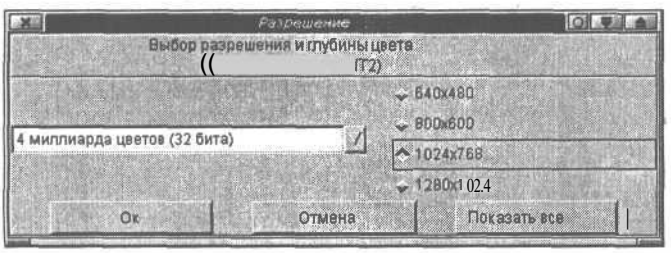


Рис. *20.1. Выбор разрешения*

Конфигуратор предложит вам выбрать разрешение и цветовую гамму. Выберите нужное вам разрешение и нажмите «Ок». Если вы хотите использовать все разрешения, нажмите на кнопку «Показать все». Тогда, чтобы увеличить разрешение нажмите комбинацию клавиш Ctrl+Alt+«+», а чтобы перейти к более низкому разрешению — Ctrl+Alt+«–». При этом нажимайте «+» и «–» на цифровой клавиатуре. Протестируйте настройки (рис. 20.2)

Следующий вопрос конфигуратора – хотите, чтобы X стартовал автоматически при перезапуске системы?

Рис. 20.2. *Тестирование настроек*

Вы можете использовать опцию Автовход, если хотите, чтобы происходила автоматическая регистрация пользователя в системе при запуске X. Из соображений безопасности не рекомендую этого делать. В этом же окне вы можете выбрать также и оконную среду, которая будет использоваться по умолчанию, например, KDE (см. рис. 20.4).

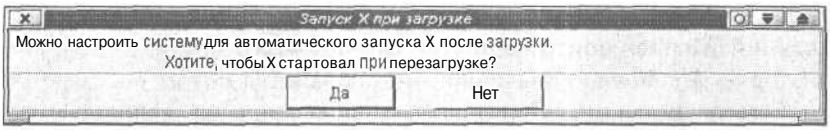


Рис. 20.3. *Автоматический запуск X Window*

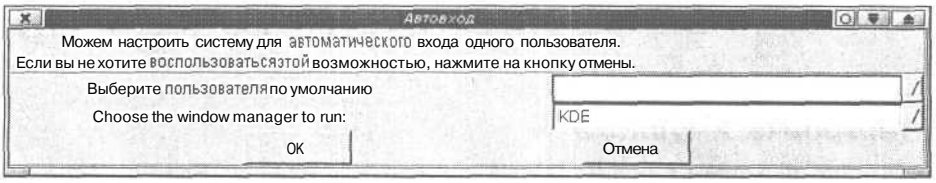


Рис. 20.4. *Автовход*

На этом настройка закончена. Но еще есть некоторые опции, которые вы можете установить. Можно изменить монитор, видеоплату, разрешение.

Для этого опять запустите **XFdrake** и нажмите на кнопку «Отмена». Появится окно, аналогичное окну на рис. 20.5.

Выберите режим изменить монитор (см. рис. 20.6).

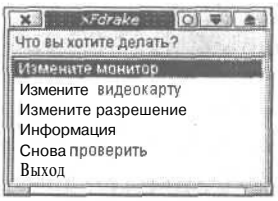


Рис. 20.5. *XFdrake*

Если вашего монитора не окажется в списке, выберите один из стандартных (Generic), который по характеристикам максимально приближен к вашему. Точно так же вы можете изменить видеоплату (см. рис. 20.7).

После выбора монитора и видеоплаты, нажмите на кнопку «Снова проверить». Если установленные параметры вас не устраивают, повторите настройку сначала.

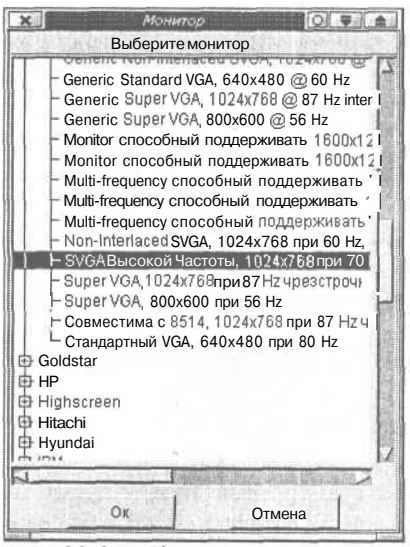


Рис. 20.6. *Выбор монитора*

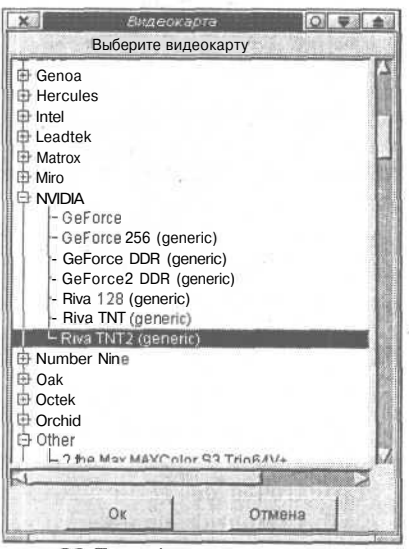


Рис. 20.7. *Выбор видеокарты*

20.4. Конвертирование шрифтов Windows

Вы уже знаете, как подключать шрифты. Для этого достаточно прописать путь к каталогу со шрифтами в файле **XF86Config** и перезапустить сервер X. Теперь разберемся, как использовать шрифты Windows. Наверняка, у вас есть некоторые шрифты в Windows, которые бы вы хотели использовать и в Linux. После конвертирования шрифтов вы сможете использовать шрифты **windows-ttf** обычным способом. Для этого запустите конфигуратор **DrakConf** и щелкните на пиктограмме **DrakFont** (или просто запустите программу **drakfont**). В появившемся окне конфигуратора (рис. 20.8) нажмите на кнопку «Взять шрифты Windows».

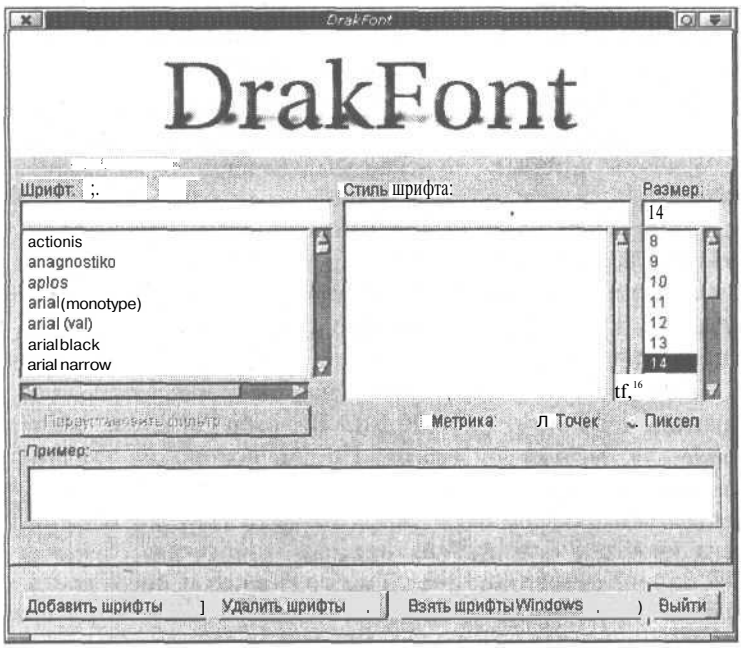


Рис. 20.8. *DrakFont*

После этого выберите интересующие вас шрифты и нажмите на одну из кнопок: «Установить все» или «Установить выбранные шрифты» (рис. 20.9).

Обратите внимание на появившуюся строчку в файле конфигурации XF86Config:

```
FontPath "/usr/X11R6/lib/X11/
fonts/win-ttf"
```

именно в каталог `/usr/X11R6/lib/X11/fonts/win-ttf` конфигуратор поместил шрифты windows после их конвертирования.

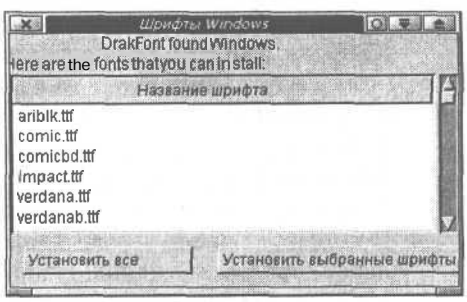


Рис. 20.9. *Шрифты Windows*

20.5. Оконная среда KDE

Теперь, когда мы уже практически настроили систему X Window, пора рассмотреть оконные среды. Обзор оконных сред я начну со своей любимой среды — **KDE**. Аббревиатура KDE означает K Desktop Environment. Обратите внимание на то, что KDE — это оконная среда, а не оконный менеджер. Последний просто предоставляет набор функций (API) для управления окнами системы X Window, а оконная среда — это набор программ, технологий и документации, которые являются попыткой сделать использование компьютера более простым.

По своей простоте и интуитивности среда KDE подобна графическим интерфейсам MacOS или Windows 9x. KDE предоставляет богатые возможности взаимодействия программ, сочетает в себе метод **LookNFeel** (Смотри и Чувствуй), а также обладает встроенным механизмом **drag-and-drop.**

Подробно рассматривать интерфейс пользователя KDE я не стану — уж довольно все просто и интуитивно понятно. Есть единственное замечание для пользователей Windows, которые привыкли дважды щелкать на объекте для его открытия: в KDE запуск объекта выполняется одним щелчком мыши. В Windows, как правило, один щелчок означает активизацию объекта, в два — его запуск. В KDE для активизации (выделения) достаточно подвести указатель мыши к объекту. Windows Explorer тоже можно настроить для работы в таком режиме, однако почему-то данный режим используют далеко не все пользователи.

При запуске KDE можно увидеть привычный нам со времен Windows 95 рабочий стол и панель KDE (см. рис. 20.10). На рабочем столе могут размещаться ярлыки для запуска программ. По умолчанию на главной панели KDE расположены четыре панели:

1. **Панель меню.** Эта панель чем-то напоминает панель быстрого запуска в Windows. На ней расположена кнопка с надписью K, предназначенная для открытия главного меню KDE. Затем следует стандартный набор кнопок: «Список окон», «Показать рабочий стол», «Персональный каталог», «Центр Управления», «Терминал», «Помощь», «Konqueror». Так же, как и в Windows, вы можете помещать на эту панель собственные ярлыки. В KDE ярлык называется конфигурационным файлом рабочего стола. Да простят меня разработчики KDE, в книге я буду использовать термин ярлык — проще и короче.
2. Следующая панель — **Виртуальные рабочие столы.** По умолчанию используется четыре рабочих стола — больше вам и не нужно, хотя можно настроить KDE для работы с большим количеством виртуальных рабочих столов. При этом следует учитывать, что каждый виртуальный рабочий стол — это дополнительные системные ресурсы.
3. Третья панель — **панель задач KDE.** На ней отображаются программы, запущенные на данном виртуальном рабочем столе. Правильнее будет сказать не программы, а окна, открытые на данном рабочем столе.
4. Последняя панель — это аналог панели SysTray в Windows. Стандартный набор программ, использующий эту панель -- **Klipper** (контроль буфера обмена), **Раскладка клавиатуры и Часы.**

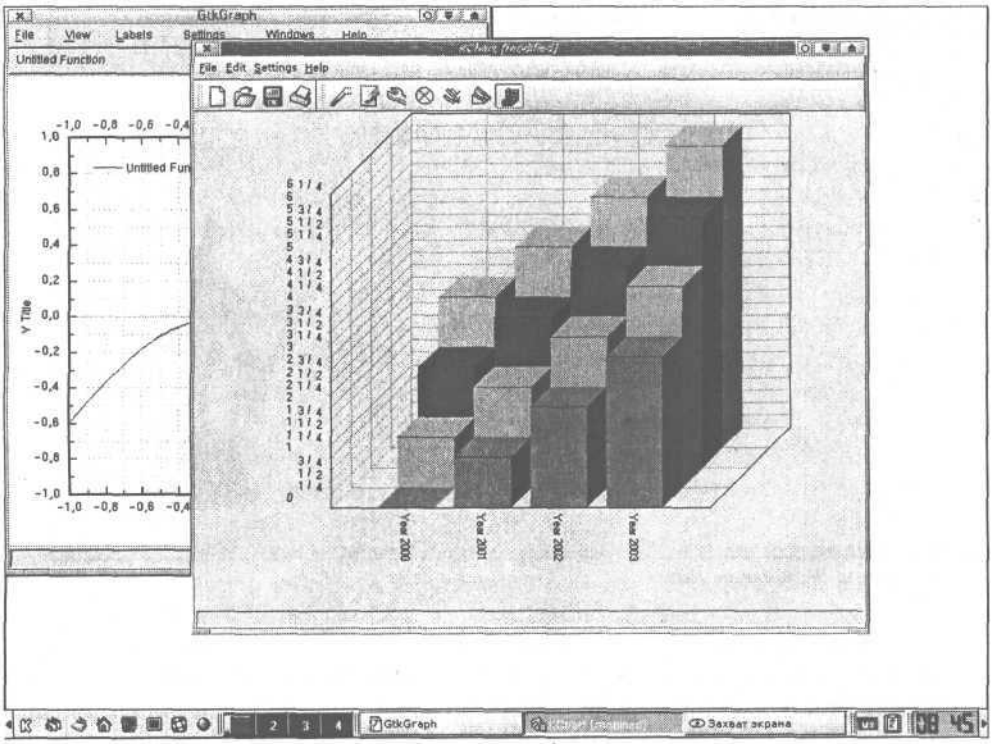


Рис. 20.10. *Оконная среда KDE*

С помощью кнопок «Влево» и «Вправо», расположенных на краях основной панели KDE, вы можете прятать эту панель, чтобы она не загромождала рабочий стол.

На рис. 20.10 я специально использовал стандартные стили и цвета оконной среды KDE. Однако, при желании KDE можно "заточить" под CDE или MacOS так, что вы не отличите от оригинала. Изменить цвета, стили, параметры рабочего стола, шрифты, параметры других программ, а также параметры локализации KDE можно с помощью **Центра Управления** (Control Center).

Итак, начнем рассмотрение всех возможностей Центра Управления KDE (см. рис. 20.11).

В разделе **File Browsing →File Manager** вы можете установить параметры файлового менеджера. Среда KDE обладает встроенным файловым менеджером — **Konqueror.** Эта программа одновременно является и файловым менеджером и браузером.

В разделе **File Browsing → Ассоциация файлов** вы можете сопоставить расширению (типу) файла какую-нибудь программу для обработки этого файла.

В разделе **Hardware** вы можете установить параметры клавиатуры и мыши. Особо тут устанавливать нечего — повтор нажатия и громкость щелчка при нажатии клавиш клавиатуры, а также обыкновенные настройки для мыши: раскладка кнопок, скорость движения и тому подобные параметры.

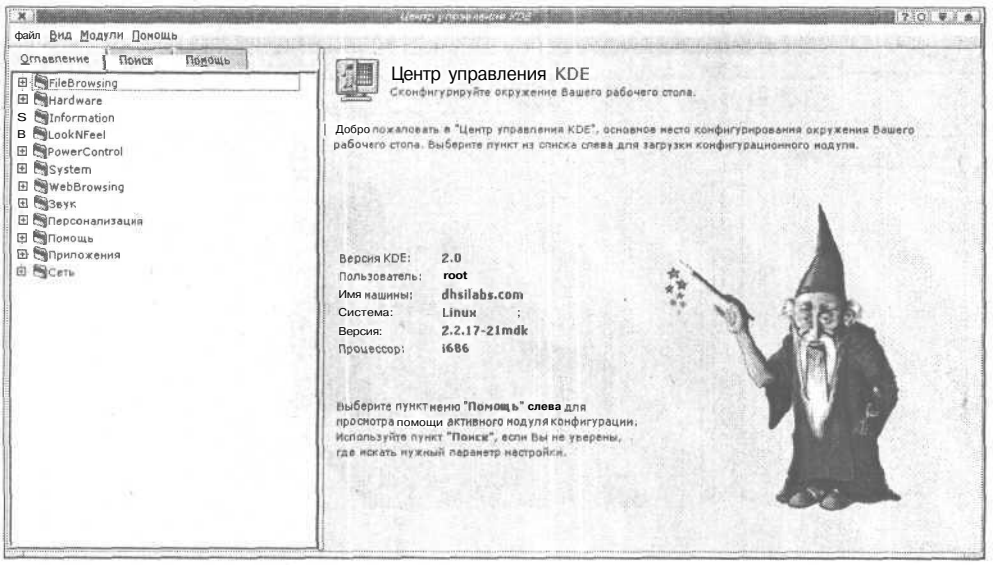


Рис. *20.11. Центр Управления KDE*

Раздел **Information** Центра управления KDE — это своеобразный аналог программы System Information. В этом разделе ничего нельзя изменить, зато вы можете узнать много полезной информации о вашей системе (см. рис. 20.12).

В разделе **LookNFell** можно установить самые разнообразные параметры среды — от размера шрифта до внешнего вида всей среды.

Раздел **LookNFeel → Font Manager** содержит информацию обо всех шрифтах, установленных в системе.

В разделе **Font** можно выбрать шрифт для различных элементов среды KDE:

1. Выбрать общий шрифт.
2. Моноширный.
3. Шрифт для значка на рабочем столе.
4. Шрифт менеджера файлов.
5. Шрифт панели инструментов.
6. Шрифт для меню.
7. Шрифт заголовка окна.

В разделе **General** определяются общие параметры рабочего стола. На вкладке **Общий вид** этого раздела можно выбрать шрифт для значков рабочего стола, установить его цвет и цвет фона.

Разделы **Icon и Icons** содержат параметры пиктограмм KDE. Всевозможные параметры панелей KDE устанавливаются в разделе **Panel.**

Одним из самых интересных разделов **Центра Управления,** влияющих на внешний вид среды, является раздел **Style** (рис. 20.13). Здесь можно выбрать стиль внешнего вида KDE, например, сделать его похожим на CDE.

CDE (Common Desktop Environment) — оконная среда, ставшая стандартом для графических Unix-станций до появления Linux. После того, как система

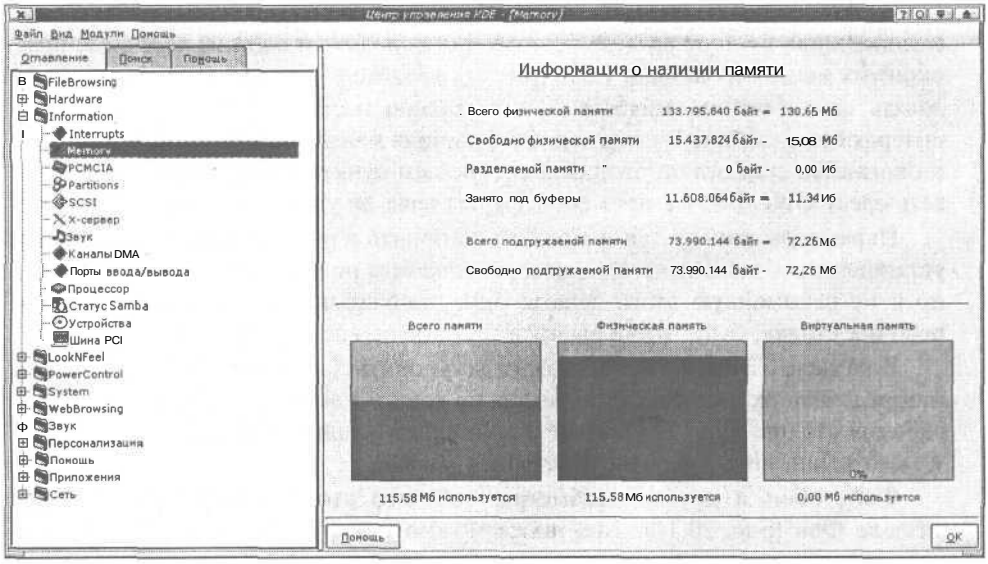


Рис. 20.12. *Системная информация*

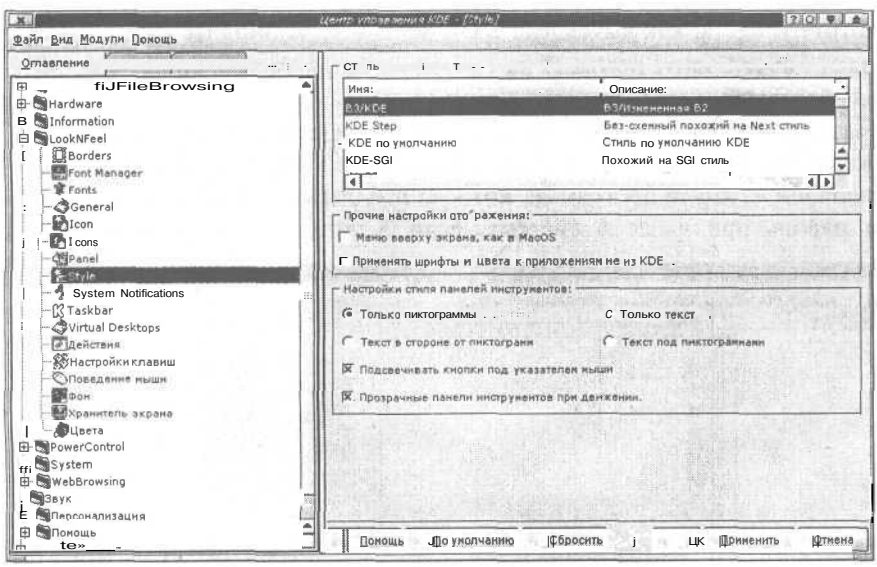


Рис. 20.13. *Раздел «Style»*

X Window была перенесена на платформу Linux, пальму первенства получила среда KDE. Хотя я не могу однозначно сказать, какая среда сейчас более популярна — Gnome или KDE. Кроме этих двух постоянно конкурирующих между собой, существует еще один распространенный оконный менеджер — **Enlightment**. Это довольно красивый менеджер окон и для него разработано довольно много различных схем. На некоторых старых компьютерах вы еще можете увидеть оконные менеджеры **Fvwm**, **Fvwm2**, **AfterStep** и **BlackBox**.

Мощность данных компьютеров не позволяет их владельцам наслаждаться полноценной оконной средой, поэтому они используют один из перечисленных оконных менеджеров. Если у вас тоже такой компьютер, то я могу порекомендовать вам оболочку **AfterStep** — это хорошо настраиваемый с необычным интерфейсом оконный менеджер. Об отличиях между оконными менеджерами и оконными средами поговорим в следующем пункте, когда будем рассматривать среду GNOME. На примере KDE вы вряд ли увидите разницу.

Параметры панели задач можно установить в разделе **Taskbar.** Особенно устанавливать здесь нечего — можно включить режим «Показать все окна», но я не рекомендую этого делать: окна, запущенные на всех виртуальных рабочих столах, будут отображаться на панели задач, что загромоздит ее.

В разделе **Virtual Desktops** можно установить количество рабочих столов и определить их названия. Максимально возможное количество виртуальных рабочих столов — 16. Различные комбинации клавиш для работы со средой можно установить в разделе **Настройка клавиш.**

Фон, обои и другие параметры рабочего стола можно установить в разделе **Фон** (рис. 20.14). Там также можно указать единые параметры для всех рабочих столов, включив режим Общий фон.

Цветовую схему среды можно установить в разделе **Цвета** (рис. 20.15). Если хотите увидеть, как выглядела KDE версии 1, установите схему KDE1.

Раздел **Power Control** поможет вам установить параметры энергосбережения для ноутбуков.

В разделе **System → Login Manager** вы можете установить параметры менеджера регистрации пользователей (рис. 20.16). Менеджер регистрации пользователей — это та программа, которая предлагает ввести имя пользователя и пароль при входе в систему, если X запускается автоматически.

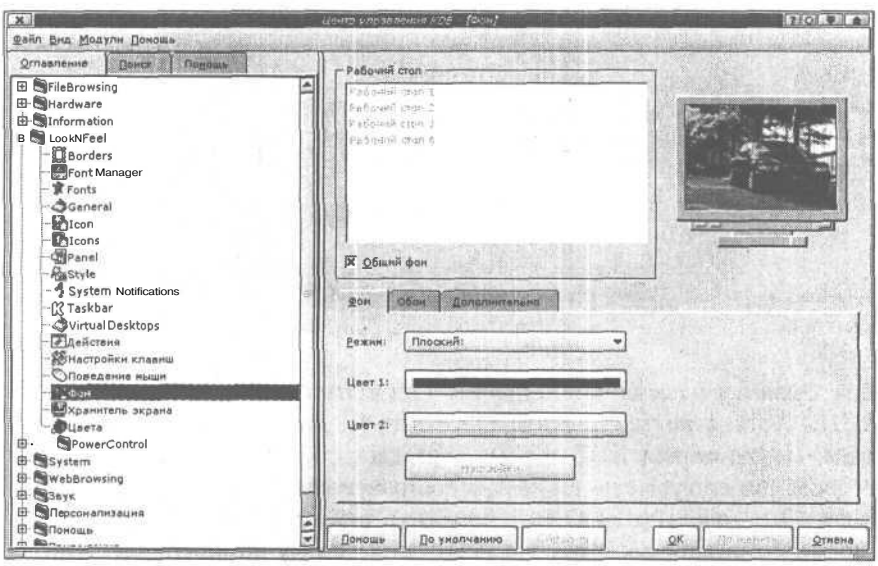


Рис. 20.14. *Раздел «Фон»*

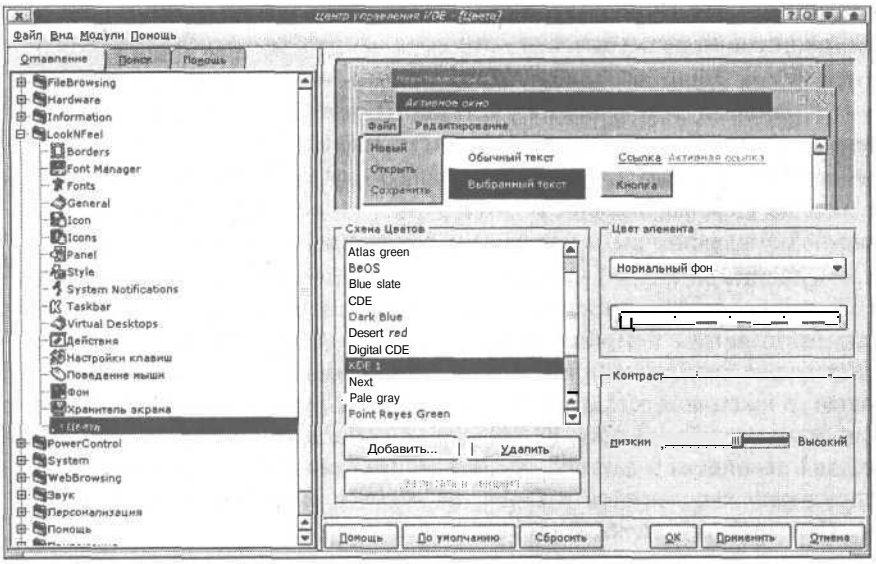


Рис. ***20.15.*** *Раздел «Цвета»*

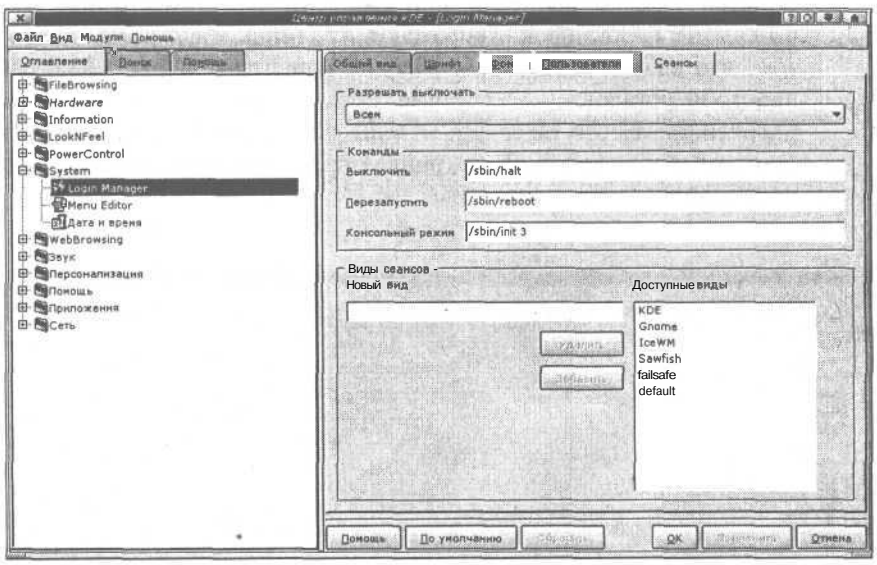


Рис. ***20.16.*** *Login Manager*

В этом разделе после установки системы я изменил всего лишь три параметра. Первый — стиль GUI на вкладке **Внешний вид:** я предпочитаю стиль **Motif.** Второй — выбрал английский язык менеджера регистрации. Третий — изменил значок пользователя: при регистрации пользователя система назначила ему значок, который мне очень не понравился, особенно учитывая тот факт, что этим пользователем оказался я. Изменение данных параметров — это дело вкуса. Системно важные параметры находятся на вкладке **Сеансы.** Здесь можно указать программы для останова системы, перезагрузки и

перехода в консольный режим, а также добавить или удалить сеансы пользователей (оконные менеджеры).

Раздел **Menu Editor** позволяет редактировать основное меню, а **Дата и время** — установить дату и время системы. Нужно отметить, что изменять параметры раздела System может только пользователь root.

Все остальные разделы не представляют для нас никакого интереса, кроме раздела **Персонализация**. В этом разделе (рис. 20.17) можно установить раскладку клавиатуры и параметры локализации (страна и язык).

Переключение раскладки клавиатуры на русский язык зависит от настройки системы. В большинстве систем переключение раскладки (русский/английский) осуществляется с помощью правой клавиши «Crtl». При этом на некоторых клавиатурах должен загореться индикатор Caps. Если переключения не произошло (в текстовом редакторе не отображаются русские буквы), попробуйте нажать клавишу «Caps Lock». Если и это не помогло, попробуйте напечатать что-нибудь в текстовом редакторе, удерживая правый «Alt». Последний вариант (наиболее редко используется в Linux, но часто в Windows): одновременное нажатие **Ctrl+Shift** или **Alt+Shift** (левый или правый).

Кстати, с помощью правого «Alt» можно проверить корректность настройки клавиатуры: если при нажатии «Alt» вместо русских букв или различных иероглифов отображаются латинские символы, то это оговорит о том, что вы не подключили русские шрифты в файле **XF86Config** или они просто не установлены в системе. Если же при удерживании «Alt» вы видите иероглифы, то это означает, что вы используете шрифт, не поддерживающий кодировку **koi8-r.** В настройках программы установите шрифт, который поддерживает кодировку koi8.

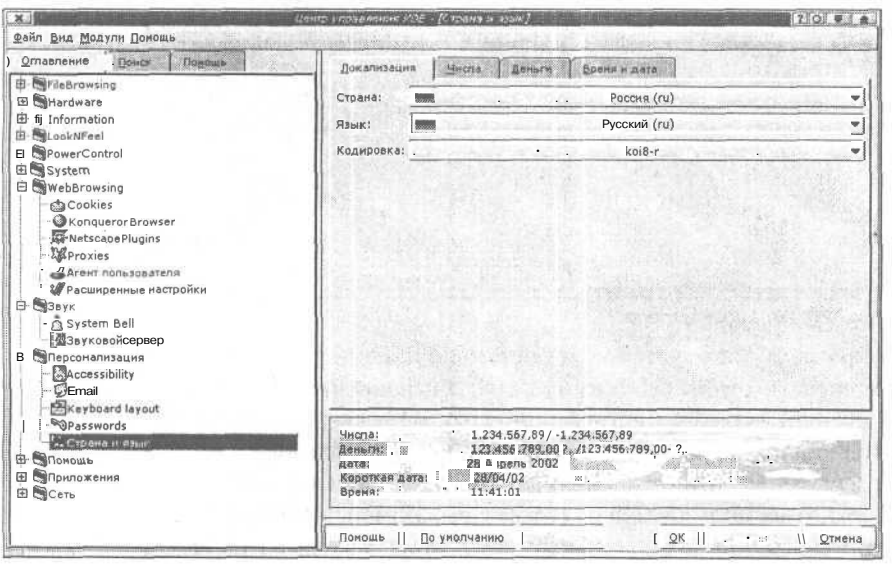


Рис. 20.17. *Выбор страны и языка*

В заключение обзора KDE приведу несколько полезных ссылок.

www.kde.org.....официальный сайт разработчиков KDE.
kde.themes.org.....всевозможные темы для KDE.
developer.kde.org.....центр разработки KDE.
kdecvs.stud.fh-heilbronn.de.....CVS-хранилище.
il8n.kde.org.....центр локализации и документации KDE.
ftp.kde.org.....главный FTP-сервер.

20.6. Оконная среда GNOME

Среда **GNOME** (GNU Network Object Model Environment — Сетевая Объектная Среда GNU) является одним из основных конкурентов среде KDE. Среда GNOME является частью проекта GNU, начатого в 1984 году и целью которого является создание свободно распространяемой Unix-подобной системы.

GNOME — дружественная рабочая среда, значительно облегчающая использование компьютера. Среда GNOME включает в себя рабочий стол, панель для запуска программ и показа информации о состоянии системы, а также набор всевозможных приложений, которые тесно взаимодействуют друг с другом. GNOME, как и KDE, является полностью открытой: каждый может выкачать исходные тексты среды и использовать их. Благодаря этому в процессе разработки GNOME участвовали сотни программистов со всего мира.

Примечание.

*Официальный сайт Gnome — **www.gnome.or***
*Проект GNU — **www.gnu.org***

В среде GNOME настраивается практически все: один раз настроив сеанс по своему вкусу, вам больше не нужно будет повторять его настройку, потому что менеджер сеансов позаботится о сохранении настроек. Как и в KDE, в GNOME поддерживается метод **drag and drop.**

Основные элементы среды — это рабочий стол и панель GNOME. На панели (узкая полоска внизу экрана) расположены кнопка главного меню и апплеты. Все остальное пространство называется рабочим столом (рис. 20.18). Апплеты — небольшие программы, которые работают внутри панели, например, апплет-часы.

Как и в KDE, кнопки со стрелками позволяют прятать и восстанавливать панель.

Работать с Gnome достаточно просто: если вы раньше работали в Windows, вы должны быстро освоить GNOME. Напомню основные операции при работе с мышью:

1. Одним щелчком левой кнопки (или правой, если вы левша) мыши можно выделить объект, а двойным — открыть его. Если объектом является программа, то она будет запущена, а если файл — будет запущена программа, ассоциированная для работы с файлами этого типа.
2. Для перемещения файла (объекта) просто перетащите его в другой каталог.
3. Для копирования файла (объекта) переместите его в каталог-назначение, удерживая клавишу «Ctrl».

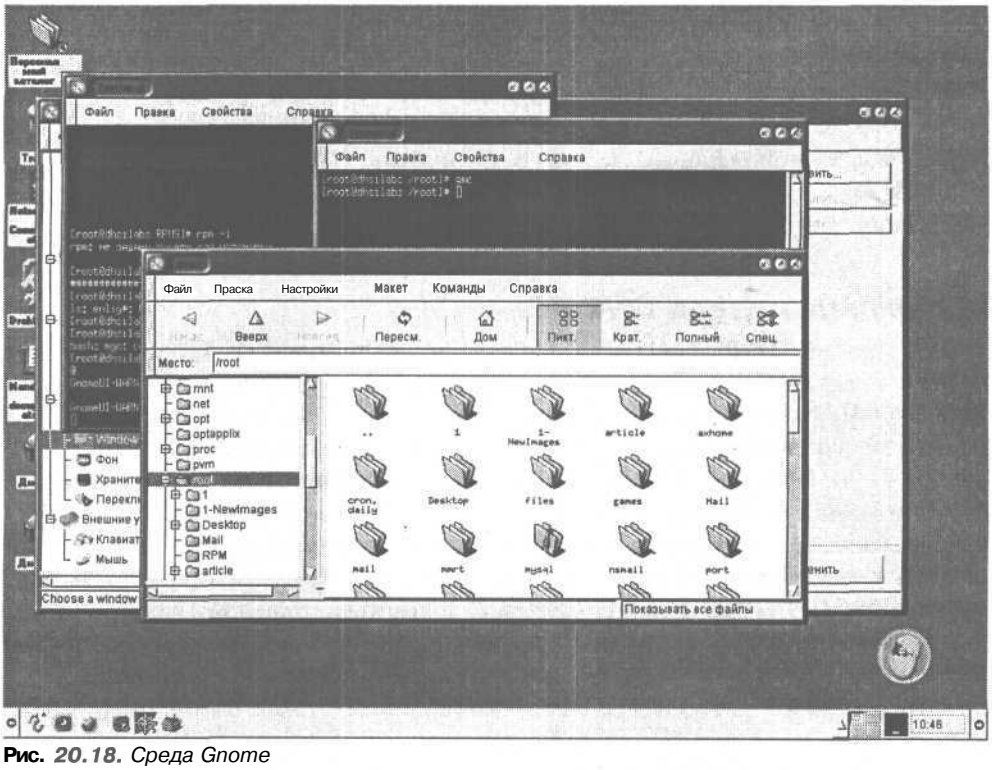


Рис. *20.18. Среда Gnome*

4. Чтобы произвести какие-нибудь операции с файлом (переименование, удаление), щелкните на нем правой кнопкой мыши.
5. По очереди щелкнув на нескольких файлах, удерживая клавишу «Shift», вы выделите группу файлов.

Все эти операции справедливы и для KDE, кроме первой: выделение объекта происходит при наведении на него указателя мыши, а его открытие — с помощью одного щелчка мыши.

Средняя кнопка мыши используется для вставки ранее выделенного фрагмента текста. Выделите фрагмент текста с помощью левой кнопки мыши и вставьте его в другое окно, нажав среднюю кнопку мыши. Если у вашей мыши только две кнопки, то для этого одновременно нажмите левую и правую кнопку мыши.

Панель задач называется **пейджером GNOME.** Название пейджера GNOME точно такое же, как и панели задач: он отображает работающие приложения. В отличие от KDE, пейджер отображает все приложения, а не только те, которые запущены на данном виртуальном рабочем столе.

Теперь нужно сказать о радикальном отличии KDE от Gnome. Чтобы вам было понятно, что именно я хочу сказать, сначала поговорим о менеджерах окон системы X Window.

Менеджер окон -- это программа, которая управляет перемещением, расположением и оформлением окон системы X Window. Популярными

оконными менеджерами являются **Enlightenment, Icewm, Window Maker, Fvwm.** Я перечислил менеджеры окон, совместимые с GNOME. Однако полностью совместимым с GNOME является только **Enlightenment.**

Среда GNOME не привязана к какому-нибудь менеджеру окон. Вы можете использовать любой из них, но чтобы обеспечить корректную работу пейджера GNOME и перетаскивание объектов, нужно убедиться, что выбранный вами менеджер окон совместим с GNOME.

Если ваш оконный менеджер не совместим с GNOME, вы не увидите пейджер GNOME и, следовательно, не сможете переключаться между окнами. Но из этой ситуации есть выход: возле переключателей виртуальных рабочих столов есть кнопка с изображением стрелки вверх. Нажав на эту кнопку, вы увидите список запущенных процессов и легко сможете переключиться в нужное окно.

Внешний вид окон и элементов управления будет зависеть от выбранного вами менеджера и его темы (совокупность стиля и цветовой схемы). В KDE менеджер окон встроен непосредственно в среду, а вы можете выбрать только тему. Вы не сможете комфортно работать в GNOME, если в вашей системе не установлено ни одного менеджера окон.

Нужно также немного сказать о менеджере файлов GNOME. По умолчанию в качестве такового используется **GNU Midnight Commander — gmc** (см. рис. 20.19).

В поле **Location** отображается текущий каталог. Дерево каталогов отображается в левой части окна gmc. В это поле вы можете ввести имя каталога и gmc отобразит его содержимое. Можно ввести также FTP-адрес, например, ftp://ftp.redhat.com.

Выделение файлов происходит так же как и в Windows Explorer: щелкните в области просмотра и, не отпуская клавишу мыши, перемещайте указатель мыши. При этом вы увидите пунктирный прямоугольник. Все файлы,

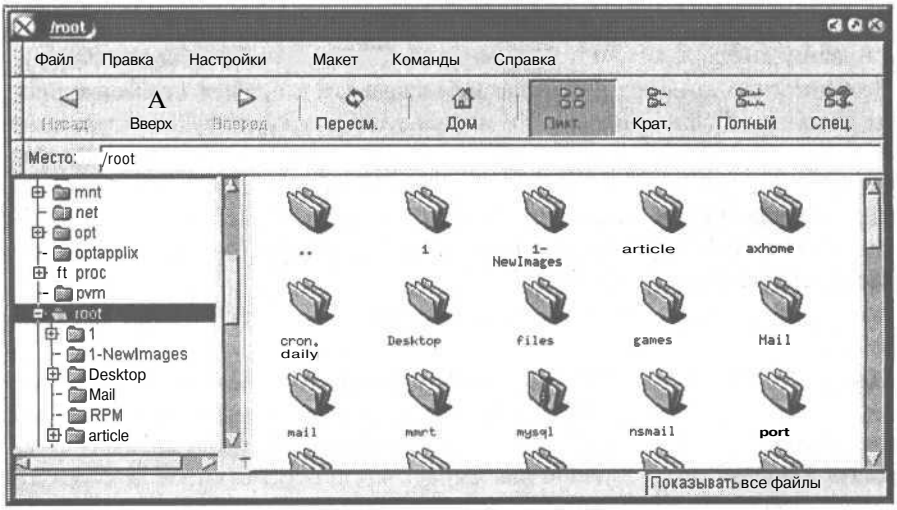


Рис. 20.19. *GNU Midnight Commander*

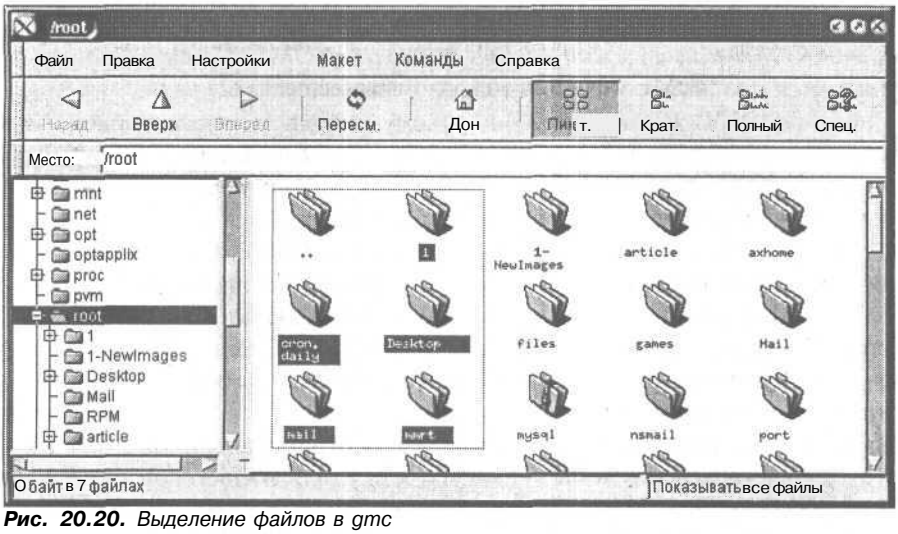


Рис. 20.20. *Выделение файлов в gmc*

попавшие в середину этого прямоугольника, будут выделенными (рис. 20.20). Можно добавить еще один файл к выделенной группе файлов, щелкнув на нем, удерживая клавишу «Ctrl».

Выбрать все файлы можно с помощью команды меню **Edit** → **Select All** (Правка → Выделить все). Выделения файлов по маске (например, *.gif) можно осуществить с помощью команды **Edit** → **Select files** (Правка → Выделить файлы).

Для перемещения файла в другой каталог просто перетащите его в нужный вам каталог. Для копирования файла выполните те же действия, удерживая клавишу «Shift». Если при перемещении файла удерживать клавишу «Alt», откроется меню, в котором можно выбрать операцию: **переместить, копировать, создать символическую ссылку.**

Можно также щелкнуть на файле правой кнопкой мыши и в появившемся меню выбрать команду **Переместить** (Move) или **Копировать** (Copy).

Большинство команд меню **gmc** не нуждается в особых комментариях, кроме команды **Файл** → **Выход.** Не используйте эту команду! Для закрытия окна менеджера файлов используйте кнопку закрытия окна или просто сверните окно с помощью кнопки сворачивания окна. При использовании команды **Выход** будут остановлены все процессы, порожденные менеджером файлов, в том числе будет остановлен процесс, обеспечивающий функционирование рабочего стола GNOME.

Для полноты описания нам осталось рассмотреть только **Центр Управления GNOME.**

Центр управление GNOME чем-то напоминает уже знакомый нам Центр Управления KDE. Как и в KDE, в левой части окна **Центра Управления** мы видим древовидную структуру разделов параметров оконной среды. Раздел **Document Handlers** предназначен для определения обработчиков документов (см. рис. 20.21).

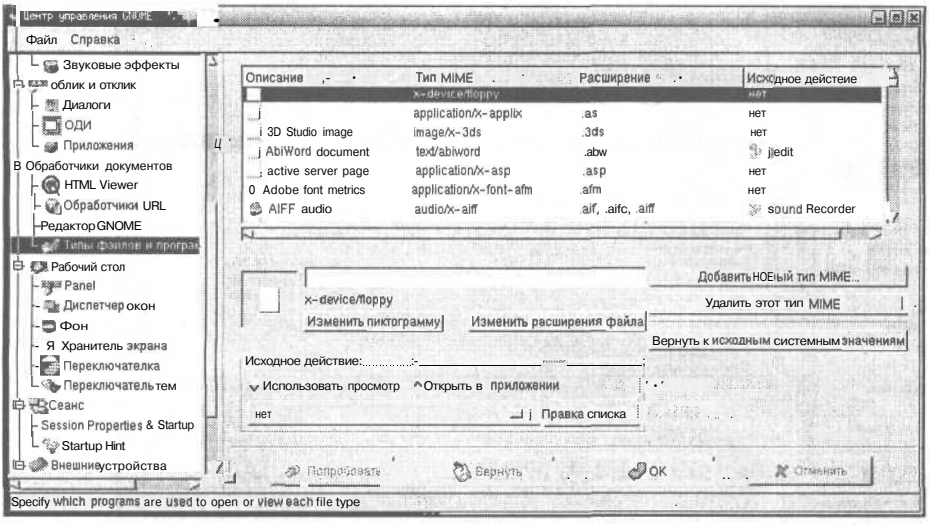


Рис. ***20.21.*** *Раздел Document Handlers*

Для каждого MIME-типа можно задать программы, которые будут использоваться для открытия, просмотра и редактирования документов этого типа (рис. 20.22) Здесь же вы можете установить значок для этого типа файлов, который будет отображаться в окне просмотра менеджера файлов **gmc**.

Раздел **Multimedia** позволяет задать звуки событий оконной среды GNOME. В этом разделе можно определить звуки для системных событий среды, таких, как вход в систему и выход из нее, событий пользователя, событий панели, а также звук для оповещения о получении нового сообщения по электронной почте (см. рис. 20.23).

Раздел **Session.** В этом разделе вы можете определить настройки вашего сеанса: включить или выключить советы при запуске среды, а также указать программы, которые должны запускаться автоматически при запуске GNOME.

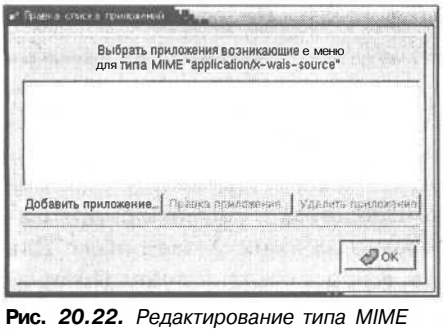


Рис. ***20.22.*** *Редактирование типа MIME*

Настройка интерфейса пользователя осуществляется в одноименном разделе, в котором вы можете установить различные опции окон среды.

Установить параметры панелей GNOME, фона рабочего стола, выбрать хранитель экрана вы можете в разделе **Рабочий стол.** В этом же разделе вы можете выбрать менеджер окон (Window Manager) (рис. 20.24). В разделе **Внешние устройства** устанавливаются параметры клавиатуры и мыши.

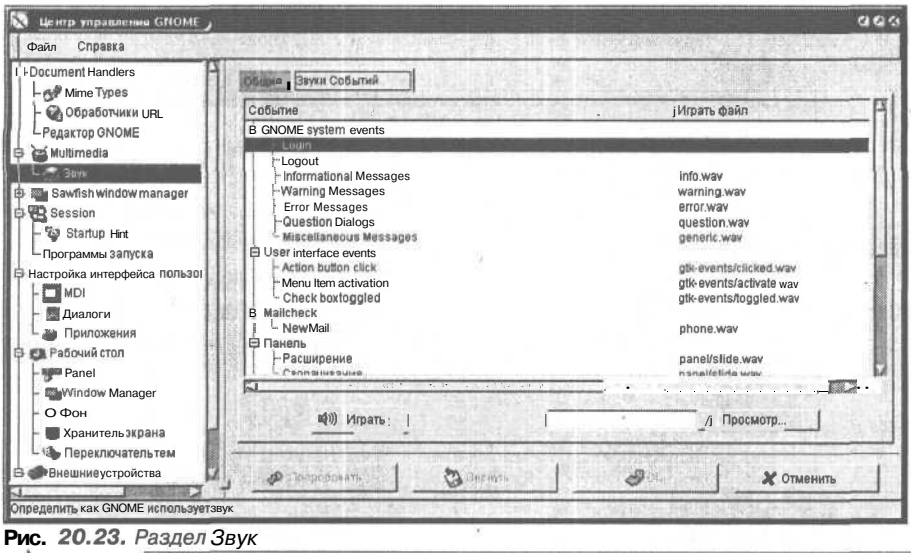


Рис. 20.23. *Раздел Звук*

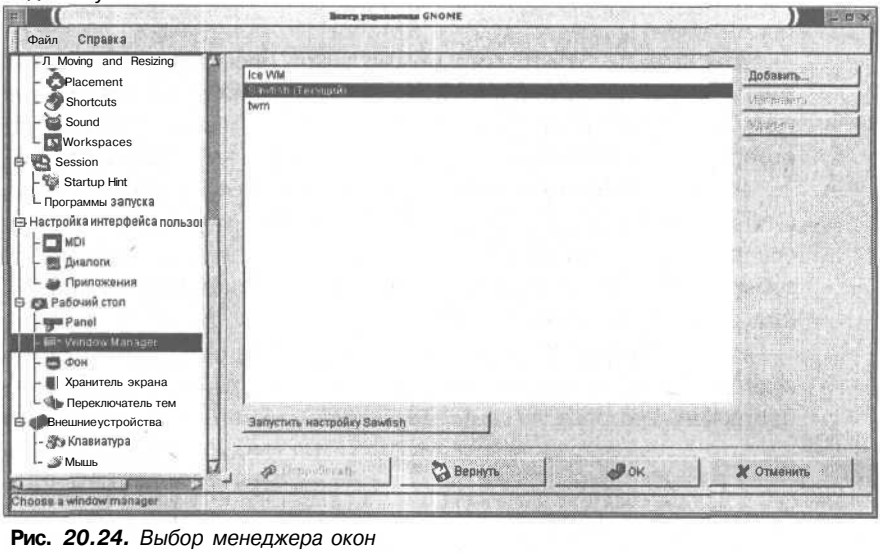


Рис. 20.24. *Выбор менеджера окон*

20.7. Настройка X-терминала

В этом пункте вы узнаете, как можно использовать старенький припавший пылью компьютер с 486-м процессором в качестве X-терминала. Для начала нужно сказать, что такое X-терминал. Как и в обыкновенном терминале, в X-терминале мы видим лишь результаты выполнения программ, а сама программа выполняется на сервере. На обыкновенном терминале нельзя запускать программы, использующие систему X Window, а X-терминал снимает это ограничение. Благодаря этому, даже на самых медленных и старых компьютерах можно работать с новым программным обеспечением. Вы когда-

нибудь пробовали установить систему X Window на компьютере IP166/16MB и запустить последнюю версию Netscape (на то время 4.51)? Для тех, кто не испытывал этого удовольствия, скажу, что запуск Netscape длился около 70-80 секунд. При работе с X-терминалом скорость выполнения программ даже на 486-м компьютере будет примерно такая же, как и на сервере. Естественно, это сильно зависит от загрузки сервера и количества X-терминалов.

При запуске X-терминала вы увидите приглашение для входа в систему в графическом режиме, подобно тому, которое вы видите при запуске вашего сервера, если сервер X у вас запускается автоматически.

Какую аппаратуру можно использовать в качестве X-терминала? Подойдет любой компьютер, даже с процессором 386DX. Обратите внимание на наименование процессора! Нужен именно DX, а не SX. В отличие от последнего, процессор 386DX полностью 32-разрядный. При использовании процессора 486 тип (DX или SX) не имеет значения, потому что они отличаются только наличием или отсутствием математического сопроцессора. Минимальный объем оперативной памяти — 8 Мб. Объем оперативной памяти сильно зависит от дистрибутива и версии X Window, которые мы будем использовать для X-терминала. Я рекомендую установить 16 или даже 32 Мб. Устанавливать ОЗУ объемом более 32 Мб не имеет смысла, так как нам нужно будет запустить только ядро системы и X Window.

Сейчас мы рассмотрим базовую настройку X-терминала. Для начала нам нужно где-нибудь достать или подготовить самостоятельно корневую файловую систему для X-терминала. В качестве операционной системы для X-терминала я рекомендую использовать Slackware-подобную систему, например, тот же **Slackware или Debian**. Это обусловлено двумя причинами. Во-первых, Slackware-подобные системы обладают меньшими требованиями к аппаратуре: чтобы более или менее нормально работать в Slackware нужен компьютер 386DX/4 Мб ОЗУ, а при использовании RedHat-подобных дистрибутивов (Mandrake, Black Cat) нужен как минимум 486/8 Мб ОЗУ. Во-вторых, я решил не создавать корневую файловую систему самостоятельно, а пойти по пути минимального сопротивления и достать корневую систему Slackware-подобной системы и использовать ее. В этом случае достаточно просто развернуть один архив и не тратить время на выбор нужных пакетов RPM. Конечно, можно достать tar-файл с корневой файловой системой и для RedHat, но тут напомнила о себе первая причина — системные требования Red Hat.

О данном методе установке я прочитал в статье Patrick Swieskowski, за что я ему очень благодарен. После того, как я все настроил самостоятельно, я написал этот пункт.

Итак, у нас есть два компьютера: более мощный (сервер) и самый обыкновенный компьютер с процессором 486.

Примечание.

Вот конфигурация «более мощного» компьютера: Intel Celeron 600MHz/192MB/Quantum FireBall 7200rpm/Riva TNT2/Netcard Realtek 10/100Mbit PCI.

Конфигурация терминала: Intel 486DX-100/16MB/Floppy 1,44MB/без HDD/Video S3 2MB PCI/PCI Netcard Realtek 10/100Mbit. Конфигурация

терминала даже несколько превышает минимальную, но ничего более старого тогда я не смог найти. Желательно, если есть такая возможность, использовать сетевые платы для шины PCI, обеспечивающие передачу данных со скоростью 100 Mbps. У меня обе платы работали в режиме 10 Mbps, потому что я соединял их напрямую, то есть без концентратора.

На сервере может быть установлен любой дистрибутив Linux. Желательно, конечно, использовать один дистрибутив как для сервера, так и для X-терминала.

Я установил на обоих, точнее только на сервере, дистрибутив Debian Linux. Корневую файловую систему можно скачать на сайте Debian — **www.debian.org.** Архив с файловой системой называется **base2_2.tgz.** Я выразился не совсем корректно, поскольку в архиве файловая система как таковая не содержится, а в нем запакованы файлы, которые должны находиться в корневой файловой системе: утилиты (каталог bin), файлы конфигурации (etc) и так далее. Вам также понадобится установочный компакт-диск с выбранным дистрибутивом для установки системы X Window. Распакуйте только что выкачанный файл в каталог /usr/xterm:

```
tar -xvzf base2_2.tgz /usr/xterm
```

После выполнения этой команды в каталоге /usr/xterm появятся файлы корневой файловой системы. Теперь немного настроим нашу корневую файловую систему, для этого сделаем ее корневой для нашей операционной системы:

```
chroot /usr/xterm
```

Создадим файл /etc/resolv.conf подобный тому, который используется на сервере. Это нужно для корректного разрешения имен на X-терминале. Можно прописать необходимые IP-адреса и имена в файле /etc/hosts. В большинстве случаев, нужно указать только IP-адрес сервера. Затем отредактируем файл /etc/fstab так:

```
192.168.0.1:/usr/xterm / nfs defaults 0 O
proc /proc proc defaults O O
```

Вы, естественно, замените 192.168.0.1 на IP-адрес сервера для X-терминалов.

Теперь установим и , настроим систему X Window для X-терминала. Вместо менеджера пакетов RPM в Debian нужно использовать программу **apt-get.** Установить X можно с помощью следующих команд:

```
apt-get update
apt-get install xserver-s3 xfonts-100dpi xfonts-base
```

Вторая команда устанавливает сервер X для видеокарт S3. Если у вас другая видеокарта, измените название пакета. Для SVGA-видеокарт сервер называется **xserver-svga**. Пока других пакетов устанавливать не нужно. Позже можно будет установить пакет с русскими шрифтами и необходимые приложения. Сейчас нас интересует функционирование самой X Window на терминале.

Если вы используете Linux Mandrake (или Red Hat) как операционную систему для терминала, установка системы X Window выполняется так, как описано выше в этой главе (см. пункт 20.1). Программа **apt-get** задаст вам несколько вопросов и на основании ваших ответов создаст файл **XF86Config**. В большинстве случаев его не нужно редактировать (пока не нужно: после установки русских шрифтов его нужно будет немного поправить).

У нас в данный момент нет учетных записей пользователей, кроме пользователя root, поэтому систему X Window лучше запускать из файла inittab:

```
X:123456:respawn:/usr/bin/Xll/X -query 192.168.0.1
```

Осталось только перекомпилировать ядро и создать загрузочный диск (компилирование ядра подробно рассматривалось в главе 18). С этой целью перейдите в каталог /usr/src/linux и введите команду **make menuconfig**. Включите следующие опции: в разделе *Networking options* включите опцию **IP: kernel level configuration,** а в появившихся новых опциях отметьте **BOOTP support.** Если вам также нужна поддержка DHCP, включите поддержку DHCP **(DHCP support).** Кстати, эту опцию нужно учитывать при настройке DHCP-клиентов. В разделе *Filesystems→ Network filesystems* включите поддержку файловой системы NFS — NFS **filesystem support.** После ее включения (с учетом того, что поддержка BOOTP уже включена) появится новая опция — **Root file system on** NFS. Ее тоже нужно включить.

Старайтесь сделать ядро как можно более компактным, исключив из его состава лишние опции. Я вообще рекомендую использовать ядра 2.2.x — они более компактны, чем 2.4.x.

Не забудьте включить в состав ядра драйвер для вашей сетевой платы.

Итак продолжим — откомпилируем ядро:

```
make dep
make bzImage
```

Если вы использовали модули, откомпилируйте их:

```
make modules
make modules install
```

После того как ядро откомпилировано, нужно записать его на дискету командой:

```
dd if=bzImage of=/dev/fd0
```

Перед этим нужно перейти в каталог /usr/src/linux/arch/i386/boot.

Теперь укажем ядру, где искать корневую файловую систему. Если бы мы создавали обыкновенную загрузочную дискету, мы бы ввели команду:

```
rdev /dev/fd0 /dev/fdO
```

Так как наше ядро должно искать корневую систему по NFS, мы должны выполнить следующие действия:

```
mknod /dev/boot255 c 0 255
rdev /dev/fdO /dev/boot255
rm -f /dev/boot255
```

Первая команда создает псевдоустройство /dev/boot255, вторая указывает ядру, что корневую файловую систему нужно искать по NFS, а третья удаляет только что созданное устройство, потому что оно больше не нужно нам. Установите корневую файловую систему сервера с помощью команды **chroot**.

Теперь нужно настроить сервер. Настройка NFS рассматривалась в главе 8, а здесь я лишь напомню некоторые моменты. На вашем сервере должен быть запущен демон nfs (nfsd). Если он не установлен, установите пакет **nfs-utils** в Mandrake и Red Hat. Если же вы последовали моему примеру и

использовали Debian на сервере, введите команду **apt-get nfs-user-server.** Затем добавьте следующую строчку в файл /etc/exports на сервере:

```
/usr/xterm 192.168.0.2/255.255.255.0(rw,no_root_squash)
```

192.168.0.2 — это IP-адрес X-терминала. Вы можете указать маску подсети, как это сделал я, тогда при подключении новых X-терминалов вам не нужно будет перечислять каждый терминал отдельно. После этого установите сервер BOOTP:

в Mandrake и RedHat — **rpm –ih /mnt/cdrom/Mandrake/RPMS/bootparamd***

в Debian — **apt-get install bootp**

Затем в файл /etc/bootptab (или /etc/bootparams) **добавьте строчку**

```
xterm01:vm=auto:ip=192.168.0.2:ht=ethernet:ha=xxxxxxxxxxx:rp=/usr/xterm
```

где: xterm01 ..... имя X-терминала;

192.168.0.2 ... его IP-адрес;

xxxxxxxxxxx ... MAC-адрес X-терминала.

В файл /etc/inetd.conf, если вы используете суперсервер **inetd,** добавьте строчку (или раскомментируйте ее):

```
bootps  dgram  udp wait root  /usr/sbin/tcpdbootpd -i -t 120
```

Можно также использовать:

```
bootps dgram udp wait root /usr/sbin/bootpd bootpd -i -t 120
```

При использовании xinetd:

```
service bootps
{
    socket_type = dgram
    protocol    = udp
    user        = root
    wait        = yes
# в Linux Mandrake / Red Hat
    server      = /usr/sbin/rpc.bootparamd
# в Debian
    server      = /usr/sbin/bootpd
}
```

Заставьте **xinetd (inetd)** перечитать файл конфигурации:

```
killall-HUPinetd
```

Теперь осталось настроить диспетчер дисплеев — **xdm**. При использовании Red Hat или Mandrake xdm, скорее всего, будет уже установлен. А вот если вы использовали «укороченную» файловую систему от Debian, xdm установлен не будет. Для его установки выполните (на файловой системе сервера) команду:

```
apt-get install xdm
```

Файлы конфигурации xdm находятся в каталоге /etc/X11/xdm. В файл Xservers добавьте строку:

```
192.168.0.2:0 foreign
```

Эта строка обеспечит подключение X-терминала. В файл Xaccess добавьте строку:

```
192.168.0.2
```

Если в вашем файле xdm-config будет строка **DisplayManager.requestPort: 0,** то закомментируйте ее. Примеры используемых файлов конфигурации **xmd** приведены в листингах 20.3...20.5. Вот, собственно, и все.

Листинг 20.3. Файл /etc/X11/xdm/Xservers

```
# $XConsortium: Xserv.ws.cpp,v 1.3 93/09/28 14:30:30 gildea Exp
$
#
#
# $XFree86: xc/programs/xdm/config/Xserv.ws.cpp,v 1.1.1.1.12.2
1998/10/04 15:23:14 hohndel Exp $
#
# Xservers file, workstation prototype
t
# Each X terminal line should
# look like:
#   XTerminalName:0 foreign
#
:0 local /usr/X11R6/bin/X
192.168.0.2:0 foreign
```

Листинг 20.4. Файл /etc/X11/xdm/Xaccess

```
192.168.0.2
```

Листинг 20.5. Файл /etc/X11/xdm/xdm-config

```
! $XConsortium: xdm-conf.cpp /main/3 1996/01/15 15:17:26 gildea
$
DisplayManager.errorLogFile:    /var/log/xdm-error.log
DisplayManager.pidFile:     /var/run/xdm.pid
DisplayManager.keyFile:     /etc/X11/xdm/xdm-keys
DisplayManager.servers:     /etc/X11/xdm/Xservers
DisplayManager.accessFile:  /etc/X11/xdm/Xaccess
! All displays should use authorization, but we cannot be sure
! X terminals will be configured that way, so by default
! use authorization only for local displays :0, :1, etc.
DisplayManager._0.authorize:    true
DisplayManager._1.authorize:    true
! The following three resources set up display :0 as the
console.
DisplayManager._0.setup:/etc/X11/xdm/Xsetup_0
DisplayManager._0.startup:  /etc/X11/xdm/GiveConsole
DisplayManager._0.reset:/etc/X11/xdm/TakeConsole
DisplayManager._0.startAttempts: 1
!
DisplayManager*resources:   /etc/X11/xdm/Xresources
DisplayManager*session:     /etc/X11/Xsession
DisplayManager*authComplain:    false
```

21

Linux в игровом зале

V

21.1. Достоинства и недостатки

В этой главе будет рассмотрена настройка Linux как рабочей станции для игрового зала. У вас может возникнуть вопрос: почему именно как рабочей станции? Ответ очень прост: любую Linux-систему довольно легко превратить из рабочей станции в сервер, причем без потери надежности и производительности, чего нельзя сказать о Windows.

Итак, допустим, что у вас есть небольшой игровой зал, скажем на 20...30 компьютеров и вам нужно по тем или иным причинам перейти на платформу Linux. Как я уже отмечал, любую из Linux-машин можно настроить как сервер и при этом можно использовать ее как рабочую станцию, то есть при этом не теряется ни одно пользовательское место при организации сервера.

Сейчас мы разберемся во всех достоинствах и недостатках (к сожалению, таковые имеются) такого преобразования. Достоинства и недостатки я буду приводить одновременно: сначала положительный момент, а затем — обратную сторону медали.

Достоинство. Самым большим достоинством является, на мой взгляд, существенная экономия денег, что немаловажно при открытии нового зала, когда первым делом нужно окупить средства, вложенные в его организацию.

Коробочные версии Windows XP Home Edition стоят около 160 долларов США. При открытии зала с парком в 30 машин общая стоимость боксовых версий обойдется вам примерно в $ 4800. При покупке OEM-версий стоимость Windows составит около $ 2400. При всем этом вы получите одноранговую сеть, состоящую из 30 компьютеров под управлением Windows. Если же вам нужно организовать сервер для доступа к Интернет, то за него придется выложить еще около $ 1000. Итого $ 5800. И это только программное обеспечение — математика, а ведь еще нужно купить железо, дополнительную аппаратуру, мебель и т.д. Если нормальный компьютер для игрового зала можно купить за $ 350...450, то зачем же увеличивать его стоимость даже на 80 долларов при использовании OEM-версии?

В случае с Linux вам достаточно купить один дистрибутив стоимостью $ 5...10 долларов (при этом вы платите только за носители информации, то

есть за компакт-диски, входящие в состав дистрибутива). Потом вы можете установить этот дистрибутив на неограниченное число компьютеров.

Недостаток. Несмотря на довольно приличную сумму сэкономленных денег, возрастут ваши ежемесячные расходы. Дело в том, что Linux-залу нужен квалифицированный системный администратор, хотя бы на первых порах — пока все не заработает так, как нужно. В этом случае услуги студента-первокурсника, пусть даже отлично знающего Windows, не будут соответствовать вашим запросам. Установить Linux сможет каждый: современные программы установки Linux все сделают за вас. А вот настроить систему такой «специалист» вряд ли сможет, а поэтому вам понадобится специалист, хорошо знающий Linux. Следовательно, и зарплата у него должна быть как минимум в два-три раза больше, чем у администратора, обслуживающего одноранговую Windows-сеть. Кроме этого, понадобится определенное время на настройку всех компьютеров, так как настройка Linux занимает больше времени, чем Windows, а особенно настройка игровых приложений под Linux. Подробнее о переходе на Linux вы можете прочитать в моих статьях «Переходим на Linux» и «Строим бесплатный Интернет-сервер», которые вы найдете на прилагаемом компакт-диске.

Достоинство. Если вы имеете хотя бы небольшой опыт работы с Linux, вы уже должны были для себя отметить надежность работы этой операционной системы. А это значит, что вам или вашему администратору не нужно по 5...10 раз в день перезагружать машину из-за того, что «программа выполнила недопустимую операцию». Большинство современных игр являются сетевыми или же обладают поддержкой сети. Операционная система Linux работает с сетью гораздо быстрее, чем любая система семейства Microsoft.

Недостаток. Да, сетевые игры под управлением Linux работают быстрее. Но это относится только к Linux-играм. А на платформу Linux портировано не такое уж и большое количество игр. Самые популярные игры продолжают существовать только в Windows-варианте, поэтому запускать такие игры вам придется из-под эмулятора Windows, что сказывается на работе игры. Во-первых, игры в родной Windows-среде работают стабильнее. Во-вторых, при работе из-под эмулятора, игры основательно «притормаживают». В-третьих, под управлением эмулятора работают далеко не все игры, хотя самые популярные все же работают. Конечно, все это в какой-то мере компенсируется более быстрой работой сети, но иногда даже не хочется играть, когда тебя «убивают» из-за того, что эмулятор не успел вовремя обновить экран. Но в любом случае, игры под Linux работают, причем некоторые даже показывают довольно неплохие показатели, например, производительность Counter Strike под управлением эмулятора составили 83...88 fps, а под Windows 98 — 92...95 fps (использовалось разрешение 800x600 и драйвер видеокарты для Linux от компании nVidia). Конфигурация компьютера: AMD Athlon 700 MHz/256 MB/ 40 GB Quantum 7200 rpm/32 MB/RivaTNT2 Pro.

Достоинство. При использовании Linux можно не покупать полноценные компьютеры, а только X-терминалы. В качестве X-терминала может высту-

пать обыкновенный PC-компьютер без жесткого диска. Все программы, в том числе X-сервер и игры, будут выполняться на сервере, а пользователь увидит лишь результат выполнения программы. Естественно, в качестве сервера нужно купить довольно мощный компьютер. В начале книги, когда обсуждалась установка Linux, я писал, что при работе с Linux более критичен объем ОЗУ, чем частота процессора. В случае с сервером терминалов частота играет тоже довольно большую роль, потому что сервер должен будет обслуживать два-три десятка клиентов. При большом количестве клиентов целесообразно будет установить несколько серверов, скажем, один сервер на каждые 25 компьютеров. При этом предпочтительнее использовать двухпроцессорные конфигурации для сервера. Настройка X-терминалов рассматривалась в гл. 20 этой книги.

21.2. Выбор аппаратного обеспечения для игрового зала

Конфигурации компьютеров могут сильно варьироваться в зависимости от многих факторов. Прежде всего, это финансовый фактор. Если с финансами у вас все в порядке, посмотрите, какую технику используют ваши конкуренты. При покупке компьютеров не покупайте «космические» конфигурации, но и не нужно отставать от ваших конкурентов. Хотя, если все правильно настроить, более старые конфигурации могут показать более высокие результаты, чем более производительная, но не настроенная техника.

Также нужно учитывать контингент пользователей игрового зала: если они и в глаза не видели Pentium II, то незачем покупать P IV. Конечно, здесь я утрирую, но это необходимо для лучшего пояснения самой идеи.

Из моего небольшого опыта администрирования игровых залов (в своей жизни я работал в игровом зале всего три месяца) могу сказать, что пользователи обращают внимание на следующее:

1. Размер монитора.
2. Видеоплату.
3. Работу сети.

Поэтому на этих трех факторах не стоит экономить. Монитор — это первое, что бросается в глаза при посещении игрового зала. К тому же, это то устройство, на которое пользователь будет смотреть глазами. Как говорится, встречают по одежке... Я бы порекомендовал использовать 17 или 19-дюймовые мониторы. Если есть выбор, старайтесь выбрать мониторы с необычным дизайном или с цветом, отличным от белого. В общем, старайтесь привлечь внимание посетителя.

На видеоплату обращают внимание все посетители: более продвинутые — при начале работы с компьютером, а менее продвинутые — в процессе работы. При этом качество видеоплаты выражается двумя словами: «быстро» или «медленно». Я рекомендую использовать видеоплаты компании nVidia, например, GeForce. Объем видеопамяти — 32 или 64 Мб. Больше устанавливать не стоит, потому что при необходимости для загрузки текстур будет использоваться оперативная память, а видеоплаты со 128-ю (или

более) мегабайтами видеопамяти стоят довольно дорого. Несмотря на проблемы, возникающие при работе видеоплат nVidia в среде Linux, при должной настройке они показывают высокие результаты. Установка драйверов nVidia будет рассмотрена немного ниже.

Не хочется создавать плохого впечатления о компании ATI, но я очень не рекомендую использовать видеоплаты производства этой компании. Да, комплектация этих плат заслуживает внимания, но вот производительность оставляет желать лучшего.

Немаловажный фактор при организации игрового зала — это сеть и качество ее работы. Ни в коем случае не используйте сеть стандарта lOBase-2(5) на коаксиальном кабеле! Намного лучше установить самый дешевый коммутатор (switch) — хотя бы на 10 Мбит/с. В рассматриваемом случае (30 компьютеров) о варианте с сетью на 10 Мбит/с можете сразу забыть. Такая скорость подойдет разве что при работе 7...10 компьютеров.

Операционная система Linux также вносит свои коррективы в конфигурацию компьютеров. При использовании эмулятора Windows лучше установить 256 Мб ОЗУ (или более). Напомню, что для нормальной работы большинства игр под управлением Windows 98 достаточно 128 МБ, а Windows XP - - 256. Минимальная конфигурация компьютера: Athlon (Celeron) 700 MHz 128 MB/20 GB/32 MB RivaTNT2 Pro/100 MBit netcard.

Нужно отметить, что процессоры AMD (в частности Athlon) работают с мультимедиа-приложениями (то есть с играми) быстрее, чем процессоры Celeron с той же частотой, но процессоры Intel в среде Linux работают надежнее. Если бы мне нужно было выбирать между процессором Athlon и Celeron, я бы выбрал процессор Celeron с более высокой частотой.

Рекомендуемая конфигурация: Intel Pentium III 900 MHz (1 GHz) 256 MB/40 Gb/64 MB nVidia GeForce 400MX. Для большинства приложений такой конфигурации будет вполне достаточно.

Можно использовать процессор Pentium IV, но при использовании этого процессора реально повышается производительность только тех приложений, которые поддерживают этот процессор, а также от алгоритма вычислений. Поэтому не все приложения будут быстрее работать на Pentium IV.

21.3. Установка драйверов для видеокарт nVidia

Лучшим дистрибутивом для рабочей станции игрового зала будет дистрибутив Alt Junior Linux 1.1. Этот дистрибутив достаточно прост в настройке и сразу после установки готов к использованию. Поэтому все дальнейшие действия будут рассматриваться на примере этого дистрибутива, а также дистрибутивов Red Hat и Mandrake Linux. Если вы используете другой дистрибутив, возможно, вам нужно будет скачать версии пакетов для своего дистрибутива, но в этом случае важен сам принцип — вы все сможете сделать по аналогии, используя любой другой дистрибутив.

Примечание.

Я не рекомендую устанавливать дистрибутив ALT Junior Linux 1.1 на сервере.

Компания nVidia не разрешает разработчикам дистрибутивов Linux включать драйвер в состав дистрибутива, однако сам драйвер бесплатно доступен на сайте nVidia — http://www.nvidia.com/view.asp?PAGE=linux

Для установки драйвера вам нужно загрузить два файла:

1. GLX-драйвер.
2. Драйвер ядра для видеокарты nVidia.

Оба файла доступны в уже собранных пакетах RPM, поэтому у вас не должно быть проблем с их установкой. При загрузке файлов обратите внимание на версии файлов: они должны совпадать.

Сначала загрузите GLX-драйвер:

http://download.nvidia.com/XFree86_40/1.0-2960/NVIDIA_GLX-1.0-2960.i386.rpm

Затем выберите драйвер ядра в зависимости от используемого вами дистрибутива, и загрузите его (см. табл. 21.1).

Драйверы ядра *Таблица 21.1*

Дистрибутив	Драйвер
Red Hat Linux 7.3	NVIDIA_kernel-1.0-2960.rh73up.i686.rpm
ALT Junior Linux	NVIDIA_kernel-1.0-1541 -alt7.i686.rpm
Mandrake Linux 8.2	NVIDIA_kernel-1.0-2960.mdk82up.i586.rpm

При загрузке файла обратите внимание на версию вашего дистрибутива, а также на используемую вами платформу (386, 586, 686).

Затем установите файлы:

```
rpm -ivh NVIDIA_kernel.i386.rpm
rpm -ivh NVIDIA_GLX.i386.rpm
```

Я специально не указывал номера версий, потому что не знаю, какой дистрибутив вы используете. На данном этапе важен порядок установки: сначала нужно установить драйвер ядра, а потом GLX-драйвер.

Сразу после установки установите «среднее» разрешение монитора. Например, если максимальное разрешение, поддерживаемое вашим монитором — 1280x1024, установите 1024x768 или даже 800x600. После настройки драйвера вы сможете установить любое другое разрешение.

Возможно, вам нужно будет обновить систему. Мне пришлось обновить следующие пакеты:

```
mkinitrd-2.7.1
mktemp-1.3.1
modutils-2.4.10
```

Также я установил обновления для моего ядра (до версии 2.4.12) и системы ALSA (также до версии 2.4.12).

Напомню, что обновить пакет вы можете с помощью команды:

```
rpm -Uvh <package.rpm>
```

Откройте в любом текстовом редакторе файл `/etc/X11/X86Config-4` и найдите строку:

```
Driver "nv"
```

(или подобную ей, например, Driver “vesa”). Эту строку нужно заменить на:

```
Driver "nvidia"
```

Убедитесь, что в вашем файле есть строка:

```
Load"glx"
```

Если ее нет, добавьте ее после строки:

```
Load"dbe"
```

После этого удалите строки:

```
Load"dri"
Load"GLcore"
```

Полный листинг файла /etc/X11/XF86Config-4 приведен ниже (см. листинг 21.1).

Листинг 21.1. Файл /etc/X11/XF86Config-4 для драйвера nVidia

```
Section "ServerLayout"
    Identifier  "Anaconda Configured"
    Screen  0   "Screen0" 0 0
    InputDevice "Mouse0" "CorePointer"
    InputDevice "Keyboard0" "CoreKeyboard"
EndSection

Section "Files"
    RgbPath   "/usr/X11R6/lib/X11/rgb"
    FontPath   "unix/:7100"
    FontPath   "/usr/X11R6/lib/X11/fonts/cyrillic/"
EndSection

Section  "Module"
    Load"dbe"
    Load"glx"
    Load"extmod"
    Load"fbdevhw"
    Load"pex5"
    Load"pex5"
    Load"record"
    Load"xie"
EndSection

Section  "InputDevice"
    Identifier   "KeyboardO"
    Driver   "keyboard"
    Option   "XkbRules"   "xfree86"
    Option   "XkbModel"   "pc105"
    Option   "XkbLayout"  "ru"
    Option    "XkbVariant"     "basic"
EndSection

Section "InputDevice"
    Identifier    "Mouse0"
    Driver    "mouse"
    Option   "Protocol"  "PS/2"
    Option   "Device"  "/dev/psaux"
    Option   "ZAxisMapping" "4  5"
```

```
    Option  "Emulate3Buttons" "no"
EndSection

Section "Monitor"
    Identifier  "Monitor0"
    VendorName  "Monitor Vendor"
    ModelName   "Monitor Model"
    HorizSync    30-61
    VertRefresh 50-120
    Option  "dpms"
    # -- 1400x1050 --
    # 1400x1050 @ 60Hz, 65.8 kHz hsync
    Modeline "1400x1050"    129 1400 1464 1656 1960
        1050 1051 1054 1100 +HSync +VSync
    # 1400x1050 @ 70Hz, 76.8 kHz hsync
    Modeline "1400x1050"    151 1400 1464 1656 1960
        1050 1051 1054 1100 +HSync +VSync
    # 1400x1050 @ 75Hz, 82.3 kHz hsync
    Modeline "1400x1050"    162 1400 1464 1656 1960
        1050 1051 1054 1100 +HSync +VSync
    # 1400x1050 @ 85Hz, 93.2 kHz hsync
    Modeline "1400x1050"    184 1400 1464 1656 1960
        1050 1051 1054 1100 +HSync +VSync
EndSection

Section "Device"
    Identifier  "RIVA TNT2"
    Driver      "nvidia"
    VendorName  "RIVA TNT2"
    BoardName   "RIVA TNT2"
EndSection

Section "Screen"
    Identifier  "Screen0"
    Device   "RIVA TNT2"
    Monitor "MonitorO"
    DefaultDepth    16
    Subsection  "Display"
    Depth   16
    Modes    "1024x768"
EndSubsection

EndSection

Section "DRI"
    Mode 0666
EndSection
```

Теперь нажмите комбинацию Ctrl+Alt+Backspace для перезагрузки сервера X. Если вы все сделали правильно, при загрузке вы должны увидеть логотип nVidia. Если компьютер при перезагрузке сервера X зависает, попробуйте отключить поддержку AGP 2x. Если и это не помогло, проверьте, установлена ли библиотека libGlwrapper:

```
# rpm -qa I grep -i libglwrapper
```

В случае если эта библиотека не установлена, установите ее:

```
# rpm -ihv libGLwrapper*
```

Перед выполнением этой команды нужно перейти в каталог, в котором находятся пакеты RPM вашего дистрибутива. Обычно библиотека libGlwrapper находится на первом диске дистрибутива.

После установки библиотеки выполните команду:

```
libglwrapper
```

При запуске некоторых игр (Quake II, Heretic II) у вас могут возникнуть проблемы с библиотекой GL. Устранить их можно с помощью следующих команд:

```
# rm -rf /usr/X11R6/lib/libGL.so
# rm -rf /usr/X11R6/lib/libGL.so.1
# ln -s /usr/X11R6/lib/libGL.so.nvidia /usr/X11R6/lib/libGL.so
# ln -s /usr/X11R6/lib/libGL.so.nvidia /usr/X11R6/lib/libGL.so.1
```

Перед этим желательно скопировать куда-нибудь файлы libGL.so и libGL.so. 1 для того, чтобы у вас была возможность восстановить их в случае необходимости.

Примечание.

Устанавливать драйвер для видеокарт nVidia вам нужно лишь в том случае, если вы планируете использовать игры, разработанные для Linux. Если же вы хотите запускать только Windows-игры, используя эмулятор wine, драйвер можно не устанавливать, поскольку и без него все нормально работает. Правда, при установке драйвера производительность Windows-игр все же повысилась.

21.4. Установка Windows-эмулятора wine

После установки драйверов видеокарты вы уже можете наслаждаться Linux-играми, однако, как я уже писал, игры (имеются в виду хорошие игры) для Linux — это большая редкость. Большинство популярных игр распространяются только в Win32-версии. Поэтому сейчас мы займемся настройкой эмулятора **wine,** который обеспечит запуск Windows-игр в среде Linux.

Стандартный эмулятор **wine** входит в состав практически любого дистрибутива, но он не обеспечивает должного уровня эмуляции операционной системы Windows. Для нормальной работы игр для Windows вам потребуется эмулятор winex (и его следующие версии -- **winex2**, **winex3**). Не путайте эмулятор **wine** с эмулятором **winex**!. Эмулятор **winex** — это отдельная разработка и, к сожалению, этот эмулятор не является бесплатным — за него нужно платить. Купить данный эмулятор можно на сайте http://www.transgaming.com. При покупке winex у вас появится возможность загрузить уже скомпилированную версию эмулятора в виде пакета rpm. На этом же сайте можно бесплатно загрузить исходный текст эмулятора, но вы потратите много времени на то, чтобы привести исходный код к пригодному для компиляции виду.

Устанавливать эмуляторы нужно в такой последовательности: **wine, winex, winex2**, **winex3**. Напомню, что эмулятор **wine,** скорее всего, уже будет установлен у вас.

21.5. Запуск игр с помощью эмулятора wine

Эмулятор **wine** гарантированно поддерживает следующие игры:

1. Counter Strike
2. StarCraft
3. Fallout
4. Fallout 2
5. Gunman
6. Quake 2
7. Quake 3
8. Soldier of Fortune
9. Unreal Tournament
10. Red Alert (все версии)
11. Diablo 2
12. Cesaer
13. Return to Castle Wolfenstain
14. Star track
15. Kingpin
16. Nox
17. Jadded Alliance
18. 4x4 Evolution
19. American McGee Alice
20. Daikatana
21. Heroes of Might and Magic III
22. Delta Force 1,2

Возможно, у вас будут работать и другие игры.

Перед установкой игры удалите все файлы с расширением reg в подкаталоге **.wine** домашнего каталога пользователя root:

```
# rm -rf /root/.wine/*.reg
```

Запустите сервер X, если он еще не запущен командой:

```
startx
```

Если сервер X загружен, но вы работаете в консоли, перейдите в графический режим и запустите графический эмулятор терминала, например, xterm. Для установки новой игры выполните команду:

```
# wine install_program
```

Предположим, что программа установки игры называется `setup.exe` и находится в корневом каталоге компакт-диска. Для установки такой игры нужно ввести команду:

```
wine /mnt/cdrom/setup.exe
```

Игра будет установлена в каталог `/usr/local/wine-c/games/<название_игры>` или же в каталог `/usr/ share/wine-c/games/<название_игры>`. Узнать, в какой из этих двух каталогов была установлена игра, вы можете, просмотрев файл `/root/.wine/.config`. В секции **Drive C** определяются настройки для диска C:

```
[Drive C]
"Path" = "/usr/share/wine-c"
"Type" = "hd"
"Label" = "MS-DOS"
"Filesystem" = "win95"
```

Пользовательские настройки эмулятора находятся в файле config, который находится в каталоге $HOME/.wine. Глобальные настройки эмулятора вы можете изменить в файле /etc/wine.reg.

После установки игры перейдите в каталог, в который была установлена игра, то есть в каталог /usr/share/wine-c/games/<название игры>. Попробуйте запустить ее, поочередно используя команды:

```
wine game.exe
winex game.exe
winex2 game.exe
winex3 game.exe
```

Естественно, вместо параметра **game.exe** нужно подставить реальное имя исполняемого файла игры. Данные команды нужно вводить в терминале X, например, **kterm**, если вы используете KDE. Если игра не запустилась, ее следует удалить. Для этого просто удалите каталог /usr/share/wine-c/games/<название игры>. Если игра запустилась, вы должны увидеть окно эмулятора **wine** (см. рис. 21.1).

Желательно сразу же открыть окно настроек программы и поэкспериментировать с настройками видеорежимов. Например, Unreal Tournament у меня намного быстрее работал при использовании программного рендеринга (Software Rendering), чем при использовании драйвера Direct3D.

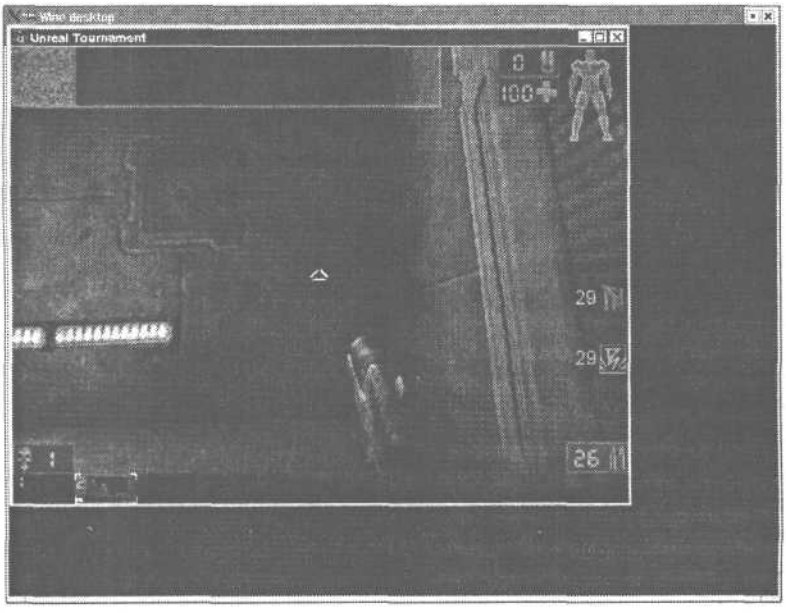


Рис. 21.1. *Окно эмулятора wine*

Теперь приступим к настройке запуска игры. Скопируйте каталог /root/.wine в каталог /root/.wine_<название игры>. Создайте файл /root/<название игры>_run:

```
touch /root/<название игры>_run
```

Содержимое этого файла зависит от эмулятора, с помощью которого запустилась игра **(wine, winex, winex2)**.

Для **wine** содержимое файла будет таким:

```
export WINEPREFIX=$HOME/.wine_<название игры>
cd "/usr/local/wine-c/games/<название игры>"
wine <исполняемый файл игры> <параметры>
```

Для **winex:**

```
export LD_LIBRARY_PATH=/usr/local/winex/lib:$LD_LIBRARY_PATH
export PATH=/usr/local/winex/bin:$PATH
export WINEPREFIX=$HOME/.wine_<название игры>
cd "/usr/local/wine-c/games/<название игры>"
winex <исполняемый файл игры> <параметры>
```

Для winex2:

```
export LD_LIBRARY_PATH=/usr/local/winex2/lib:$LD_LIBRARY_PATH
export PATH=/usr/local/winex2/bin:$PATH
export WINEPREFIX=$HOME/.wine_<название игры>
cd "/usr/local/wine-c/games/<название игры>"
winex2 <исполняемый файл игры> <параметры>
```

Введите команду для изменения прав доступа:

```
chmod u+x < название игры >_run
```

Теперь для запуска игры можно использовать команду **/root/<название игры>_run**.

После установки всех игр удалите библиотеки Microsoft, которые будут установлены в каталог /usr/local/wine-c/system. Иногда эти библиотеки устанавливаются в другие каталоги, поэтому внимательно изучите содержимое каталога /usr/local/wine-c и удалите лишние файлы.

Выполните команду **chmod -R o+w /usr/local/wine-c**. Эта команда установит права доступа к каталогу /usr/local/wine-c, в котором производятся запись в играх и сохранение конфигураций пользователям.

Для включения полноэкранного режима установите значение переменной файла /root/.wine/config **Managed,** равное N, а также закомментируйте переменную **Desktop:**

```
; Allow the window manager to manage created windows
"Managed" = "N"
; Use a desktop window of 640x480 for Wine
; "Desktop" = "800x600"
```

После того, как все будет настроено, создайте пользователя game. Используя эту учетную запись, посетители игрового зала будут регистрироваться в системе. Скопируйте все файлы настроек в каталог /home/game и установите должным образом права доступа. Для этого можете использовать следующие команды:

```
cp /root/*_run /home/game
cd /home/game
```

```
chmod o+x *_start
cd /root/Desktop/* /home/game/Desktop
chown -R game:game /home/game/Desktop
mkdir /home/game/.kde/apps/share/WINE
cp -R /root/.kde/apps/share/WINE /home/game/.kde/apps/share/WINE
chown -R game:game /home/game/.kde/apps/share/WINE
```

Теперь пользователь game сможет запускать установленные вами игры. Как всегда существует одна маленькая деталь, о которой постоянно забываешь: попробуйте объяснить посетителю, не знающему даже как правильно завершить работу в Windows 98, что такое терминал **xterm** и что для запуска игры **quake** нужно ввести команду ***quake_run***. Вы правы, это будет довольно сложно, поэтому, чтобы не усложнять себе жизнь, каждый день отвечая на вопросы наподобие: «а как запустить этот самый xterm?» и чтобы не шокировать посетителей, создайте на рабочем столе ярлыки для всех файлов `*_run`.

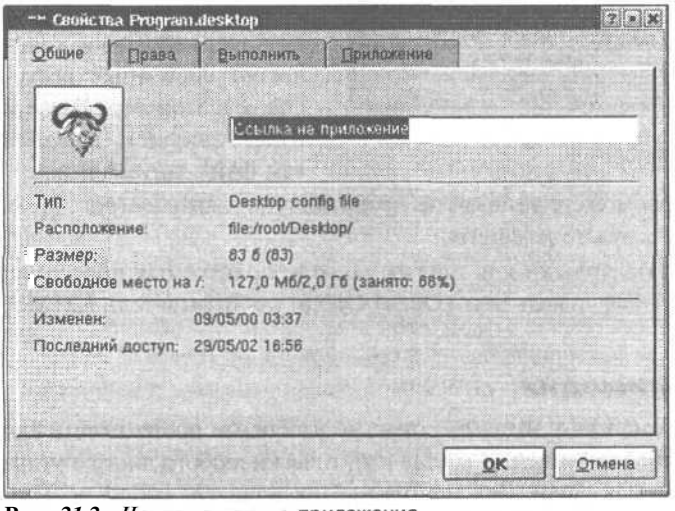


Рис. ***21.2,*** *Новая ссылка на приложение*

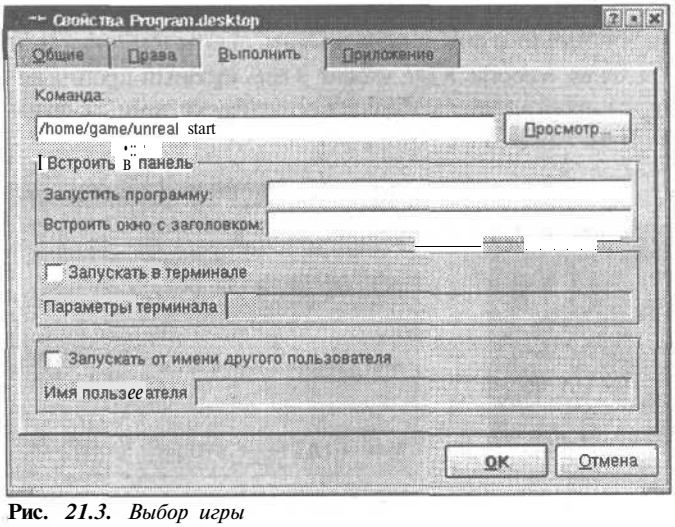


Рис. ***21.3.*** *Выбор игры*

Для этого щелкните правой кнопкой мыши на рабочем столе KDE и выберите команду **Создать→ Ссылку на приложение** (см. рис. 21.2).

В качестве рабочего стола по умолчанию я рекомендую использовать именно KDE, потому что этот рабочий стол максимально приближен к стандартному рабочему столу Windows и при работе с ним посетители будут задавать меньше вопросов.

После этого введите название игры и выберите для нее значок. После этого перейдите на вкладку **Выполнить** и выберите файл для запуска (рис. 21.3). Из рис. 21.3 видно, что при щелчке на этом ярлыке будет запущена игра **UNREAL** (файл

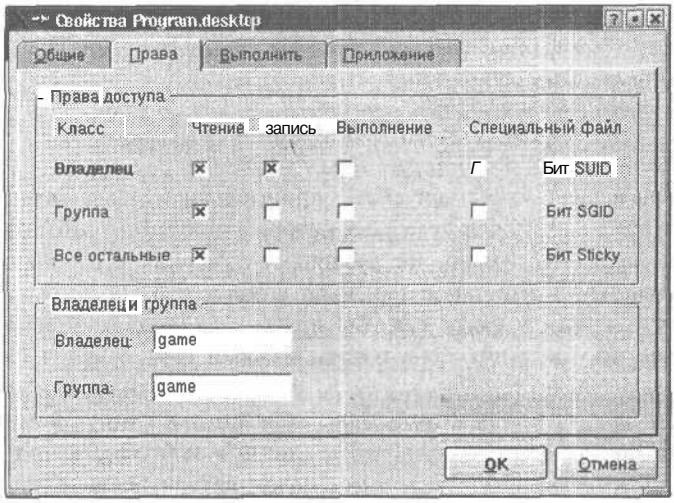


Рис. 21.4. *Права доступа*

unreal_start). Я рекомендую запускать данные файлы в терминале. Для этого можете включить режим **Запускать** в терминале и ввести параметры терминала, а можете просто ввести вместо команды **/home/game/unreal_start** команду **xterm -e Доте/game/unreal_start**. Данная команда запустит терминал xterm, который использует небольшое количество системных ресурсов, а в этом терминале и будет запущен нужный вам файл запуска игры.

На вкладке **Права** вы можете установить права доступа к ярлыку (рис. 21.4). Обычно здесь ничего не нужно изменять.

Можно также создать ярлыки для других часто используемых программ: браузер Mozilla, клиент **licq**, пакет **Star (Open) Office,** проигрыватель XMMS.

21.6. Средства мультимедиа

Операционная система Linux обладает довольно удобными программами для работы с мультимедиа-информацией. В состав практически любого дистрибутива Linux входит проигрыватель **xmms**. Эта программа представляет собой полный аналог популярной программы **winamp**, позволяет проигрывать многие типы аудио-файлов, включая MP3. Как и программа **winamp**, проигрыватель xmms позволяет подключать различные плагины и использовать скины (см. рис. 21.5).

Кроме проигрывателя **xmms,** в состав KDE входит очень удобный проигрыватель **K Media Player.** Данный проигрыватель обладает практически теми же функциями, что и **xmms** (рис. 21.6).

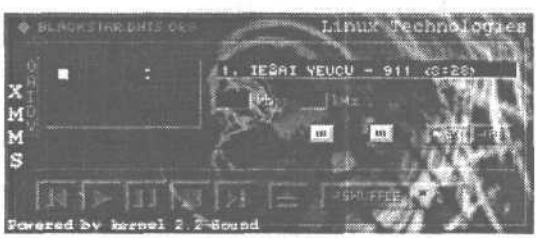


Рис. 21.5. *Проигрыватель xmms*

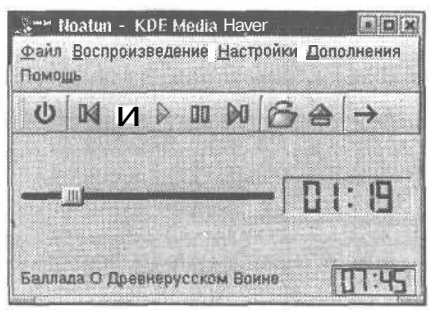


Рис. 21.6. *Проигрыватель K Media Player*

Для прослушивания аудио компакт-дисков можно использовать программу **cdplayer** (рис. 21.7). Существуют также текстовые версии этой программы — **cdp** и **cdplay**, которые предназначены для запуска из консоли.

Если вы не установили на своем сервере систему X Window, а слушать музыку все равно хочется, установите программу mpgl23. Данная программа работает в текстовом режиме и позволяет прослушивать аудиоформаты MPEG1, MPEG2, MPEG3. Поддерживаются также списки песен.

Просмотреть видеофильмы форматов VCD и MPEG вы можете с помощью проигрывателя gtv (см. рис. 21.8). Для увеличения размера окна включите режим Double, а для непрерывного воспроизведения — режим Loop.

Рис. 21.7. *Проигрыватель аудио-CD*

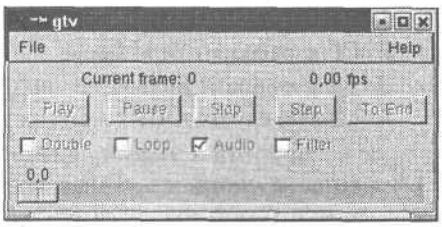


Рис. 21.8. *Проигрыватель gtv*

В большинстве случаев в вашей системе будет установлен мощный редактор MIDI-файлов Brahms. Данный редактор позволяет как создавать новые MIDI-файлы, так и редактировать уже существующие.

Настроить параметры воспроизведения, а также узнать о многих параметрах вашей аудиоподсистемы вы можете с помощью программы управления звуковым сервером aRts (см. рис. 21.9).

Настроить громкость звучания можно с помощью программы **Sound Mixer** (см. рис. 21.10). С ее помощью можно настроить как общую громкость, так и громкость отдельного аудиоустройства, а также установить баланс звучания.

Любителям Karaoke могу посоветовать программу **Media/Karaoke Player.**

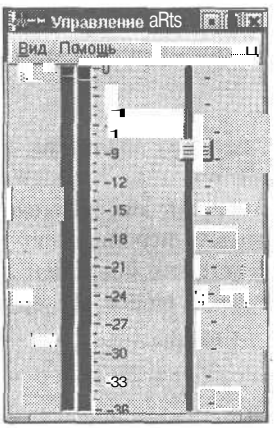


Рис. 21.9. *Программа управления звуковым сервером aRts*

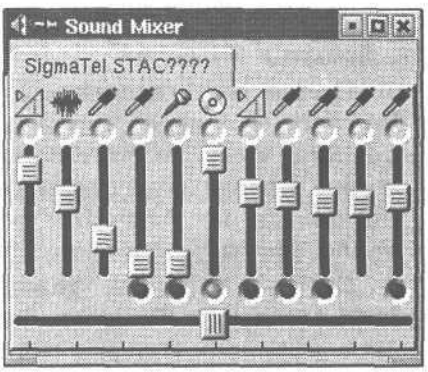


Рис. 21.10. *Программа Sound Mixer*

В состав Linux также входят программы для записи компакт-дисков **(cdrecord),** создания аудио-CD. Как видите, стандартный набор утилит для работы с мультимедиа информацией настолько широк, что операционную систему Linux нельзя назвать только серверноориентированной.

21.7. Администрирование зала

Вы уже справились с самой сложной задачей — настроили рабочее место посетителя. По сравнению с этой задачей администрирование игрового зала является второстепенным вопросом. Цель любого игрового зала — это получение прибыли, а последнее возможно лишь при условии, что:

1. Все игры будут работать, причем они должны работать быстро и без сбоев.
2. Сеть работает без сбоев.
3. Графический интерфейс пользователя интуитивно понятен.
4. Можно слушать MP3 и смотреть MP4, а также проигрывать аудио компакт-диски.

Другими словами, клиенты будут посещать ваш зал, если в нем будет создана соответствующая обстановка. А каким образом вы администрируете ваш игровой зал, посетителей мало интересует. Например, если у посетителя вышло время, можно просто подойти и сказать ему об этом. Конечно, если такое позволяют размеры вашего игрового зала. В самом деле, не будете же вы идти через весь зал, чтобы сообщить посетителю номер 47, что ему уже нужно уходить или доплатить за дополнительное время? Можно автоматизировать этот процесс и автоматически отключить его от системы через определенное время.

В этой главе я сделал все возможное, чтобы описать запуск игр под Linux, и, я надеюсь, что игры в вашем игровом зале будут работать достаточно быстро. О сети позаботится сама операционная система, как вы уже знаете, реализация стека протоколов TCP/IP в операционной системе Linux намного эффективнее, чем в Windows.

Разработчики оконных сред KDE и Gnome позаботились о интуитивности пользовательского интерфейса, решив за нас этот вопрос. А об использовании средств мультимедиа мы уже говорили в предыдущем пункте, поэтому сейчас мы можем с чистой совестью перейти к организационным моментам.

21.7.1. Доступ к Интернет

Прежде всего определимся, нужен ли вашему игровому залу доступ к Интернет. Если нужен, то для каких целей. В большинстве случаев он нужен для того, чтобы посетители могли использовать так называемые Online-версии игр или подсоединяться к всемирным игровым серверам. Такую возможность предоставляют разработчики многих современных игр. Возможно, ваш зал — это не просто игровой зал, а еще и Интернет-кафе.

Итак, мы выяснили, что может быть три варианта:

1. Зал без доступа к Интернет.
2. Зал с доступом к Интернет для Online-игр.
3. Интернет-кафе.

В первом случае вы сразу можете перейти к следующему пункту — «Управление доступом». Сейчас же будет рассмотрена настройка сервера во втором и в третьем случаях.

Если вам нужно обеспечить только работу Online-игр, вам нужно будет установить и сконфигурировать такие службы:

1. **IPChains** или **IPTables** (в зависимости от версии ядра).
2. Прокси-сервер **Socks5**.

Настраивать бастион нужно в любом случае — он обеспечивает безопасность вашей внутренней сети. При настройке бастиона учитывайте особенности используемых вами игр. Например, выделенный сервер игры **Unreal Tournament** использует 7777 порт. На бастионе нужно будет разрешить порт 7777, если вы хотите, чтобы к вашему серверу могли подключиться извне, например, из другого игрового зала. Настройка бастионов уже обсуждалась в одноименной гл. 14 — «Бастионы».

Сервер **Socks5** нужно настроить только в том случае, если ваша игра требует реальный IP-адрес. В этом случае можно использовать или IP-маскарадинг, настройка которого обсуждается в гл. 14, или сервер **Socks5**. Сервер **Socks5** нужен еще для Организации рабочего места администратора, чтобы он на протяжении рабочего дня мог общаться со своими знакомыми по ICQ.

В третьем случае (Интернет-кафе) вам нужно настроить такие службы:

1. Бастион.
2. Прокси-сервер SQUID.
3. Сервер DNS.
4. Сервер Socks5.
5. Web-сервер.
6. Почтовый сервер.

Первые три службы вам нужно настроить обязательно, а все остальные — по вашему желанию. Как уже было отмечено, бастион нужен из соображений безопасности. Сервер SQUID нужен для кэширования Web-страниц клиентов, при этом совсем не обязательно устанавливать на сервере Web-сервер.

Сервер DNS также необходим для повышения производительности. Вы можете использовать сервер DNS вашего провайдера, однако, если вы настроите собственный сервер DNS, вы:

1. Повысите скорость разрешения имен DNS.
2. Сэкономите на трафике.

Желательно настроить почтовый сервер для повышения скорости отправки сообщений посетителей. Можно опять же таки использовать или сервер провайдера или какой-нибудь бесплатный SMTP-сервер, например, smtp.mail.ru, но использование собственного сервера будет удобнее и дешевле. Напомню, что настройка почтовика обсуждалась в гл. 13, сервера DNS — в гл. 10, а сервера SQUID — в гл. 15.

Теперь, когда Интернет-сервисы уже настроены и у каждой рабочей станции есть доступ к Интернет, можно приступить к теории управления пользователями.

21.7.2. Управление пользователями

Сначала разберемся, что мы подразумеваем под управлением пользователями. Обычно все управление заключается в отслеживании времени работы посетителя и когда его время вышло, сообщении ему об этом. Естественно, если пользователей много, проследить за каждым — это довольно трудная задача. Даже если у вас будет журнал, в котором вы будете записывать время работы каждого пользователя, через пару дней вам основательно надоест каждые десять минут проверять, у какого посетителя вышло время. Например, если в вашем распоряжении 30 компьютеров, вам нужно будет каждые 10 минут просматривать все 30 записей.

Управлять пользователями можно по-разному. Можно по истечении определенного времени просто «отрубить» пользователя от системы. На что в ответ вы получите массу жалоб и вряд ли ваш зал будет пользоваться популярностью при таком управлении. Вы, конечно, можете привести аргументы в свое оправдание: мол, он (посетитель) знает, что оплатил один час и должен «чувствовать» время. Однако, в нашем случае нужно учитывать тот факт, что у игрока отсутствует это самое «чувство времени», во время игры он не ощущает, прошло полчаса или пятьдесят минут. Поэтому, скорее всего, посетитель не успеет сохранить игру до того, как его отключат от системы. Можно предупредить посетителя о таких правилах, но при этом вы заставляете его быть в постоянном напряжении, постоянно поглядывая на часы. От такой игры никто не получит удовольствие.

Мы уже пришли к выводу, что теория «жесткого» управления нам не подходит, и сейчас рассмотрим более лояльный способ управления. Через определенное время, например, за пять минут до того, как у посетителя выйдет время, мы предупредим его об этом. За это время посетитель успеет сохранить игру и доплатить, если он захочет продолжить игру. Думаю, пять минут будет вполне достаточно, чтобы дойти к столику администратора.

Все вышеописанные функции выполняются специальным программным обеспечением для игровых залов. Для игровых залов, использующих операционную систему Windows, создана масса программ такого рода. К сожалению, мне не встречался нормальный пакет программ управления игровым залом для Linux. Можно было бы использовать **K12 Linux Terminal Server,** но этот программный комплект больше подходит для управления учебным классом, чем для управления игровым залом. В нет есть много ненужных функций, которые вы вряд ли будете использовать. Вам нужна программа, которая:

1. Предупредила посетителя, что через определенное время ему нужно освободить место.
2. Через определенное время «отрубила» его от системы.
3. С помощью которой вы могли бы отправить сообщение любому посетителю.

Как видите, для вас вполне достаточно трех этих функций. Аналогичное программное обеспечение ведет также протокол: кто, когда и сколько работал. Нам же эта функция не нужна, потому что протоколы ведет сама Linux (точнее, программы протоколирования). В любой момент вы можете посмотреть, кто и сколько работал. Например, узнать, когда регистрировался и сколько времени отработал в системе пользователя den можно с помощью команды (см. рис. 21.11):

```
last den
```

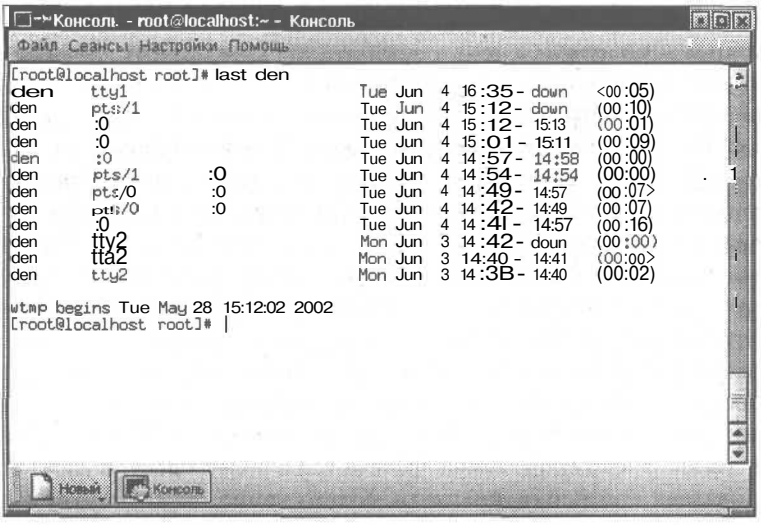


Рис. ***21.11.*** *Журнал регистрации*

Аналогично, если вы введете команду **last** без параметра, то увидите полный отчет о времени работы пользователей. Узнать время последней регистрации пользователя можно с помощью команды lastlog (рис. 21.12). Программы **last и lastlog** являются средствами просмотра файла /var/log/lastlog, который нельзя просмотреть «невооруженным глазом».

Вернемся к нашему программному обеспечению для управления посетителями. В силу невозможности найти какое-нибудь достойное уже созданное программное обеспечение, я решил написать свою «программу» для управления игровым залом. Данное решение не претендует на первое место среди программ такого рода, но обладает всеми необходимыми функциями и достаточно простое в обращении. Обычно программы такого рода состоят из двух частей: модуль-клиент и модуль-сервер. Модуль-клиент обычно установлен у администратора и он может управлять множеством компьютеров локальной сети. Модуль-сервер запускается на компьютере посетителя и опрашивает некоторый порт. Как только модуль-сервер получил от администратора команду, он выполняет определенные действия, например, при получении команды **timeout** он отсоединяет пользователя от системы.

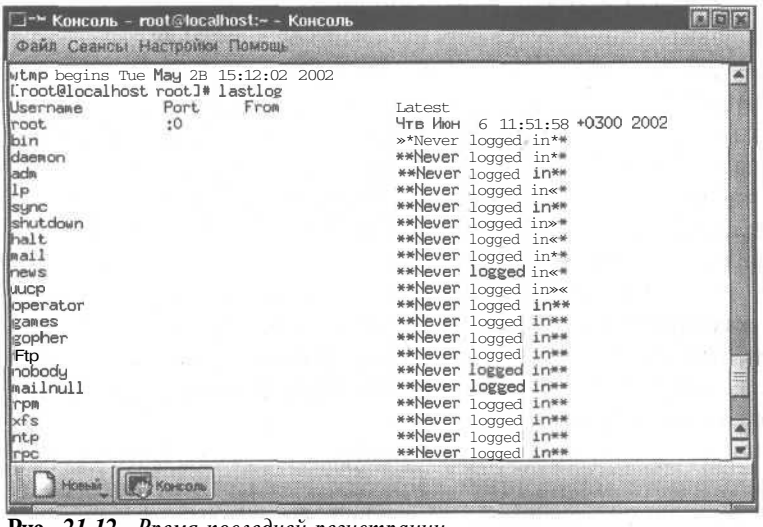


Рис. ***21.12.*** *Время последней регистрации*

В предлагаемом мною решении модуль-клиент, как и модуль-сервер, отсутствуют. Сейчас

разберемся почему. Мы настраиваем основной сервер так, чтобы к нему подключались все остальные компьютеры в сети — компьютеры посетителей. Поскольку, пользователь уже зарегистрирован в нашей системе, для того, чтобы отключить его, достаточно просто локально «прибить» процесс этого пользователя. Под процессом следует понимать оконный менеджер данного пользователя.

Естественно, все компьютеры сети будут X-терминалами вашего сервера. Настройка X-терминала обсуждалась в гл. 20. При настройке руководствуйтесь такими правилами. Имя пользователя должно совпадать с именем рабочей станции. Например, если имя рабочей станции game1, то на этой станции должен быть зарегистрирован пользователь game1. На сервере должны быть зарегистрированы все пользователи: game1, game2, ..., gameN. Пароли установите по своему усмотрению, но пароли пользователей на сервере и на рабочих станциях тоже должны совпадать. Все это необходимо для регистрации пользователя на сервере. Если настройка X-терминала показалась вам слишком сложной, сейчас рассмотрим более простой путь. В гл. 20 рассматривалась настройка «чистого» X-терминала, то есть загрузка X-терминала осуществлялась по сети, а на самом компьютере даже не был установлен жесткий диск. Сейчас же мы попытаемся настроить «условный» X-терминал. Почему условный? Операционная система будет устанавливаться на компьютеры посетителей как обычно, вместе с системой X Window. Затем в файле /etc/initab вы заменяете строку:

```
X:123456:respawn:/usr/bin/X11/X
```

на строку:

```
X:123456:respawn:/usr/bin/X11/X -query 192.168.0.1
```

Данная команда (X -query 192.168.0.1) обеспечивает загрузку системы X по умолчанию (уровень выполнения 5) и при этом будет использоваться сервер X с IP-адресом 192.168.0.1. Не сложно догадаться, что компьютер с таким адресом — это и есть ваш сервер. Настройку сервера терминалов выполните так, как описано в гл. 20. При этом на сервере и клиенте желательно установить одну и ту же версию системы X Window.

Если на всех компьютерах установлено одно и то же оборудование, а в большинстве случаев это так, поступите таким образом: настройте систему X Window только на сервере, а затем обеспечьте доступ по NFS клиентам к файлам системы X Window. В этом случае на компьютере клиента вообще не нужно устанавливать систему X Window, а запускать ее непосредственно с сервера по сети, используя NFS. Настройка сетевой файловой системы (NFS) обсуждалась в гл. 8. Я рекомендую использовать именно второй способ. Запуск игр тоже можно осуществлять по сети, предварительно расположив их в каталоге, доступному по NFS. Естественно, для запуска и нормальной работы игр по сети нужна сеть, обеспечивающая скорость передачи данных 100 Мбит/с. Концентраторы (hub) в данной сети лучше заменить коммутаторами (switch).

Теперь перейдем к написанию самой программы. Данную программу мы напишем, используя «подручные» средства: стандартные программы Linux и командный язык интерпретатора **shell.** Во-первых, командный

язык интерпретатора **bash** уже рассмотрен в этой книге. Во-вторых, если написать эту программу на C или Pascal, то читатель должен владеть данным языком программирования, что усложнит чтение книги.

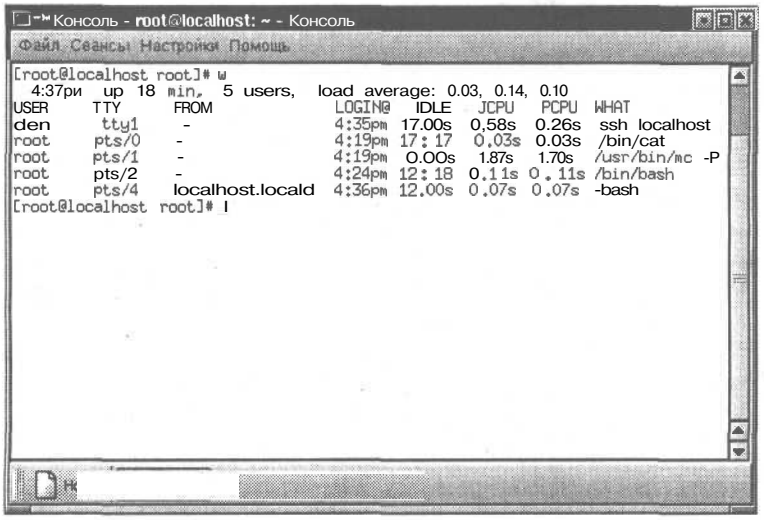


Рис. 21.13. *Команда w*

Просмотреть все зарегистрированные в системе пользователи можно с помощью команды w (рис. 21.13).

С помощью данной команды можно выяснить, сколько времени работает пользователь, использование процессора пользователем, какая программа выполняется в данный момент, а также общую загрузку системы (load average). Кроме другой полезной информации, команда w сообщает нам с какой машины произошла регистрация пользователя в нашей системе. Будем рассматривать случай, когда имя пользователя будет совпадать с именем машины, что впоследствии значительно упростит вам администрирование залом.

Вывести все процессы, которые принадлежат пользователю, можно с помощью команды:

```
ps –user username
```

Рис. 21.14. *Процессы, принадлежащие пользователю root*

На рис. 21.14. показаны процессы, принадлежащие пользователю root.

Теперь рассмотрим исходный текст этой программы.

Листинг 21.2. Программа newclient

```
#!/bin/bash
# Управление игровым залом – добавление нового клиента
# Распространяется по лицензии GPL
# (c) 2002 Denis Kolisnichenko, dhsilabs@mail.ru
# Шрифт для отображения сообщения
FONT="-cronyx-fixed-*-*-*-*-*-*-*-*-*-*-koi8-r"
# Размеры окна
GM="700x70"
# Сообщение
MSG="Ваше время вышло. В течение 5 минут вы можете оплатить
дополнительноевремя"
if [ $# -lt 4 ]; then
{ echo "Usage: newclient warntime time user num";
exit 1;
}
fi;
# Спим
sleep $1
# Отображаем предупреждение, поскольку время warntime вышло
xmessage -display server:$4 -fn $FONT -geometry $GM -bg black -fg green $MSG
sleep $2
P=`ps –user $3 | grep -i gnome-session | /bin/awk -F " " '{ print $1 }'`
echo $P
# Убиваем сессию пользователя
kill -9 $P
echo "Time of user $3 is out"
```

Запуск программы:

```
newclient 3540 3600 game1 1
newclient 55m 60m game2 2
newclient 55m 1h game3 3
newclient 23h 1d game4 4
```

При запуске программы нужно указать четыре параметра. Первый из них — это время, через которое будет отображено сообщение. Сообщение можно изменить по своему вкусу, отредактировав значение переменной MSG (так же, как и другие переменные). Следующий параметр — это время, по истечению которого пользователь будет «отрублен» от системы. Время можно указывать в секундах, в минутах (суффикс т), часах (суффикс h), днях (суффикс d).

Третий и четвертый параметры — это соответственно имя пользователя и номер дисплея. Номер дисплея будет совпадать с номером пользователя, если вы настроите систему согласно моим рекомендациям. Например, если имя пользователя gamel, то номер дисплея — 1.

Обратите внимание на следующую строку программы:

```
xmessage -display server:$4 -fn $FONT -geometry $GM -bg black -fg green $MSG
```

Данная строка обеспечивает отображение сообщения MSG на дисплее с номером $4 компьютера server. X-терминал посетителя будет подключен как раз к дисплею с номером $4. При этом посетитель увидит на экране примерно то, что показано на рис. 21.15.

Рис. 21.15. *Предупреждение об истечении времени*

Следующий аспект, на который вам нужно обратить внимание — это оконная среда пользователя. Если пользователь использует среду Gnome, то в списке процессов пользователя будет процесс gnome-session. Если завершить этот процесс, пользователь будет отключен от системы. На этом и основывается данный метод работы программы. В листинге 21.2 подразумевается, что пользователь использует среду Gnome:

```
P='ps –user $3 | grep -i gnome-session | /bin/awk -F " " '{print $1 }' '
```

Если ваши посетители используют среду KDE, измените эту строку на аналогичную ей:

```
P='ps -user $3 | grep -i kdeinit | /bin/awk -F " " '{ print $1 }' '
```

Как проконтролировать, какую среду использует посетитель? Очень просто: при установке системы установите одну из сред: или KDE, или Gnome. Можно также изменить исходный текст программы newclient, добавив соответствующую проверку, но зачем усложнять себе жизнь?

Еще раз рассмотрим запуск программы. Данную программу можно запускать в фоновом режиме, освободив консоль:

```
newclient 55m 60m game1 1 &
```

Эта команда будет выполняться в фоновом режиме. Как только выйдет время (1 час), на консоли вы увидите сообщение:

```
Time of user game1 is out
```

21.7.3. Ограничение доступа пользователя

Операционная система Linux обладает достаточно высокими средствами защиты информации, поэтому, используя стандартную конфигурацию (обыкновенный пользователь, а не суперпользователь), вы обеспечите высокий уровень безопасности. Другими словами, по поводу безопасности можете не волноваться: пользователь все равно ничего не сделает такого, что может повлечь за собой разрушение системы. Единственное, что я могу порекомендовать, переместите файлы /usr/bin/mc и /usr/bin/kcontrol-panel в каталог пользователя root: посетителю незачем изучать аналог Norton Commander для Linux и тем более настраивать среду KDE.

22 Антивирусная защита

22.1. Антивирусные программы

Можно сказать, что популярность той или иной операционной системы определяет количество вирусов, которые предназначены для этой системы. Действительно, чем распространеннее операционная система, тем больше вирусов для нее написано. И это вполне оправдывает логику вирусописателей: никто не будет писать вирус для операционной системы, если ее используют всего несколько десятков или сотен человек.

С ростом популярности операционной системы Linux увеличился и ее «вирусный рейтинг». Если раньше считалось, что не существует вирусов, «обитающих» в Unix-подобной среде, то в последнее время ситуация несколько изменилась. Одними из самых распространенных вирусов для Linux являются: Bliss, Lion, **Nuxbee**, **Ramen**, **RST** и Satyr. Большинство вирусов для Linux являются «червями», то есть сетевыми вирусами, которые распространяются по сети с помощью тех или иных сетевых служб, например, службы электронной почты. Я не буду описывать каждый из вирусов — об этом вы можете прочитать на сайте http://www.avp.ru.

Самыми популярными отечественными антивирусными программами являются AVP и DrWeb. Последняя известна еще со времен DOS-подобных операционных систем. Разработчики данных программ выпустили версии AVP («Лаборатория Касперского», http://www.avp.ru) и DrWeb («Лаборатория Данилова», http://www.drweb.ru) для Linux.

Обе программы являются мощным барьером для всех типов вирусов, но я почему-то предпочитаю AVP, хотя в книге будет описана работа с обоими антивирусами.

Программа **AVP** для Linux (как и DrWeb для Linux) помимо специфических для Linux вирусов «знает» также вирусы для операционной системы Windows. Конечно, в Linux эти вирусы работать не будут, но данная возможность очень полезна при установке антивируса на почтовом сервере: ведь инфицированные файлы могут быть отправлены локальным пользователям, большинство из которых работает в операционной системе Windows. Кроме того, обе программы обладают функцией эвристического анализа, которая позволяет находить вирусы, которые еще не известны программе, а имеют черты, свойственные вирусам.

22.1.1. Программа DrWeb для Linux

Теперь перейдем непосредственно к процессу антивирусной проверки. Сначала будет рассмотрен **DrWeb,** а затем — **AVP**. В следующем п. 22.2 будет рассмотрена антивирусная проверка входящей электронной почты.

Установка DrWeb ничего необычного в себе не таит:

```
rpm -ihv drwebd-4.27-linux.i386.rpm
```

Естественно, номер версии у вас будет отличаться. Самую последнюю версию вы можете достать на сайте лаборатории Данилова — www.drweb.ru. Программа **DrWeb** обычно устанавливается в каталог /opt/drweb. Затем вы должны получить ключ **drweb32.key**, другими словами, купить ключ. Это тоже можно сделать на вышеуказанном сайте. Конечно, вы можете достать ключ любым другим способом, но это уже ваши заботы. Данный ключ вам нужно скопировать в каталог /opt/drweb, а потом файл drweb32.key скопировать в файлы drweb.key и drwebd.key:

```
cp ./drweb32.key /opt/drweb
cp /opt/drweb/drweb32.key /opt/drweb/drweb.key
cp /opt/drweb/drweb32.key /opt/drweb/drwebd.key
```

Затем откройте в любом текстовом редакторе файл /opt/drweb/drweb.ini и установите нужные вам параметры. Я рекомендую установить следующий параметр:

```
OutputMode = Terminal
```

По умолчанию данный параметр имеет значение **Color,** но лучше все-таки установить значение **Terminal.** При этом обеспечивается более «удобоваримый» вывод на терминал.

Теперь можно проверить работу антивируса:

```
/opt/drweb/drweb /root/cih.exe
```

Как видно из отчета (см. рис. 22.1), антивирус **DrWeb** проверил один файл /root/cih.exe и нашел в нем вирус Win95.CIH.1035. **DrWeb** не вылечил ни одного файла (Cured: 0), не удалил ни одного файла (Deleted: 0), не переименовал и не переместил (Renamed: 0 и Moved: 0). Проверка не заняла много времени (0 секунд), а скорость сканирования составила 45 Кб/с.

Если у вас нет вируса для тестирования, вы можете воспользоваться стандартным файлом проверки — test.com. Если почему-то данный файл у вас отсутствует, создайте текстовый файл и добавьте в него всего одну строку:

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Затем сохраните его под именем test.com. В результате проверки данного файла вы должны получить сообщение:

```
EICAR Test File (Not a Virus!)
```

Последнее, что вам осталось настроить — это автоматическое обновление антивирусных баз данных. Антивирусные базы данных содержат образцы вирусов, с помощью которых антивирус идентифицирует тот или иной тип вируса. Эти базы доступны для всех пользователей по адресу: http://www.dials.ru/drweb/free.

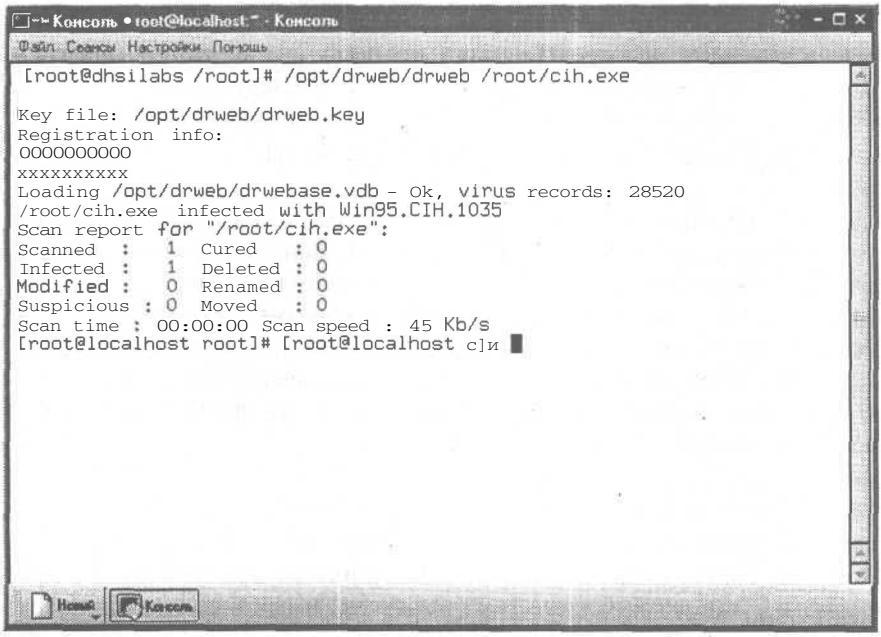


Рис. 22.1. Антивирус DrWeb

Однако обновлять базы вручную — это занятие неблагодарное, поэтому настройте свою систему так, чтобы она самостоятельно обновляла антивирус. Специально для этой цели существует модуль **update.pl.** Он должен быть установлен в каталоге `/opt/drweb/update`.

Запускать данный модуль нужно так:

```
./update.pl /opt/drweb
```

`/opt/drweb` — это каталог, в который следует поместить обновленные базы. Естественно, для автоматического обновления у вас должно быть установлено соединение с Интернет до начала обновления. Лучше всего, если у вас выделенная линия: тогда вы вообще забудете что такое процесс обновления антивируса.

Если обновление было произведено успешно, добавьте запуск данного модуля в сценарий автозагрузки системы. Для этого создайте файл update_drweb:

```
#!/bin/sh
/opt/drweb/update/update.pl /opt/drweb
```

и поместите его в каталог `/etc/cron.daily`. Естественно, этот файл нужно сделать исполнимым. Теперь вы можете быть уверенны в том, что у вас самая свежая антивирусная база.

В сценарии автозагрузки можно добавить запуск демона **drwebd**. Перед этим следует убедиться, что демон нормально работает. Запустите демон вручную:

```
# /opt/drweb/drwebd
```

Вы должны увидеть примерно следующее:

```
Key file: /opt/drweb/drwebd.key
Registration info:
0000000000
xxxxxxxxxx
Loading /opt/drweb/drwebase.vdb — Ok, virus records: 28520
Daemon is installed, TCP socket created on port 3000
```

Данное сообщение свидетельствует о том, что демон функционирует нормально. Вот теперь добавьте запуск демона **drwebd** в сценарии автозагрузки системы.

22.1.2. Программа AVP для Linux

Программа AVP обнаруживает и удаляет следующие типы вирусов:

1. Стелс-вирусы (невидимки).
2. Полиморфные вирусы.
3. Вирусы, предназначенные для запуска в операционных системах Windows 9x, Windows NT, Linux/UNIX, OS/2.
4. Макро-вирусы, заражающие документы MS Office.
5. Вирусы для Java-апплетов.
6. «Троянские кони».

Для установки данной программы вам необходим компьютер с процессором 386 (или старше) и 8 Мб оперативной памяти (рекомендуется 16 Мб). Естественно, на компьютере должна быть установлена операционная система Linux.

Вставьте инсталляционный компакт-диск с антивирусом в CD-ROM и смонтируйте его командой:

```
mount -t iso9660 /dev/hdd /mnt/cdrom
```

где /dev/hdd — это имя устройства CDROM.

Перейдите в каталог /mnt/cdrom/Products/KAVLinux. В этом каталоге должен находиться файл AVPWSELinux.tgz. Скопируйте его в какой-нибудь каталог вашей файловой системы. Если для данного файла установлено право на исполнение, права доступа следует изменить, например:

```
chmod 444 AVPWSELinux.tgz
```

Затем распакуйте файл командой:

```
tar zxvf AVPWSELinux.tgz
```

В результате распаковки в текущем каталоге появятся два новых файла: AVPInstaller и AVPWSLinux.tgz. Для начала установки введите команду:

```
./AVPInstaller
```

Появится приглашение с выводом номера версии устанавливаемого продукта. Я использовал версию AVP 3.0 build 135.1. Вы должны увидеть примерно это сообщение:

```
I think you use Redhat-like system. You didnt write package in
command line. Search in current directory?
```

Таким образом программа установки сообщает вам тип вашей операционной системы (Redhat-совместимый) и предлагает найти пакет с установочными файлами *в* текущем каталоге, на что вы должны ввести в ответ «у»,

то есть согласиться. Затем программа установки спросит вас, не хотим ли мы установить AVP:

```
In package /KAVLinux/AVPWSELinux/AVPWSLinux.tgz found AVPWSLinux
(Antiviral Toolkit Pro for Linux) (Foundation files)), version
3.0 build 135. Do you want to install it?
```

Введите в ответ «у» и нажмите Enter. Установщик распакует необходимые файлы и спросит вас, не хотите ли вы запустить AVP для Linux:

```
Do you want run /opt/AVP/AvpLinux.
```

Пока запускать AVP рановато: мы еще не подготовили к первому запуску. Если вы попробуете сейчас запустить его, получите много сообщений об ошибках, а потом сканер завершит свою работу. Следует сказать, что все файлы будут установлены в каталог /opt/AVP.

Прежде всего нужно скопировать ключевой файл. Этот файл должен находиться на дискете, которая входит в ваш лицензионный комплект установки. Смонтируйте дискету:

```
mount /dev/fdO /mnt/floppy
```

На дискете вы найдете один-единственный файл с расширением key. Скопируйте его в файл /opt/AVP/AvpUnix.key. При копировании обратите внимание на регистр букв в имени файла. Ключевой файл должен называться именно **AvpUnix.key**, а не как-либо иначе.

Затем откройте в любом текстовом редакторе файл /opt/AVP/AvpUnix.ini (см листинг 22.1).

Листинг 22.1. Файл AvpUnix.ini

```
[AVP32]
DefaultProfile=defUnix.prf
LocFile=None
[Configuration]
KeyFile=AvpUnix.key
KeysPath=/opt/AVP
SetFile=avp.set
BasePath=/opt/AVP
SearchInSubDir=No
```

В файле defUnix.prf содержатся настройки пользователя (профиль пользователя). Редактировать эти настройки рекомендуется с помощью программы **AVPTuner**, которая будет рассмотрена немного позже.

Ключевой файл называется AvpUnix.key. Путь, по которому расположены антивирусные базы, определяется параметром BasePath. Лучше не изменять данное значение. Лично у меня, когда я изменил значение параметра BasePath, **AvpLinux** продолжал искать антивирусные базы в локальном каталоге. Если вы все-таки изменили путь к базам, укажите полный путь к файлу avp.set. В этом файле прописаны все подключаемые базы. При этом подразумевается, что базы и файл avp.set расположены в одном и том же каталоге. Например, если вы решили установить базы в каталог /opt/AVP/Bases, внесите следующие изменения в файл конфигурации:

```
[Configuration]
KeyFile=AvpUnix.key
KeysPath=/opt/AVP
SetFile=/opt/Bases/avp.set
BasePath=/opt/AVP/Bases
SearchInSubDir=No
```

Естественно, файл антивирусных баз должен быть распложен в каталоге /opt/AVP/Bases. Для первого запуска воспользуйтесь базами, которые расположены на компакт-диске в каталоге /mnt/cdrom/Bases, а позже займемся их обновлением. Если вы скопировали базы из раздела Windows, внимательно проследите за регистром букв в имени файла. Иногда случается такое, что при копировании файлов из раздела Windows первая буква названия файла является прописной. Переименуйте такие файлы в соответствии с файлом avp.set, иначе сканер **AvpLinux** не будет работать (точнее, не найдет нужные ему базы). При работе с операционной системой Linux всегда следует помнить о регистре букв в именах файлов.

Теперь можно приступить к первому запуску программы. Введите команду (см. рис. 22.2):

```
/opt/AVP/AvpLinux/home/den
```

После завершения процесса сканирования вы увидите отчет о проверенных объектах (см. рис. 22.3).

Как видно из рис. 22.3, было проверено 290 файлов и 87 каталогов. Скорость сканирования — 78 Кб/с, а время сканирования — 10 секунд. Не было найдено ни вирусов, ни «подозрительных» объектов (Suspicious).

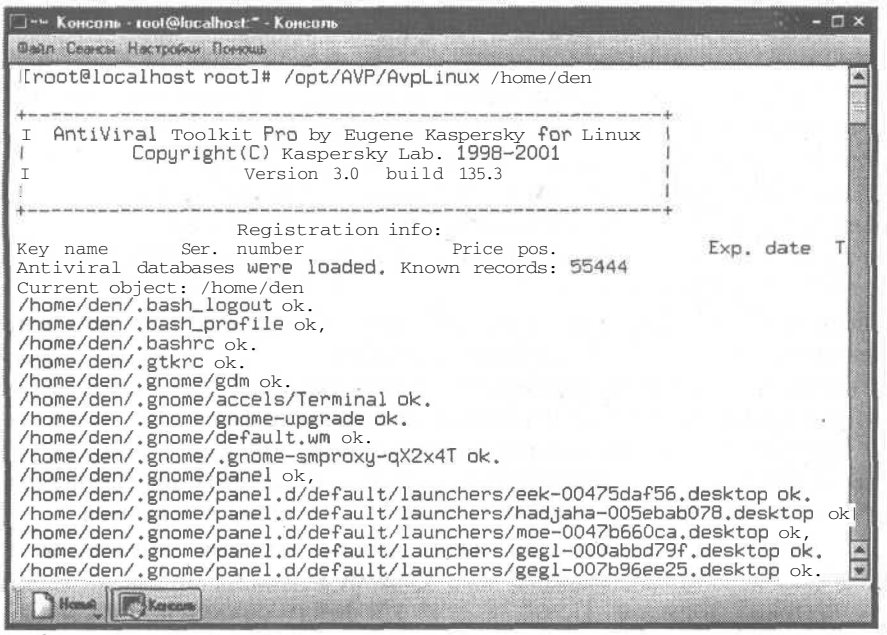


Рис. 22.2. *Антивирус AVP*

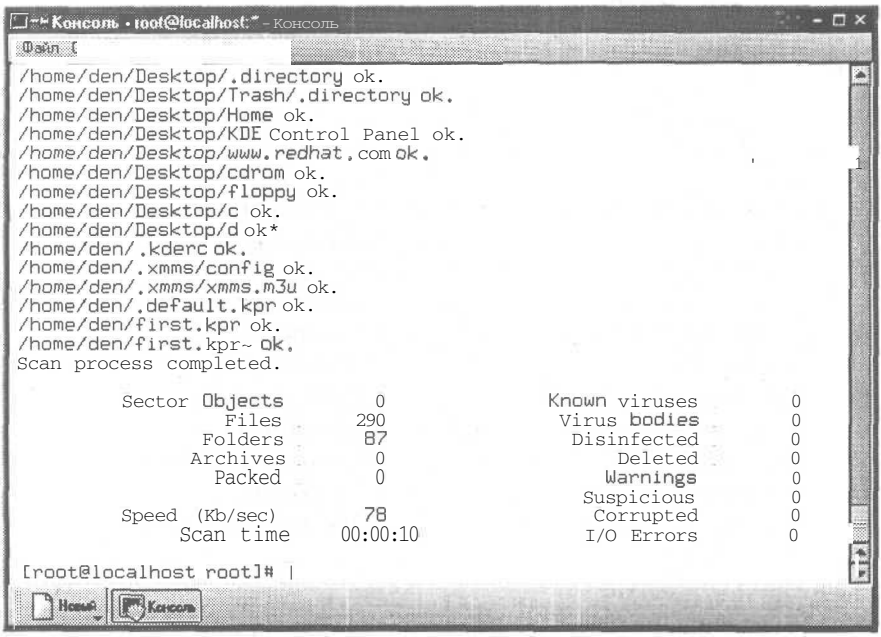


Рис. 22.3. *Отчет AVP*

Если будет найден инфицированный файл, программа предложит вам выполнить следующие действия:

Report Only (Ok).....ничего не делать (только отчет).

disinfect.....попытаться вылечить файл.

Delete.....удалить.

Чтобы выбрать какое-нибудь действие, нажмите «Enter», «d» или «D» соответственно. Лечение файлов не всегда возможно, поэтому, если AVP не вылечил ваш файл, вам придется удалить его.

Случается и такое, что вирус поражает программу-сканер, то есть **AvpLinux**. В этом случае рекомендуется восстановить AVP из дистрибутива. В самом крайнем случае (если нет дистрибутива) можно попытаться вылечить файл, выбрав действие **disinfect.**

Иногда вы можете получить сообщение, что файл **AvpLinux** поврежден или заражен неизвестным вирусом. Это может быть вызвано тем, что вы (или кто-то другой) переименовали файл **AvpLinux** и пытаетесь запустить переименованную версию. В старых версиях AVP для Linux имя исполнимого файла было «зашито» в саму программу, поэтому обнаружив при запуске, что имя отличается от исходного, AVP выдавал такое сообщение, несмотря на это, что файл был целым и невредимым. Как правило, в новых версиях AVP эта ошибка устранена.

Программа AVP для Linux не лечит файлы, которые запакованы в архивы, файлы, инфицированные «троянскими конями», почтовые базы, файлы почтовых форматов. Поэтому, если эти файлы окажутся зараженными, их нужно будет удалить.

Кроме обыкновенных файлов и каталогов, можно также сканировать загрузочные сектора жесткого диска. Для включения режима сканирования MBR-сектора используется ключ «-P-». Для включения режима сканирования Boot-секторов используется ключ «-B-». Для отключения данных режимов используются ключи «-P» и «-B» соответственно. При попытке лечения зараженного сектора AVP перезапишет инфицированный сектор стандартным загрузочным сектором операционной системы DOS версии 6.0, поэтому будьте очень внимательны при лечении секторов — после такого «лечения» вы не сможете вообще запустить операционную систему. В этом случае нужно сделать резервное копирование всех файлов, а потом уже приступать к лечению. Полезно будет также создать загрузочную дискету Linux. Обычно она создается при установке системы, но ее никогда не поздно сделать с помощью программы **qmkbootdisk**.

Теперь перейдем к демону AVP. В отличие от сканера, демон **AvpDaemon** загружет антивирусные базы в память всего один раз -- при запуске. В результате чего значительно сокращается время проверки объектов. Эта особенность определяет сферу применения демона AVP: почтовые, файловые и Web-серверы, где требуется быстро проверять поступающие объекты.

При установке в каталоге `/etc/init.d` будет создана ссылка **avpdaemon** на файл `/opt/AVP/avpdaemon.rh`, если вы используете Redhat-подобную операционную систему. Поэтому демон AVP будет загружаться автоматически при запуске системы. Сейчас можно выполнить его запуск вручную, введя команду:

```
/etc/init.d/avpdaemon start
```

Определить всевозможные параметры антивируса AVP можно с помощью программы **AVPTuner**. Запустите ее командой: `/opt/AVP/AVPTuner` (см. рис. 22.4).

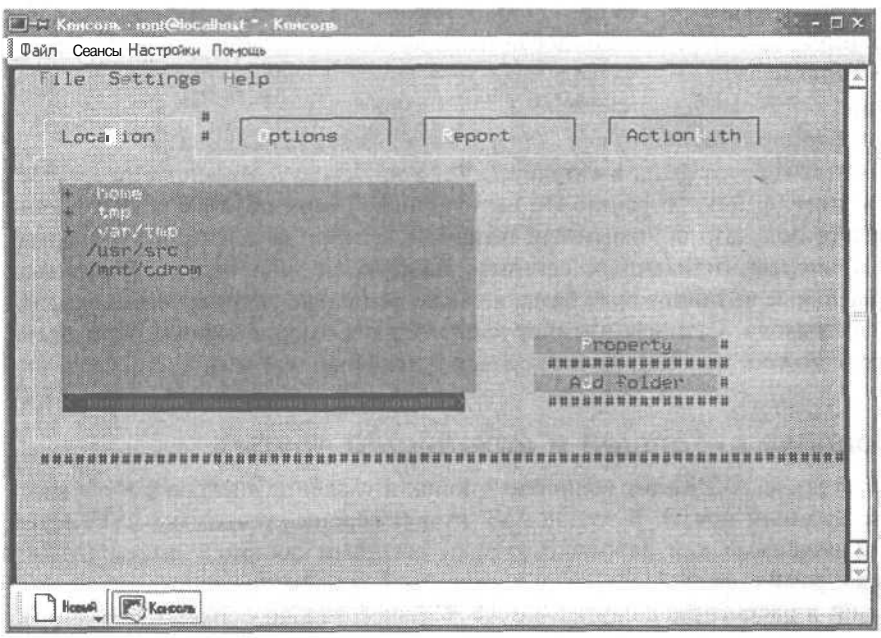


Рис. 22.4. *Программа AVPTuner*

После запуска данной программы вам будут доступны следующие закладки: Location, Objects, Scanning, Actions, Options, Mail.

На закладке Location задаются каталоги, которые бы вы хотели проверить. Добавить еще один каталог можно, нажав на кнопку «Add folder». Знак плюса возле имени каталога означает включение его в область сканирования, минуса — исключения его из области сканирования. Изменить режим можно с помощью клавиши «Пробел». Для каждого каталога можно задать режим проверки, а также файлы, которые нужно проверить (программы, все файлы, по желанию пользователя). При выборе файлов для сканирования я рекомендую использовать режим Smart. При этом режиме проверяются все файлы, способные содержать код вируса.

На закладке Objects указываются все типы объектов, которые вы хотите проверить на наличие вирусов.

Параметры сканирования можно задать на закладке Scanning, а на закладке Actions можно определить вид действия при обнаружении инфицированного объекта.

На закладке Options можно указать дополнительные параметры сканирования. Я рекомендую включить режим эвристического анализа (Code Analyzer). Режим избыточного сканирования (Redundant Scanning) лучше не включать, поскольку значительно увеличивается время сканирования.

На закладке Mail можно включить режим отправки отчетов с результатами сканирования администратору или любому другому пользователю.

Автоматическое обновление антивирусных баз происходит с помощью программы **AvpUpdater**. Для автоматического обновления через Интернет используйте команду:

```
/opt/AVP/AvpUpdater -ui=ftp://ftp.kasperskylab.ru/updates
```

Данную команду полезно добавить в расписание демона crond. Создайте файл update_avp:

```
#!/bin/sh
/opt/AVP/AvpUpdater -ui=ftp://ftp.kasperskylab.ru/updates
```

Поместите этот файл в каталог `/etc/cron.daily`. Измените права доступа к этому файлу, разрешив его выполнение. Таким образом, каждый день вы будете получать обновленный файл `daily.avc`, содержащий информацию о вирусах, найденных сегодня. Каждую неделю будут обновляться еженедельные антивирусные базы, а также основные антивирусные базы по мере их выхода. Размеры антивирусных баз небольшие (кроме основных), поэтому можете особо не беспокоиться о трафике.

22.2. Проверка входящей и исходящей почты

Программа **AVPKeeper** выполняет поиск и удаление вирусов в сообщениях электронной почты. В состав AVP входят версии программы AVPKeeper, предназначенные для работы с такими агентами доставки почты (MTA): **sendmail**, postfix, **qmail**. Поскольку в книге рассматривается только программа sendmail, я рассмотрю сопряжение AVPDaemon только с этой программой.

AVPKeeper может работать в двух режимах: локальном и глобальном. Давайте разберемся, для чего предназначены эти режимы. Первый подойдет только в случае, если вы администрируете небольшой почтовик, который обслуживает небольшое количество пользователей. При выборе локального режима работы будут проверяться только входящие сообщения, то есть только те, которые приходят извне (из Internet) нашим пользователям. В этом случае предполагается, что в нашей сети нет вирусописателей, которые распространяют вирусы.

Глобальный режим работы больше подходит для почтовых серверов больших организаций. При этом режиме проверяется не только входящая почта, но и исходящие сообщения: вдруг среди наших пользователей есть распространители вирусов.

Надежнее работает локальный режим, поэтому мы будем использовать именно его. Принципиально большой разницы в настройках нет: скоро вы сами убедитесь в этом.

Начнем с установки программы AVPKeeper. В состав обычного дистрибутива AVP, который мы установили ранее эта программа не входит. Для ее установки скопируйте файл `kavselinux.tgz` из каталога `/mnt/cdrom/Products/KAVLinux` в какой-нибудь каталог вашей файловой системы. Затем установите нужные права доступа (нужно сбросить право на выполнение) и распакуйте его:

```
chmod 444 kavselinux.tgz
tar zxvf kavselinux.tgz
```

После этого запустите инсталлятор **kavinstaller**. В процессе установки ВНИМАТЕЛЬНО отвечайте на задаваемые инсталлятором вопросы. Во-первых, инсталлятор обнаружит, что уже установлена предыдущая версия AVP и спросит, что с ней делать. Нужно ответить «Оставить», чтобы предыдущая версия осталась на диске. Затем инсталлятор спросит вас, какие версии **AVPKepper** нужно устанавливать. Ответить Да (y) нужно только на вопрос:

```
В пакете /KAVLinux/kavs/kavkeeper-sendmail-linux-3.5.136.tgz обнаружен
kavkeeper-sendmail (KAV Keeper for sendmail), версия 3.5 build 135. Do you
want to install it? y
```

Затем вам будет задана серия вопросов о том, что делать со старыми файлами:

```
Found [file] What do you want doing with this file? (Overwrite/Write with
new extension/Skip)
```

Нужно ответить **w** — перезаписать с новым расширением. Потом, в случае некорректной установки, у вас будет возможность все восстановить.

После установки нового антивируса — KAV (Kaspersky AntiVirus) названия сканера, демона, программы настройки изменятся. Для сканирования нужно использовать программу kavscanner, для настройки -- **kavtuner**. В качестве демона будет использоваться kavdaemon. Для автоматического обновления теперь нужно использовать программу **kavupdater**. Две программы — AVP и KAV — прекрасно «уживаются» на одном компьютере: поэтому я настоятельно не рекомендую удалять предыдущую версию.

Теперь перейдем к настройке **sendmail**. Прежде всего, остановите **sendmail** командой:

```
/etc/init.d/sendmail stop
```

После этого перейдите в каталог `/opt/AVP/kavkeeper` и скопируйте каталог `sendmail-cf` в каталог `/usr/share`. Затем выполните одну из следующих команд:

```
m4 kav_glb.mc > /etc/sendmail.cf
m4 kav_loc.mc > /etc/sendmail.cf
```

Первую команду нужно выполнить, если вам нужен глобальный режим работы программы **AVPKeeper**, а вторую - - если необходим локальный режим. Как видите, нет большой разницы в настройке режимов AVPKeeper.

Примечания.

1. Перед выполнением этих команд скопируйте куда-нибудь старый `/etc/sendmail.cf` — он вам еще пригодится.

2. Файлы `kav_glb.mc` и `kav_loc.mc` нужно использовать только для экспериментального запуска. После того, как все нормально будет работать, отредактируйте данные файлы и внесите параметры, специфичные для вашей машины. Конечно, если вы когда-нибудь устанавливали такие параметры. В большинстве случаев данные файлы подойдут для большинства машин.

В качестве примера приведу листинг файла локального режима работы `kav_loc.mc` (см. листинг 22.2).

Листинг 22.2. Файл kav_loc.mc

```
divert(-1)
dnl This is the macro config file used to generate the /etc/
sendmail.cf
dnl file. If you modify thei file you will have to regenerate
the
dnl /etc/sendmail.cf by running this macro config through the
m4
dnl preprocessor:
dnl
dnl m4 /etc/sendmail.mc > /etc/sendmail.cf
dnl
dnl You will need to have the sendmail-cf package installed for this to
dnl work.
include('../m4/cf.m4')
define('confDEF_USER_ID',"8:12")
OSTYPE('linux')
undefine('UUCP_RELAY')
undefine('BITNET_RELAY')
define('confAUTO_REBUILD')
define('confTO_CONNECT', '1m')
define('confTRY_NULL_MX_LIST',true)
define('confDONT_PROBE_INTERFACES',true)
dnl define('KAVKEEPER_MAILER','/usr/local/bin/kavkeeper')
dnl define('KAVKEEPER_CONFIG','/etc/kavkeeper.ini')
```

```
dnl define( 'KAVKEEPER_LOCAL_MAILER','mail.local')
FEATURE('smrsh','/usr/sbin/smrsh')
FEATURE(mailertable)
FEATURE ('virtusertable', 'hash -o /etc/mail/virtusertable')
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE (use_cw_file)
FEATURE(local_kav)
MAILER(smtp)
FEATURE('access_db')
FEATURE('blacklist_recipients')
dnl We strongly recommend to comment this one out if you want
dnl to protect
dnl yourself from spam. However, the laptop and users on
dnl computers that do
dnl not hav 24x7 DNS do need this.
FEATURE('accept_unresolvable_domains')
dnl FEATURE('relay_based_on_MX')
```

Если вас по каким-либо причинам не устраивает стандартный файл kav_loc.mc, например, вы используете специфические настройки, внесите строки, выделенные жирным шрифтом, в свой **mc-файл** и выполните команду:

```
m4 mysystem.mc > /etc/sendmail.cf
```

При использовании глобального режима, в свой mc-файл нужно внести следующие строки:

```
dnl define('KAVKEEPER_MAILER','/usr/local/bin/kavkeeper')
dnl define('KAVKEEPER_CONFIG','/etc/kavkeeper.ini')
define('KAV_LOCAL_HACK')
dnl define('confDEF_USER_ID','kavuser:kavuser')
dnl define('confRUN_AS_USER','kavuser')
dnl define('KAVKEEPER_MAILER_FLAGS', 'APhnu9')
MAILER(kavkeeper)
```

Все! Настройку **sendmail** можно считать завершенной. Осталось только проверить, как все работает. Запустите **sendmail (/etc/init.d/sendmail start)** и выполните команду:

```
uuencode /root/kern386.exe kern386.exe | mail -s Just_Run_It evg
```

Этой командой файл, инфицированный вирусом Win95.CIH, будет отправлен локальному пользователю evg. Сейчас начинается самое интересное. Проверьте свою почту (см. файл /var/mail/root). Должно быть что-то типа того, что представлено в листинге 22.3.

Листинг 22.3. Сообщения о найденных вирусах

```
Return-Path: o
From: root@domain.ru
To: root@localhost.localdomain
Subject: SENDER ! Virus found in message from you !
MIME-Version: 1.0
Content-Type: text/plain; charset="US-ASCII"
```

```
You sent to user evg message with VIRUS
=======================================
KAV Report:
=======================================
kern386.exe infected: Win95.CIH.1035
=======================================
Bye !

Return-Path:<>
From:root@domain.ru
To:root@domain.ru
Subject:ADMIN ! ALARM ! Virus found !
MIME-Version:1.0
Content-Type:multipart/mixed;
    boundary="=NEXT=AVPCHECK=2002=184=1025707050=1225=0="

This is a MIME-encapsulated message

--=NEXT=AVPCHECK=2002=184=1025707050=1225=0=
Content-Type:text/plain
Content-Transfer-Encoding:US-ASCII

User root@localhost.localdomain send to user evg. mail with virus.
------------------

KAV report:
------------------

kern386.exe infected: Win95.CIH.1035
------------------
```

Первое сообщение говорит о том, что письмо, содержащее вирус, было успешно отправлено, но оно не было доставлено адресату. Второе сообщение информирует администратора системы, что локальному пользователю evg пришло письмо, содержащее вирус.

В файле протокола `/var/log/kavkeeper-` [date] `.log` вы также найдете сообщения о вирусе.

Программу **AVPKeeper** можно настроить по-разному: для автоматического удаления вирусов и удаления вирусов вручную. В первом случае пользователь, отправивший сообщение с вирусом, получает уведомление об этом, сообщение о найденном вирусе направляется администратору, а само сообщение (вместе с ним и вирус) удаляется. Во втором случае происходит все так же, как и в первом, но сообщение не удаляется, а переадресовывается администратору. Второй режим рекомендую использовать, если у вас уйма свободного времени и вашим хобби является исследование вирусов. Эти режимы можно задать в файле `kavkeeper.ini`. Более подробную информацию вы можете получить, прочитав документацию на программу **AVPKeeper.**

23

Прочие возможности

23.1. SATAN

Нет, в этой главе мы будем говорить не о религии. Программа SATAN, как могло вам показаться с первого взгляда, ничего общего с религией не имеет. SATAN (Security Administrator Tool for Analyzing Networks) — это утилита для анализирования сети и выявления дыр в различных узлах. SATAN представляет собой мощный сетевой сканер, который сканирует порты всех компьютеров вашей (и не только вашей) сети и информирует вас о возможной дыре в системе безопасности того или иного узла.

Итак, давайте приступим к установке программы. Сразу же скажу, что если вы не знакомы с программированием на C, вам будет трудно установить эту программу. Данная программа распространяется в исходных текстах. Никогда нельзя быть уверенным в том, что программа, прекрасно работающая на FreeBSD, откомпилируется и будет корректно работать в Linux, несмотря на мобильность языка C. Все же, если вы обладаете хотя бы небольшими навыками в программировании, можно изменить исходные тексты, «заточенные» под FreeBSD так, чтобы они смогли работать в Linux.

В Интернет можно найти «пропатченную» версию SATAN для Linux, однако эта версия еще хуже собирается, чем версия для FreeBSD. Мы же поступим следующим образом: скачайте обе версии — для FreeBSD и для Linux. Версию для Linux можно загрузить по адресу:

```
http://www.ibiblio.org/pub/packages/security/Satan-for-Linux/
satan-1.1.1.linux.fixed2.tgz
```

Версия для FreeBSD доступна на сайте автора — http://www.fish.com/satan.

Распакуйте версию для Linux и перейдите в каталог `satan-1.1.1`. Затем введите команду:

```
perl reconfig
```

На что в ответ вы получите сообщение, что у вас не установлен Perl версии 5, хотя на вашей машине, скорее всего, уже будет установлена более поздняя версия. Первые изменения нужно сделать в файле `reconfig`. Не

вдаваясь в тонкости программирования на Perl, просто закомментируйте строки 51...70: начиная со строки:

```
for $dir (@all_dirs) {
```

и по строку:

```
#die "\nCan't find perl5! Bailing out...\n" unless $PERL;
```

Эти строки обеспечивают поиск интерпретатора Perl. Если Perl у вас установлен в каталог, отличный от `/usr/bin`, **то перед строкой:**

```
print "\nPerl5 is in $PERL\n";
```

добавьте строку:

```
$PERL=/path-to-perl/
```

Таким образом переменной `$PERL` будет присвоено имя каталога Perl.

Затем перейдите в каталог src/boot и откройте файл `boot.c`. В нем нужно найти и закомментировать следующую строку:

```
char *strchr();
```

Она находится в самом начале файла (строка 24).

Потом замените файл `/src/fping/fping.c` одноименным файлом из дистрибутива SATAN для FreeBSD. Следующий шаг — найдите и закомментируйте следующие строки (у меня это строки 189...191):

```
#ifndef SYS_ERRLIST_DECLARED
extern char *sys_errlist[];
#endif
```

Кажется, все. Если вдруг компилятор выдаст вам примерно такое сообщение:

```
structure has no member named 'th_sport' (или 'th_dport').
```

то член структуры **th_sport** замените на **source, а th_dport — на dest.** Струкрура **tcphdr** (TCP Header) — это описание заголовка TCP. В BSD член структуры, обозначающий порт-источник называется **th_sport**, а в Linux этот элемент структуры называется **source** (аналогично, элемент **th_dport** называется dest). Структура **tcphdr** описана в файле `/usr/include/netinet/tcp.h`.

Теперь можно ввести команду:

```
make linux
```

Параметр **linux** — это цель для сборки программы SATAN для операционной системы Linux.

Если сборка программы прошла без ошибок, приступите к изменению путей программ, необходимых для SATAN. Пути прописаны в файле `config/paths.pl`.

Обычно этот файл должен выглядеть так, как это показано в листинге 23.1. Если расположение каких-либо программ у вас отличается от приведенного в файле (или вы хотите использовать другие программы), измените соответствующие пути.

Листинг 23.1. Файл paths.pl

```
$FINGER="/usr/bin/finger";
$FTP="/usr/bin/ftp";
$RPCINFO="/usr/sbin/rpcinfo";
```

```
$RUSERS="/usr/bin/rusers";
$SHOWMOUNT="/usr/sbin/showmount";
$YPWHICH="/usr/bin/ypwhich";
$NSLOOKUP="/usr/bin/nslookup";
$XHOST="/usr/bin/X11/xhost";
$PING="/bin/ping";
$MOSAIC="/usr/bin/netscape";
$TCP_SCAN="bin/tcp_scan";
$UDP_SCAN="bin/udp_scan";
$FPING="bin/fping";
$NFS_CHK="bin/nfs-chk";
$YP_CHK="bin/yp-chk";
$SAFE_FINGER="bin/safe_finger";
$MD5="bin/md5";
$SYS_SOCKET="bin/sys_socket";
$BOOT="bin/boot";
$GET_TARGETS="bin/get_targets";
$TIMEOUT="bin/timeout";
$SATAN_CF="config/satan.cf"; $SERVICES="config/services";
```

Для работы SATAN вам необходим браузер **Netscape.** Подойдут также **Mosaic и Lynx,** но лучше использовать **Netscape.** Для запуска SATAN введите

```
./satan
```

Никаких параметров при этом указывать не нужно. При запуске SATAN становится HTTP-сервер и запускает браузер **Netscape,** перенаправляя его на себя. Порт сервера SATAN выбирается случайным образом. Поскольку SATAN при запуске становится Web-сервером, можно попытаться «зайти» на этот сервер, подобрав номер порта. Однако при этом у вас ничего не выйдет — нужно еще знать идентификатор сессии: SATAN разрешает работать только с определенным идентификатором, который создается при каждом запуске SATAN.

Сканировать порты имеет право только пользователь root. В открывшемся окне **Netscape** вы увидите **Панель управления** SATAN (см. рис. 23.1). Выберите пункт **Data Management** и создайте новую базу данных, с которой вы будете работать. Введите имя базы данных (имя по умолчанию — **satan-data**) и нажмите на кнопку «Open/Create».

После создания базы данных результатов можно приступить к выбору цели для сканирования. Перейдите на страницу **Target Selection.** Вы увидите окно, в котором нужно ввести узел для сканирования и уровень сканирования (см. рис. 23.2).

Сканировать можно как отдельный узел, так и всю подсеть, в которой находится узел. Уровень сканирования может быть легким **(light),** средним **(normal)** и высоким **(heavy).** Установив необходимые параметры, нажмите на кнопку «Start the scan».

Сканирование производится примерно так: сканер SATAN получает всю возможную информацию о заданном узле, потом, если задан режим сканирования подсети, начнает сканировать всю сеть. Весь это процесс отобража-

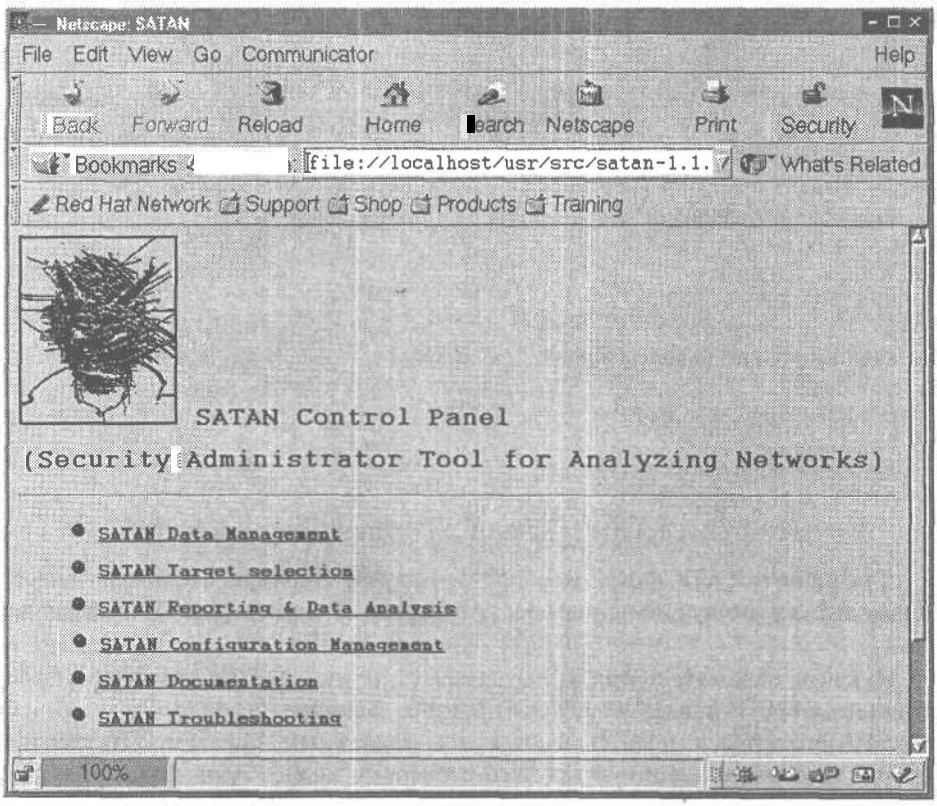


Рис. 23.1. *Панель управления SATAN*

ется в окне браузера Netscape. Время сканирования зависит от режима сканирования и размера сети. Например, отдельный узел локальной сети в режиме light сканируется несколько секунд.

После сканирования я рекомендую вам выйти из SATAN, зарегистрироваться в системе как обыкновенный пользователь (сейчас вы были зарегистрированы как пользователь root), заново запустить SATAN и спокойно просматривать результаты сканирования (Reporting and Data Analysis). Долго работать с SATAN под пользователем root не рекомендуется, как и переходить на другой сайт, не завершив работу SATAN. При переходе на другой узел Netscape передаст ему переменную HTTP_REFERER, в которой будут указаны идентификатор сессии и номер порта сервера SATAN.

Для редактирования конфигурационного файла SATAN можно воспользоваться утилитой **Configuration Management,** хотя можно редактировать файл satan.cf и вручную.

Среди прочих параметров есть одновременно интересный и опасный параметр: **max_proximity_level.**

Этот параметр задает уровень сканирования соседей. По умолчанию этот параметр равен нулю и соседние сети не сканируются. Чтобы лучше понять,

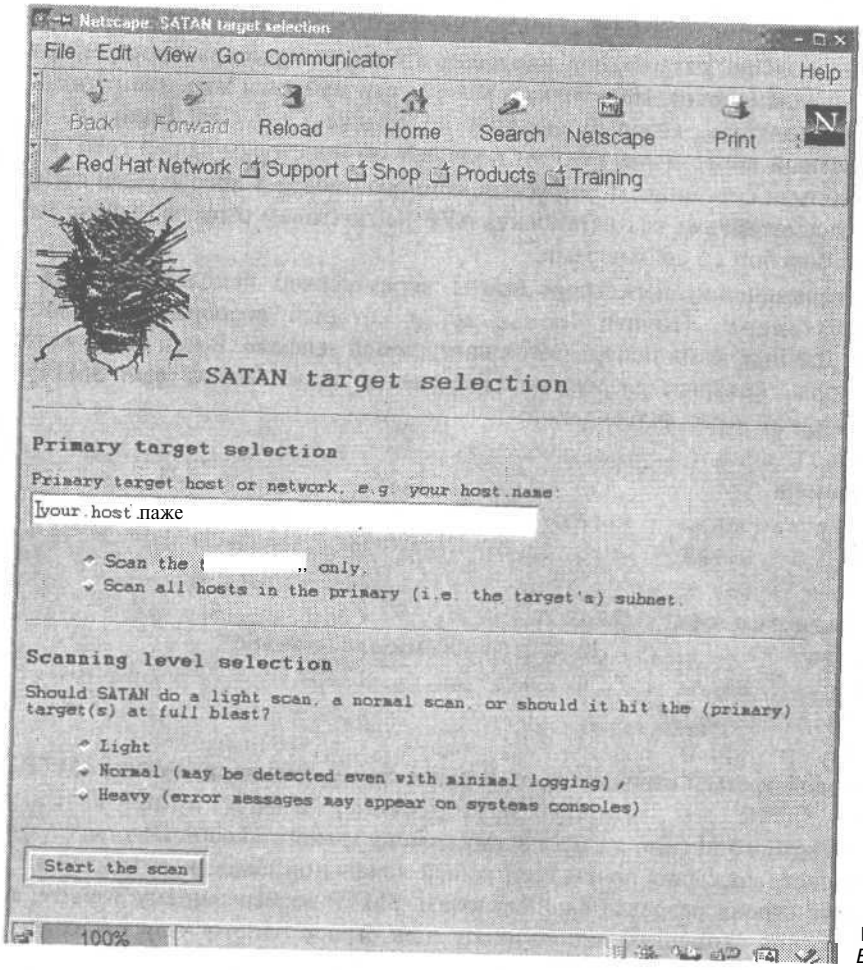


Рис. 23.2. *Выбор цели*

что это такое. Представьте себе такую ситуацию. SATAN сканирует узел host.dep1.company.ru. Сканер определяет, что домен dep1.company.ru обслуживает сервер DNS ns.dep1.company.ru. Пусть одна из записей NS этого сервера указывает на узел ns.company.ru. Автоматически домен company га попадает в число сканируемых. Поэтому никогда не устанавливайте значение параметра **max_proximity_level** большее *2,* иначе это может закончиться сканированием всего Интернета.

23.2. *Защита от слама*

Вам, наверное, надоело получать письма спаммеров примерно такого содержания:

```
Вы можете заработать в течение следующих 180 дней на рассылке e-mail
КАЖЕТСЯ НЕВОЗМОЖНЫМ?? Прочитайте детали, в этом нет никакой каверзы
или обмана.
```

И при этом они (спаммеры) нагло заявляют: «ЭТО НЕ СПАМ !!!»

А еще больше раздражают вирусописатели, которые распространяют сетевые вирусы (черви). Наверняка в вашей сети найдется хоть один доверчивый пользователь, который щелкнет на ссылке Cool Girl! Enjoy It!. И тогда злостный вирус-червь (Worm) поползет по просторам вашей сети. Не очень приятная ситуация... С вирусами немного проще: в предыдущей главе мы уже рассматривали как установить AVP на почтовом сервере, сейчас же займемся борьбой со спаммерами.

Для ограничения пересылки почты через сервер используется файл /etc/mail/access (точнее access.db, в который преобразуется файл access). Данный файл используется программой **sendmail**. **В** нем вы можете указать узлы, которым разрешено (запрещено) использовать ваш SMTP-сервер. Формат этого файла такой:

```
узел I сеть | пользователь действие
```

Например:

```
localhost.localdomain    RELAY
localhost          RELAY
127.0.0.1          OK
spammer@spamworld.com    REJECT
spamworld.com              ERROR:"550 Access denied"
192.168.1          RELAY
host.mydomain.ruREJECT
mydomain.ru        RELAY
```

В первой-третьей строках мы разрешаем самим себе использовать SMTP-сервер. Затем мы запрещаем пересылку почты пользователю spammer@spamworld.com, а также всему домену spamworld.com. Шестая строка разрешает пересылку почты всей нашей локальной подсети — 192.168.1.*. Последняя строка разрешает использовать SMTP-сервер нашему домену, а предпоследняя запрещает использовать наш сервер одному узлу из домена mydomain.ru — host.

Разница между действием OK и RELAY заключается в том, что в первом случае (OK) пересылка разрешается, даже если другие правила **sendmail** запретили пересылку почты, например, имя узла не разрешено (при использовании DNS).

Для запрещения пересылки можно просто использовать REJECT. Тогда пользователь увидит сообщение «Access denied». Действие ERROR более информативно, так как вы можете указать любое свое сообщение (установить реакцию на ошибку). Действие ERROR можно записать по-другому: ERROR:D.S.N:Message, где D.S.N — это код ошибки в соответствии с RFC 1893.

Для того чтобы новые правила доступа вступили в силу, введите команду:

```
makemap hash /etc/mail/access < /etc/mail/access
```

Изменения вступят в силу сразу после завершения работы программы **makemap.** Перезагружать **sendmail** при этом не нужно!

23.3. Ограничение системных ресурсов

Иногда пользователи жалуются, что администраторы сильно «урезают» их права. Однако делается это с благой целью -- обеспечить здоровье системы. Например, представьте, что пользователь-вредитель Пупкин запустит такой сценарий:

```
#!/bin/bash
# Сценарий loop
echo "Бесконечный цикл"
./loop
```

Этот сценарий выводит строку «Бесконечный цикл», а потом запускает самого себя. Получается что-то наподобие прямой рекурсии. Рано или поздно такой сценарий использует все системные ресурсы, и при запуске полезного процесса вы получите сообщение:

```
Unable to fork
```

То есть невозможно создать процесс. Этого можно легко избежать, если вы определите параметр **nproc** в файле `/etc/security/limits.conf`. В файле `limits.conf` задаются ограничения ресурсов системы для пользователя или группы пользователей. Формат файла таков:

```
<domain> <type> <item> <value>
```

Первое поле **(domain)** может содержать:

1. Имя пользователя.
2. Имя группы. Перед именем группы нужно указать символ «@».
3. Символ «*». Данное ограничение будет ограничением по умолчанию.

Второе поле (type) — это тип ограничения: мягкое (soft) или жесткое (hard). Мягкое офаничение определяет число системных ресурсов, которое пользователь все еще может превысить, жесткое офаничение превысить невозможно. При попытке сделать это, пользователь получит сообщение об ошибке.

Что касается элемента ограничения **(item),** то им может быть:

core.....ограничение размера файла core (Кб).
data.....максимальный размер данных (Кб).
fsize.....максимальный размер файла (Кб).
memlock.....максимальное заблокированное адресное пространство (Кб).
nofile.....максимальное число открытых файлов.
stack.....максимальный размер стека (Кб).
cpu.....максимальное время процессора (минуты).
nproc.....максимальное число процессов.
as.....ограничение адресного пространства.
maxlogins.....максимальное число одновременных регистрации в системе.
locks.....максимальное число файлов блокировки.

Рассмотрим несколько примеров. Например, нам нужно установить максимальное число процессов для пользователя user. Это можно сделать с помощью таких записей:

```
user soft nproc 50
user hard nproc 60
```

Первая строка определяет мягкое ограничение (равное 50), а вторая — жесткое.

Допустим, у вас есть группы dialup1 и dialup2. В каждую группу входят 30 пользователей. При этом у вас есть всего 30 входящих линий, поэтому нужно обеспечить одновременную работу не более 15 пользователей из каждой группы. Это делается так:

```
@dialup1 – maxlogins 14
@dialup2 -- maxlogins 14
```

В первом и втором случае из каждой группы пользователей одновременно работать смогут не более 15 (maxlogins 14 — отсчет начинается с нуля). При регистрации шестнадцатый пользователь увидит сообщение:

```
Too many logins for 'dialup1'
```

Иногда бывает полезным ограничить самого себя, то есть пользователя root. Рассмотрим, как это сделать. В файле /etc/securetty, который уже упоминался выше, указываются терминалы и виртуальные консоли, из которых может регистрироваться пользователь root. Я рекомендую вообще запретить регистрацию пользователя root из консоли. Для этого удалите (или закомментируйте) все строки в файле /etc/securetty. Если вам будут нужны максимальные привилегии, используйте команду **su** (super user). После ввода этой команды, программа запросит у вас пароль пользователя root, и если пароль правилен, вы получите привилегии пользователя root.

24

Вместо заключения

Напоследок — несколько рекомендаций по администрированию сервера. После того, как вы установили и настроили операционную систему, нужно регулярно «присматривать» за сервером, не полагаясь на надежность и защищенность Linux. Конечно, если у вас обыкновенный роутер, после настройки системы вы с чистой совестью можете поставить ваш сервер на полку и забыть о нем до тех пор, пока не понадобится внести новые правила фильтрации.

Если же ваш сервер выполняет более серьезные задачи, например, является сервером аутентификации, Web или FTP-сервером компании, нужно хотя бы один раз в день просматривать системные протоколы с целью обнаружить несанкционированный доступ к системе.

Что же делать, если вы обнаружили следы атаки на ваш сервер? Нужно деактивировать тот сервис, через который злоумышленник проник в вашу систему. Можно даже вообще отключить сервер из сети (компьютерной, а не сети питания!), если вы можете себе такое позволить. Затем нужно попытаться «заштопать дыру»: если взлом произошел из-за неправильной (или неполной) настройки сервера, подправьте конфигурационный файл сервиса, который был взломан. Если же взлом произошел из-за недоработок сервиса со стороны разработчиков, посетите сайт разработчиков и скачайте новую версию сервиса или патч к ней, исправляющий данную ошибку. Перед этим убедитесь, что эта ошибка исправлена в новой версии или в патче, в противном случае сообщите об ошибке разработчикам. Желательно проверить другие сервисы системы и установить в случае необходимости к ним «заплатки».

Не дожидайтесь грома с неба, а посещайте конференции, форумы, подпишитесь на специальные рассылки по безопасности. Следует также подписаться на рассылки разработчиков тех сервисов, которые используются в вашей системе. Вы должны узнать о потенциальных дырах раньше, чем злоумышленник и успеть обновить систему до попытки взлома.

Как выследить злоумышленника? Для этого нужно провести небольшое расследование. Соберите как можно больше информации о нем, одного IP-адреса будет явно маловато. Если это пользователь другой сети, обратитесь к администратору сети злоумышленника: вместе вы добьетесь большего.

Если злоумышленником является пользователь вашей сети, проверьте, тот ли это IP-адрес, который он постоянно использует при работе в сети. Бывают случаи, когда пользователи используют чужие IP-адреса для такого рода операций, в то время как истинные владельцы этих адресов вообще выключили компьютер и ничего не подозревают о взломе. Используя неэлектронные методы коммуникации, например, поговорив с этим пользователем лично или

позвонив к нему по телефону, выясните, работал ли вообще он за компьютером во время взлома и был ли компьютер в это время включен, при этом ничего не говорите ему о взломе. В любом случае, прежде чем обвинить кого-либо во взломе, вы должны быть уверены на все 100%.

Нужно отметить еще один немаловажный факт: если вы думаете, что безопасность вашей сети заключается только в безопасности вашего сервера, вы ошибаетесь. В случае, если ваша сеть не является подсетью Интернет (всем компьютерам сети назначены настоящие IP-адреса), ваша задача упрощается: вам всего лишь нужно установить антивирусы на компьютерах пользователей и регулярно обновлять антивирусные базы, желательно также установить антивирус на почтовом сервере. Если рабочие станции вашей локальной сети используют операционную систему Windows (а в большинстве случаев это так), отключите на шлюзе порты 135...139 и порт 129. Более подробно об этом написано в гл. 17. Также нужно обращать внимание на заплатки к Internet Explorer (только не те, которые будут периодически рассылаться по электронной почте пользователям вашей сети любителями троянских коней). Загружать обновления для IE нужно только с сайта Microsoft. А если вы убедите ваших пользователей использовать Netscape Communicator вместо IE, TheBat! вместо Outlook, вы таким образом избавите себя от периодического посещения сайта Microsoft.

Предположим, что вы являетесь администратором Web или FTP-сервера. В этом случае как можно чаще делайте резервные копии данных пользователей. Если систему в случае сбоя можно будет восстановить в течение 40-60 минут, то с данными пользователей будет сложнее. При этом нужно правильно организовать процесс резервного копирования, чтобы не запутаться в созданных копиях. Чтобы избежать этого, правильно выберите стратегию резервного копирования. Одна из таких стратегий обсуждалась в гл. 4. По возможности используйте массивы RAID — они обеспечивают более высокий уровень надежности. Однако использование RAID не освобождает вас от резервного копирования.

Не спешите обновлять программное обеспечение сервера. Если у вас все прекрасно работает, совсем не обязательно устанавливать самые новые версии программного обеспечения. Обновлять программное обеспечение сервера нужно лишь при возникновении необходимости. Например, в новом пакете исправлены некоторые ошибки предыдущих версий (в данном случае обновление называется патчем) или в новой версии появились необходимые вам новые функции. Помните, что новые версии совсем не обязательно будут работать стабильнее старых, уже проверенных. Старайтесь не использовать нестабильные (со вторым нечетным номером версии) ядра системы. Такие ядра можно установить на домашней машине и, убедившись, что они надежно работают, установить на сервер.

Еще раз отмечу, что системному администратору нужно постоянно повышать свой уровень квалификации. Благо, сейчас проблемам безопасности посвящено много русскоязычных ресурсов. Если у вас что-то не получается настроить или что-то работает не так как вам бы хотелось, обратитесь к документации, документам HOWTO, различным конференциям. На компакт-диске собраны основные документы HOWTO на русском и английском языках.

Успехов!

Приложения

Приложение А. Конфигурационные файлы Linux

Практически все конфигурационные файлы ОС Linux находятся в каталоге /etc. У вас могут быть далеко не все файлы и каталоги, представленные в табл. АЛ. Наличие тех или иных файлов определяется дистрибутивом и наличием определенных пакетов.

Конфигурационные файлы *Таблица А. 1*

Имя	Тип	Описание
codepages	Каталог	Содержит различные кодировки
CORBA/servers	Каталог	Здесь расположены серверы CORBA. Обычно эти серверы используются оконной средой Gnome
crond*	Каталог	Содержит сценарии, которые будут выполняться демоном crond по определенному расписанию. Описание определяется в файле /etc/crontab
cups	Каталог	Конфигурационные файлы системы печати CUPS
default/useradd	Файл	Информация по умолчанию для программы useradd
DrakConf	Каталог	Содержит файлы настройки конфигуратора DrakConf
emacs	Каталог	Конфигурационные файлы редактора emacs (настройки по умолчанию)
gnome	Каталог	Конфигурационные файлы приложения оконной среды Gnome
gtk	Каталог	Настройки программ, использующих библиотеку Gtk. Все файлы имеют суффикс, который соответствует коду языка. В зависимости от выбранного языка будет использован один из файлов. В этих файлах, как правило, можно определить кодировку и шрифт для приложения
htdig	Каталог	Конфигурационные файлы системы индексирования (поисковой системы) ht:/Dig
httpd	Каталог	Конфигурационные файлы сервера Apache
mail	Каталог	Конфигурационные файлы программы sendmail
openldap	Каталог	Конфигурационные файлы LDAP
pam.d	Каталог	Параметры модуля аутентификации PAM
postfix	Каталог	Конфигурационные файлы программы postfix
PPP	Каталог	Конфигурационные файлы демона pppd
profile.d	Каталог	Различные сценарии инициализации
rc.d	Каталог	Сценарии инициализации системы. Данные сценарии выполняются при запуске системы. Уровень выполнения сценариев указывается в файле /etc/inittab
skel	Каталог	Настройки пользователя по умолчанию. При создании нового каталога пользователя содержимое данного каталога будет скопировано в новый каталог
ssh	Каталог	Конфигурационные файлы программы ssh
urpmi	Каталог	Конфигурационные файлы программы urpmi. Эта программа позволяет устанавливать пакеты RPM обыкновенным пользователям. Устанавливаться могут только авторизированные пакеты

Продолжение табл. А. 1

Имя	Тип	Описание
X11	Каталог	Конфигурационные файлы X Window
xinet.d	Каталог	Дополнительные конфигурационные файлы суперсервера xinetd
aliaces	Файл	Файл псевдонимов системы электронной почты
anacrontab	Файл	Расписание планировщика Anacron
at.deny	Файл	Расписание планировщика at
auto.*	Файл	Файлы конфигурации программы automount (сервис autofs)
bashrc	Файл	Установки интерпретатора bash
conf.linuxconf	Файл	Файл конфигурации конфигуратора linuxconf. В нем определяются модули программы linuxconf
crontab	Файл	Расписание планировщика crond
csh.cshrc	Файл	Конфигурационный файл интерпретатора C Shell
csh.login	Файл	Команды, которые будут выполнены при регистрации в системе пользователя, использующего C Shell
exports	Файл	Экспортируемые файловые системы. Используется сервисом NFS
fax.config	Файл	Установки системы приема и передачи факсов
fb.modes	Файл	Примеры видеорежимов
fdprm	Файл	Параметры дисковода на гибких дисках
filesystem	Файл	Поддерживаемые файловые системы
fstab	Файл	Содержит статическую информацию о файловых системах. Указанные в этом файле файловые системы будут смонтированы при загрузке системы
ftpusers	Файл	Содержит список пользователей, которым запрещена регистрация на сервере FTP
gettydefs	Файл	Настройки терминалов по умолчанию (используются программой getty)
gpm-root.conf	Файл	Файл конфигурации сервера gpm. Сервер gpm обеспечивает поддержку мыши на виртуальных консолях
group	Файл	Информация о группах
gshadow	Файл	Информация о группах при использовании Shadow Passwords
host.conf	Файл	Содержит конфигурационную информацию библиотеки разрешения имени узла сети. В этом файле определяется порядок поиска имен узлов сети
hosts	Файл	Содержит статическую информацию о соответствии имени определенному IP-адресу. Этот файл участвует в процессе разрешения имен
hosts.allow	Файл	Содержит список узлов, которым разрешен доступ к этой машине
hosts.deny	Файл	Содержит список узлов, которым запрещен доступ к этой машине
identd.conf	Файл	Параметры идентификации
inetd.conf	Файл	Файл конфигурации суперсервера inetd
inittab	Файл	Таблица инициализации системы. Используется программой init
isapnp.gone	Файл	Некоторые параметры PnP для устройств ISA
issue, issue.net	Файл	Определяют приветствие при попытке регистрации в системе. Файл issue выводится на консоль при локальной регистрации, а issue.net — при регистрации по сети, например, по протоколу telnet
lilo.conf	Файл	Конфигурационный файл LILO
lmhosts	Файл	Часть пакета Samba. Его назначение аналогично назначению одноименного файла в Windows NT. Другими словами, этот файл предназначен для преобразования имен NetBIOS в IP-адреса
login.defs	Файл	Некоторые дополнительные параметры для программ useradd и groupadd
lynx.cfg	Файл	Настройки по умолчанию браузера lynx
mime-magic	Файл	«Магический» файл MIME-типов
mime-types	Файл	Здесь задаются MIME-типы. Этот файл может использоваться сервером Apache вместо файла apache-mime.types, но его нужно прописать в файле httpd.conf
modules.conf	Файл	Содержит список загружаемых модулей и определяет их параметры
motd	Файл	Сообщение дня (Message of The Day)
mtab	Файл	Содержит информацию о смонтированных в данный момент файловых системах

Продолжение табл. А. 1

Имя	Тип	Описание
networks	Файл	Содержит информацию о других сетях
passwd	Файл	Информация об учетных записях пользователей
printcap	Файл	Информация об установленных в системе принтерах
proftpd.conf	Файл	Конфигурационный файл сервера ProFTPD
protocols	Файл	Список поддерживаемых протоколов, согласно стандарта RFC 1340
pwdb	Файл	Установки библиотеки pwdb
quota.conf	Файл	Информация о квотах
sendmail.cf	Файл	Основной файл конфигурации программы sendmail
services	Файл	Содержит разрешенные Интернет-сервисы. Этот файл отвечает требованиям стандарта RFC 1700
passwd	Файл	Информация об учетных записях пользователей при включенных теневых паролях (Shadow Passwords)
shells	Файл	Содержит список зарегистрированных в системе интерпретаторов командной строки
shutmsg	Файл	Содержащееся в этом файле сообщение обычно выводится клиентам сервера wu-ftpd при завершении работы сервера
smb.conf	Файл	Основной файл конфигурации пакета Samba
termcap	Файл	Настройки терминалов
xinetd.conf	Файл	Файл конфигурации суперсервера xinetd

Каталог /etc/xinet.d

Чтобы уменьшить объем основного файла конфигурации `xinetd.conf`, суперсервер **xinetd** использует каталог `/etc/xinetd.d`. В нем расположены файлы описания сервисов, которые не включены в состав основного файла конфигурации. Желательно, чтобы имя файла совпадало с именем сервиса, но это не является обязательным условием. Суперсервер просто использует содержимое этих файлов параллельно с файлом конфигурации. В этих файлах содержатся директивы, подобно основному файлу `xinetd.conf`. Если в файле `xinetd.conf` содержатся описания практически всех сервисов, то в этих содержится, как правило, установки для одного определенного сервиса. Однако никто не мешает вам описать несколько сервисов в одном файле.

При использовании этого каталога файл `xinetd.conf` будет выглядеть как в листинге А.1.

Листинг АЛ. Файл xinetd.conf

```
#
t Пример простейшей конфигурации сервера xinetd
#
defaults
{
    instances       = 60
    log_type    = SYSLOG authpriv
    log_on_success      = HOST PID
    log_on_failure      = HOST RECORD
}
includedir /etc/xinetd.d
```

В секции defaults определяются настройки по умолчанию для всех сервисов. Затем директива includedir включает содержимое всех файлов из указанного каталога в файл конфигурации. Файлы описания сервисов выглядят примерно так (листинг А.2):

Листинг А.2

```
# по умолчанию: отключен
# описание: сервер echo [tcp]
service echo
{
    type        = INTERNAL
    id          = echo-stream
    socket_type = stream
    protocol=  tcp
    user        = root
    wait        = no
    disable       = yes
}
```

Файл /etc/urpmi/urpmi.cfg

Профамма urpmi позволяет обыкновенным пользователям устанавливать RPM-пакеты. Обычно устанавливать новые пакеты имеет право только пользователь root. В файле `urpmi.cfg` содержится список носителей, из которых пользователь может установить пакет.

Файл /etc/aliases

Определяет псевдонимы пользователей при работе с системой электронной почты (см. листинг А.3). Формат данного файла следующий:

```
псевдоним: реальный_пользователь
```

После изменения этого файла нужно перестроить базу псевдонимов. Для этого введите команду **newaliases**.

Листинг А.3. Файл /etc/aliases

```
# Основные псевдонимы — они должны присутствовать
MAILER-DAEMON:  postmaster
postmaster: root
# Общая переадресация почты
bin:     root
daemon:      root
games:       root
ingres:      root
nobody:      root
system:      root
toor:        root
uucp:        root
t Хорошо известные псевдонимы
manager:root
dumper:      root
```

```
operator:    root
decode:      root
# Этот пользователь может читать почту пользователя root
#root:       marc
```

Файл /etc/host.conf

Этот файл является системой разрешения имени узла сети. В нем задается порядок поиска имени компьютера сети. Порядок поиска определяется директивой **order** (см. листинг А.4). Можно использовать варианты: **hosts, bind, nis,** а также всевозможные их комбинации. Параметр **hosts** директивы **order** означает, что поиск имени компьютера будет производиться в локальном файле /etc/hosts. Параметр **bind** используется для запроса сервера DNS. Параметр **nis** опрашивает сетевую службу информации (Network Information Server), если таковая используется в вашей сети.

Разрешение в IP-адрес имени будет выполнено именно в таком порядке, как указано в директиве order. Директива **multi** означает, что сервер может поддерживать несколько IP-адресов. Рекомендуется установить значение **on.**

Листинг А.4. Файл /etc/host.conf

```
order hosts,bind
multi on
```

Файл /etc/hosts

Файл /etc/hosts содержит статическую информацию для преобразования имени компьютера в IP-адрес. Файл имеет формат:

IP-адрес полное_имя псевдоним

Пример файла /etc/hosts приведен в листинге А.5.

Листинг А. 5. Файл /etc/hosts

```
127.0.0.1    localhost.localdomain   localhost
127.0.0.1    dhsilabs.com            dhsilabs
192.168.1.2  pppO.com                ppp0
```

Файл /etc/networks

В этом файле содержится информация о других сетях. Формат этого файла такой:

домен адрес_сети псевдоним # комментарий

Некоторые программы, например, **netstat** позволяют указывать имя сети вместо ее адреса, поэтому этот файл существует в основном для удобства администратора, так как иногда имя запомнить проще, чем адрес. Пример файла networks приведен в листинге А.6.

Листинг А.6. Файл /etc/networks

```
.net99 192.168.99.0 net99 # Net 99
.net98 192.168.99.0 net98 # Net 98
.net97 192.168.99.0 net97 # Net 97
```

Файл /etc/motd

В данном файле содержится «сообщение дня» (Message of The Day). Это сообщение увидят пользователи после регистрации в системе. Данное сообщение должно быть написано латинскими буквами. Все мои старания русифицировать эту службу ни к чему не привели.

Файл /etc/resolv.conf

В этом файле указывается информация о серверах DNS. Директивы **nameserver** определяют, какими серверами, и в каком порядке необходимо пользоваться для разрешения имени узла. Директива **search** обеспечивает поиск домена, если он не указан. Обычно первым в этой директиве задается свой домен (см. листинг А.7).

Листинг А. 7. Файл /etc/resolv.conf

```
-search .dhsilabs.com .com .ru
 nameserver 127.0.0.1
 nameserver 192.168.1.1
 nameserver 192.168.99.1
```

Вместо директивы **search** можно использовать ее устаревшую версию – **domain,** которая не производит поиск домена, а просто указывает один домен, как правило, собственный.

Файл /etc/protocols

Листинг А.8. Файл /etc/protocols

```
# /etc/protocols: файл определения протоколов
#
# Протоколы Интернет (IP)
#
# Этот файл основан на стандарте RFC 1340
ip          0  IP          # протокол Интернет (IP)
icmp        1  ICMP        t протол межсетевых управляющих сообщений
igmp        2  IGMP        # управление группами
ggp         3  GGP         # протокол шлюз-шлюз
ipencap     4  IP-ENCAP    # IP, инкапсулированный в IP
st          5  ST          # режим дейтаграмм ST
tcp         6  TCP         # протокол управления передачей
egp         8  EGP         # протокол внешнего шлюза
pup         12 PUP         # PARC - универсальный протокол пакетов
udp         17 UDP         # протокол пользовательских дейтаграмм
hmp         20 HMP         # протокол мониторинга узла
xns-idp     22 XNS-IDP     # Xerox NS IDP
rdp         27 RDP         tt"reliable datagram" protocol
iso-tp4     29 ISO-TP4     # протокол передачи ISO класс 4
xtp         36 XTP         # протокол передачи Xpress
ddp         37 DDP         # протокол отправки дейтаграмм
idpr-cmtp 39 IDPR-CMTP # IDPR Control Message Transport
ipv6       41 IPv6       t Протокол IPv6
```

```
ipv6-route 43 IPv6-Route# Заголовок маршрута для IPv6
ipv6-frag  44 IPv6-Frag # Заголовок фрагмента для IPv6
ipv6-crypt 50 IPv6-Crypt# Закодированный заголовок для IPv6
ipv6-auth  51 IPv6-Auth # Заголовок аутентификации для IPv6
ipv6-icmp  58 IPv6-ICMP # ICMP для IPv6
ipv6-nonxt 59 IPv6-NoNxt# Заголовок "No Next" для IPv6
ipv6-opts  60 IPv6-Opts f Параметры назначения для IPv6
rspf       73 RSPF      # Radio Shortest Path First.
vmtp       81 VMTP      # Versatile Message Transport
ospf       89 OSPFIGP   # Open Shortest Path First IGP
ipip       94 IPIP      # Другая инкапсуляция IP
encap      98 ENCAP      t Другая инкапсуляция IP
```

В файле описаны доступные протоколы DARPA Интернет. Номера протоколов и их имена определяются информационным центром DDN.

Файл /etc/services

Листинг А.9. Файл /etc/services

```
# /etc/services:
#
# Сетевые службы
#
# Соответствует стандарту RFC 1700
tcpmux          1/tcp                   # Мультиплексор порта TCP
echo        7/tcp
echo        7/udp
discard     9/tcp       sink null
discard     9/udp       sink null
systat      11/tcp      users
daytime     13/tcp
daytime     13/udp
netstat     15/tcp
qotd        17/tcp      quote
msp         18/tcp                  # протокол отправки сообщений
msp         18/udp                  # протокол отправки сообщений
chargen     19/tcp      ttytst source
chargen     19/udp      ttytst source
ftp-data    20/tcp
ftp         21/tcp
fsp         21/udp      fspd
ssh         22/tcp                  # Протокол удаленного доступа SSH
ssh         22/udp                  # Протокол удаленного доступа SSH
telnet      23/tcp
# 24 -- зарезервирован
smtp        25/tcp      mail
# 26 - не присвоен
time        37/tcp      timserver
time        37/udp      timserver
rlp         39/udp      resource    # распределение ресурсов
nameserver  42/tcp      name        # IEN 116
```

```
whois 43/tcp    nicname
re-mail-ck 50/tcp                  # Протокол удаленной проверки почты
re-mail-ck 50/udp                  # Протокол удаленной проверки почты
domain     53/tcp     nameserver      # Служба имен
domain     53/udp     nameserver
mtp        57/tcp                  # запрещен
bootps     67/tcp                  # сервер BOOTP
bootps     67/udp
bootpc     68/tcp                  # клиент BOOTP
bootpc     68/udp
tftp       69/udp
gopher     70/tcp                  # Internet Gopher
gopher     70/udp
rje        77/tcp     netrjs
finger     79/tcp
www        80/tcp     http       # WorldWideWeb HTTP
www        80/udp                  # HyperText Transfer Protocol
link       87/tcp     ttylink
kerberos   88/tcp     kerberos5 krb5 # аутентификация  Kerberos v5
kerberos   88/udp     kerberos5 krb5 # аутентификация  Kerberos v5
supdup     95/tcp
# 100 - зарезервирован
hostnames 101/tcp     hostname        # usually from sri-nic
iso-tsap  102/tcp     tsap            # part of ISODE.
csnet-ns  105/tcp     cso-ns          # also used by CSO name server
csnet-ns  105/udp     cso-ns
#3com-tsmux    106/tcp         poppassd
#3com-tsmux    106/udp         poppassd
rtelnet    107/tcp                    # Удаленный Telnet
rtelnet    107/udp
pop2       109/tcp    pop-2      postoffice#  POP версия 2
pop2       109/udp    pop-2
pop3       110/tcp    pop-3           # POP версия 3
pop3       110/udp    pop-3
sunrpc     111/tcp    portmapper      # RPC 4.0 portmapper TCP
sunrpc     111/udp    portmapper      # RPC 4.0 portmapper UDP
auth       113/tcp    authentication tap ident
sftp       115/tcp
uucp-path  117/tcp
nntp       119/tcp    readnews untp  # USENET News Transfer Protocol
ntp        123/tcp
ntp        123/udp                    # сетевой протокол времени
netbios-ns 137/tcp                    # сервер имен NETBIOS
netbios-ns 137/udp
netbios-dgm    138/tcp                # служба дейтаграмм NETBIOS
netbios-dgm    138/udp
netbios-ssn    139/tcp                # служба сессий NETBIOS
netbios-ssn    139/udp
imap2      143/tcp    imap       # почтовый протокол Interim v2
imap2      143/udp    imap
```

```
snmp        161/udp                 # протокол Simple Net Mgmt Proto
snmp-trap   162/udp     snmptrap    # Traps for SNMP
cmip-man    163/tcp                 # ISO mgmt over IP (CMOT)
cmip-man    163/udp
cmip-agent  164/tcp
cmip-agent  164/udp
xdmcp       177/tcp                 # протокол X Display
xdmcp       177/udp
nextstep    178/tcp     NeXTStep Nextstep   # NeXTStep оконный
nextstep    178/udp     NeXTStep Nextstep   # сервер
bgp         179/tcp                 # протокол Border Gateway
bgp         179/udp
prospero    191/tcp
prospero    191/udp
irc         194/tcp                 # Internet Relay Chat
irc         194/udp
smux        199/tcp                 # мультиплексор SNMP Unix
smux        199/udp
at-rtmp     201/tcp                 # маршрутизация AppleTalk
at-rtmp     201/udp
at-nbp      202/tcp                 # разрешение имени AppleTalk
at-nbp      202/udp
at-echo     204/tcp                 # AppleTalk echo
at-echo     204/udp
at-zis      206/tcp                 # информация о зоне AppleTalk
at-zis      206/udp
qmtp        209/tcp                 # Протокол передачи почты qmail
qmtp        209/udp                 # The Quick Mail Transfer Protocol
z3950       210/tcp     wais        # NISO Z39.50 database
z3950       210/udp     wais
ipx         213/tcp                 # IPX
ipx         213/udp
imap3       220/tcp                 # Interactive Mail Access (IMAP)
imap3       220/udp                 # Интерактивная почта, вер 5 (IMAP)
rpc2portmap     369/tcp
rpc2portmap     369/udp             # Coda portmapper
codaauth2   370/tcp
codaauth2   370/udp                 # Coda-сервер аутентификации
ulistserv   372/tcp                 I UNIX Listserv
ulistserv   372/udp
ldap        389/tcp                 t Lightweight Directory Access Protocol
ldap        389/udp                 # Lightweight Directory Access Protocol
https       443/tcp                 # MCom
https       443/udp                 # MCom
snpp        444/tcp                 t Simple Network Paging Protocol
snpp        444/udp                 # Simple Network Paging Protocol
saft        487/tcp                 # Simple Asynchronous File Transfer
saft        487/udp                 I Simple Asynchronous File Transfer
npmp-local  610/tcp     dqs313_qmaster # npmp-local / DQS
npmp-local  610/udp     dqs313_qmaster I npmp-local / DQS
```

```
npmp-gui   611/tcp    dqs313_execd          t npmp-gui / DQS
npmp-gui   611/udp    dqs313_execd          # npmp-gui / DQS
hmmp-ind   612/tcp    dqs313_intercell      # HMMP Indication / DQS
hmmp-ind   612/udp    dqs313_intercell      # HMMP Indication / DQS
#
#  специфические  сервисы UNIX
t
exec       512/tcp
biff       512/udp    comsat
login      513/tcp
who        513/udp    whod
shell      514/tcp    cmd             # без пароля
syslog     514/udp
printer    515/tcp    spooler         # спулер принтера
talk       517/tcp
ntalk      518/udp
route      520/udp    router routed   # протокол RIP
timed      525/udp    timeserver
tempo      526/tcp    newdate
courier    530/tcp    rpc
conference531/tcp     chat
n,etnews    532/tcp   readnews
netwall    533/udp                    # -for emergency broadcasts
uucp       540/tcp    uucpd           # uucp daemon
afpovertcp548/tcp                     # AFP over TCP
afpovertcp548/udp                     # AFP over TCP
remotefs   556/tcp    rfs_server rfs  # Brunhoff remote filesystem
klogin     543/tcp                    # Kerberized 'rlogin' (v5)
kshell     544/tcp         krcmd      # Kerberized 'rsh' (v5)
kerberos-adm   749/tcp                # Kerberos 'kadmin' (v5)
#
webster    765/tcp                    # Network dictionary
webster    765/udp
*
#
ingreslock      1524/tcp
ingreslock      1524/udp
prospero-np    1525/tcp               # Prospero non-privileged
prospero-np    1525/udp
datametrics    1645/tcp  old-radius   # datametrics / old radius entry
datametrics    1645/udp  old-radius   # datametrics / old radius entry
sa-msg-port    1646/tcp  old-radacct  # sa-msg-port / old radacct entry
sa-msg-port    1646/udp  old-radacct  # sa-msg-port / old radacct entry
radius         1812/tcp               # Radius
radius         1812/udp               # Radius
radacct        1813/tcp               # Radius Accounting
radacct        1813/udp               # Radius Accounting
cvspserver     2401/tcp               t CVS client/server operations
cvspserver     2401/udp               # CVS client/server operations
venus          2430/tcp               # codacon port
venus          2430/udp               # Venus callback/wbc interface
```

```
venus-se   2431/tcp              # tcp side effects
venus-se   2431/udp              # udp sftp side effect
codasrv    2432/tcp              # not used
codasrv    2432/udp              # server port
codasrv-se 2433/tcp              # tcp side effects
codasrv-se 2433/udp              # udp sftp side effect
mysql      3306/tcp              # MySQL
mysql      3306/udp              # MySQL
rfe        5002/tcp              # Radio Free Ethernet
rfe        5002/udp              # Actually uses UDP only
cfengine   5308/tcp              # CFengine
cfengine   5308/udp              # CFengine
bbs        7000/tcp              t BBS service
#
# аутентификация Kerberos, версия 5
#
kerberos4       750/udp   kerberos-iv kdc# Kerberos (сервер) udp
kerberos4       750/tcp   kerberos-iv kdc# Kerberos (сервер) tcp
kerberos_master 751/udp                   # аутентификация Kerberos
kerberos_master 751/tcp                   # аутентификация Kerberos
passwd_server   752/udp                   # сервер паролей Kerberos
krb_prop        754/tcp                   # Kerberos slave propagation
krbupdate       760/tcp   kreg            # регистрация Kerberos
kpasswd         761/tcp   kpwd            # Kerberos "passwd"
kpop            1109/tcp                  # Pop, используя Kerberos
knetd           2053/tcp                  # демультиплексор Kerberos
zephyr-srv      2102/udp                  # сервер Zephyr
zephyr-clt      2103/udp                  # соединение Zephyr serv-hm
zephyr-hm       2104/udp
eklogin         2105/tcp  # Kerberos: закодированная регистрация
#
# Неофициальные сервисы (необходимы для NetBSD)
#
supfilesrv 871/tcp
supfiledbg 1127/tcp
#
# Протоколы отправки дейтаграмм
#
rtmp       1/ddp
nbp        2/ddp
echo       4/ddp
zip        6/ddp
# Сервисы Debian
poppassd   106/tcp                # Eudora
poppassd   106/udp                # Eudora
mailq      174/udp
mailq      174/tcp
ssmtp      465/tcp                # SMTP через SSL
gdomap     538/tcp
gdomap     538/udp
```

```
snews       563/tcp                    t NNTP через SSL
ssl-ldap    636/tcp                    t LDAP через SSL
omirr       808/tcp          omirrd
omirr       808/udp          omirrd
rsync       873/tcp                    # rsync
rsync       873/udp                    # rsync
swat        901/tcp                    # Add swat service used via
inetd
simap       993/tcp                    # IMAP через SSL
spop3       995/tcp                    # POP-3 через SSL
socks      1080/tcp                    # прокси-сервер socks
socks      1080/udp                    # прокси-сервер socks
rmtcfg     1236/tcp
xtel       1313/tcp
support    1529/tcp
cfinger    2003/tcp                 # GNU Finger
ninstall   2150/tcp                 # служба ninstall
ninstall   2150/udp                 # служба ninstall
afbackup   2988/tcp                 # Afbackup system
afbackup   2988/udp                 # Afbackup system
icp        3130/tcp                 # Internet Cache Protocol (Squid)
icp        3130/udp                 t Протокол кэширования (Squid)
postgres   5432/tcp                 # СУБД POSTGRES
postgres   5432/udp                 # СУБД POSTGRES
fax "      4557/tcp                 # FAX: передача факса (старый)
hylafax    4559/tcp                 # HylaFAX протокол клиент-сервер (новый)
noclog     5354/tcp                 # noclogd, используя TCP (nocol)
noclog     5354/udp                 # noclogd, используя UDP (nocol)
hostmon    5355/tcp                 # hostmon, используя TCP (nocol)
hostmon    5355/udp                 # hostmon, используя TCP (nocol)
ircd       6667/tcp                 # Internet Relay Chat
ircd       6667/udp                 t Internet Relay Chat
webcache   8080/tcp                 # WWW: служба кэширования
webcache   8080/udp                 # WWW: служба кэширования
tproxy     8081/tcp                 # Прозрачный прокси
tproxy     8081/udp                 # Прозрачный прокси
mandelspawn     9359/udp   mandelbrot
amanda     10080/udp        # служба резервного копирования amanda
kamanda    10081/tcp        # CPK amanda (Kerberos)
kamanda    10081/ud         # CPK amanda (Kerberos)
amandaidx  10082/tcp        # служба резервного копирования amanda
amidxtape  10083/tcp        # служба резервного копирования amanda
isdnlog    20011/tcp             # система протоколирования isdn
isdnlog    20011/udp             # система протоколирования isdn
vboxd      20012/tcp             I voice box system
vboxd      20012/udp             # voice box system
jserver    22273/tcp             # Модуль JServer
binkp      24554/tcp             # Binkley
binkp      24554/udp             # Binkley
asp        27374/tcp             # Протокол поиска адреса
a'sp       27374/udp             t Протокол поиска адреса
```

```
tfido       60177/tcp              # Ifmail (FIDO Mail)
tfido       60177/udp              # Ifmail
fido        60179/tcp              I Ifmail
fido        60179/udp              # Ifmail
# Локальные службы
linuxconf  98/tcp
smtp  465/tcp
imaps 993
pop3s 995
```

В файле `services` определены сетевые службы, установленные в системе. Если вы не используете или не хотите использовать какие-нибудь службы, просто отключите их здесь, закомментировав нужную вам строку.

Файл /etc/modules.conf

В этом файле содержится список модулей, которые будут встроены в ядро при загрузке системы (см. листинг АЛО).

Листинг А. 10. Файл /etc/modules.conf

```
alias net-pf-4 ipx
pre-install pcmcia_core /etc/rc.d/init.d/pcmcia start
alias parport_lowlevel parport_pc
pre-install plip modprobe parport_pc ; echo 7 > /proc/parport/
0/irq
alias sound-slot-0 via82cxxx_audio
```

При конфигурации модулей можно использовать директивы **if, else, elseif, endif.** Конструкция этих директив такова:

```
if  ВЫРАЖЕНИЕ
    любые директивы по установке модулей
elseif ВЫРАЖЕНИЕ
    любые директивы по установке модулей
endif
```

Директива **path** определяет местонахождение модулей:

```
path=путь
```

В директиве **path** можно использовать тэги **net, misc, scsi, video, ipv4** и другие. Каждый тэг определяет модули какого-нибудь типа. Если вам нужно нестандартное расположение каких-либо модулей, например, модулей сетевых устройств, используйте директиву **path** так:

```
path[net]=путь.
```

Общий вид директивы **path** следующий: **path**[тэг]=путь.

Для установки модулей можно использовать директивы **install, pre-install, post-install.** В этих директивах можно указать определенные команды. Команда, указанная в директиве **install,** будет выполнена вместо команды **insmod**. Команды, указанные в директивах **pre-install** или **post-install,** будут выполнены соответственно до и после установки модуля. Синтаксис этих директив таков:

```
директива модуль команда
```

Директива **remove** предназначена для извлечения модуля. Ее синтаксис:

remove модуль команда

Если указана команда, то она будет выполнена вместо команды **rmmod**. Директива **alias** определяет псевдонимы модулей. Например, **alias iso9660 isofs**. Описание других директив **(keep, add, define)** вы найдете в справочной системе.

Приложение Б. Общие параметры программ для системы X Window

Каждая программа, предназначенная для работы в системе X Window, имеет параметры, представленные в табл. Б.1.

Параметры программ X Window *Таблица Б. 1*

Параметр	Описание
-background <red | green | blue>	Устанавливает цвет фона
-background цвет	Устанавливает цвет фона окна
-bg цвет	Устанавливает цвет фона окна
-display система:номер_дисплея	Указывает нужный сервер X. По умолчанию используется номер 0
-fg цвет	Устанавливает основной цвет окна
-fn шрифт	Устанавливает шрифт окна. Шрифт можно выбрать с помощью программы xfontsel
-font шрифт	Устанавливает шрифт окна. Шрифт можно выбрать с помощью программы xfontsel
-foreground_color <red | green | blue>	Устанавливает основной цвет окна
-foreground цвет	Устанавливает основной цвет окна
-geometry ширина x высота + x +y	Устанавливает размер и расположение окна
-geometry ширина x высота	Устанавливает размер окна
-geometry + x +y	Устанавливает расположение окна в пикселях
-height n	Устанавливает вертикальный размер окна
-position x y	Устанавливает положение верхнего левого угла окна
-reverse	Меняет местами основной и цвет фона окна
-rv	Аналогичен параметру -reverse
-size ширина высота	Устанавливает размер окна
-Wb <red | green | blue>	Устанавливает цвет фона
-Wf цвет	Устанавливает основной цвет окна
-WG ширина x высота + x + y	Устанавливает размер и расположение окна
-WG ширина x высота	Устанавливает размер окна
-WG + x + y	Устанавливает расположение окна
-Wh строка	Установка вертикального размера окна по строкам
-Wi	Запустить программу, свернув ее в значок
-width столбцы	Установить ширину окна в столбцах
-Wp x y	Устанавливает положение верхнего левого угла окна
-Wr система:номер	Указывает нужный сервер X. По умолчанию используется номер 0
-Ws ширина высота	Установка размеров окна в пикселях
-Wt шрифт	Использовать указанный шрифт
-Ww столбцы	Установка ширины окна в столбцах

Приложение В. Компактное и нормальное ядра

В этом приложении вы найдете листинги файлов конфигурации компактного и нормального ядер. В гл. 18 я рекомендовал вам исключать из состава ядра не нужные вам функции. Например, если ваш сервер оборудован только IDE-дисками, то зачем ему поддержка SCSI? В большинстве случаев, при установке дистрибутива ядро настраивается таким образом, чтобы оно могло работать на большом числе компьютеров разных конфигураций. Для этого в состав ядра (или в загружаемые модули) включается поддержка всех чипсетов, файловых систем, сетевых плат и других устройств, что увеличивает размер ядра и делает его более громоздким. Уменьшив число ненужных вам функций, вы повысите производительность всей системы. Но будьте внимательны: отключение одной опции может повлиять на другие. Здесь точно такая же ситуация как и с пакетами RPM: есть зависимые друг от друга пакеты, а есть и несовместимые. Если вы отключите какую-нибудь опцию, которая нужна другим опциям ядра, во время компиляции вы получите сообщение в виде предупреждения (warning) и ядро при этом вообще может не собраться.

Существует и еще один вариант неудовлетворения зависимости опций, когда ядро нормально собирается, но при перезапуске компьютера невозможно запустить систему, используя это ядро. Во время выполнения программы **make menuconfig** (**config** или **xconfig**) вы не узнаете о подобной несовместимости. Все предупреждения и ошибки вы увидите во время компиляции ядра (make zImage). Запомните одно простое правило: ядро должно собираться. Если ядро не собирается, значит, вы допустили ошибку при работе с программой **make menuconfig.**

В листинге B.1 представлен пример компактного ядра. Я «заточил» его под свой домашний компьютер, поэтому не гарантирую, что оно у вас будет работать. В листинге B.1 вы можете увидеть основные принципы, по которым я собирал ядро. Во-первых, я отказался от загружаемых модулей и включил все необходимые мне драйверы непосредственно в ядро системы. Конечно, такой подход не уменьшит размер файла ядра, но тогда это ядро можно использовать при загрузке с дискеты. Во-вторых, я отключил поддержку всех чипсетов, кроме того, который установлен в моей системе. Я также отказался от поддержки PnP: я не собираюсь добавлять в систему новые устройства, поэтому держать лишний код в ядре для определения нового оборудования, которое никогда не будет установлено, глупо. Обо всех остальных опциях вы узнаете, просмотрев листинг B.1. Компактные ядра, подобные этому, можно использовать на сервере, выполняющем только определенные функции, например, маршрутизацию. Обычно такой сервер после установки и настройки запирают в какой-нибудь комнате и больше не подходят к нему. Естественно, при добавлении новых устройств, для которых необходим драйвер, которого нет в ядре, возникнут определенные неудобства при компилировании ядра. Но вы же не добавляете новые устройства каждый день?

В листинге В.2 представлен обычный файл конфигурации ядра. Такое ядро должно работать на большинстве компьютеров.

Как использовать эти листинги? Очень просто: запускаете **make menuconfig** и сравниваете названия опций с параметрами конфигурационного файла: что и где вы должны догадаться по смыслу, а потом сравните значения параметров. Тем более, что файл конфигурации разбит на части, названия которых аналогичны названию разделов программы menuconfig. Можно также пойти по более простому пути (хотя он не очень удобен): запустите make с параметром config. Программа будет задавать вам вопросы, на которые вы должны ответить Y, N или M. Y - - включить данную функцию в состав ядра, N — не включать в состав ядра, M — включить в виде модуля. Вопросы будут задаваться примерно так:

```
CONFIG_EXPERIMENTAL ?
```

На что вы должны ответить Y или N.

Листинг В.1. Пример файла конфигурации компактного ядра

```
#
# Automatically generated by make menuconfig: don't edit
#

#
# Code maturity level options
#
CONFIG_EXPERIMENTAL=N

#
# Processor type and features
#
# CONFIG_M386 is not set
# CONFIG_M486 is not set
# CONFIG_M586 is not set
# CONFIG_M586TSC is not set
CONFIG_M686=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X8 6_INVLPG=y
CONFIG_X86_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_TSC=y
CONFIG_X86_GOOD_APIC=y
CONFIG_1GB=y
# CONFIG_2GB is not set
# CONFIG_MATH_EMULATION is not set
CONFIG_MTRR=y
# CONFIG_SMP is not set

#
# Loadable module support
#
```

```
# CONFIG_MODULES is not set

#
# General setup
I
# CONFIG_BIGMEM is not set
CONFIG_NET=y
CONFIG_PCI=y
# CONFIG_PCI_GOBIOS is not set
# CONFIG_PCI_GODIRECT is not set
CONFIG_PCI_GOANY=y
CONFIG_PCI_BIOS=y
CONFIG_PCI_DIRECT=y
# CONFIG_PCI_QUIRKS is not set
CONFIG_PCI_OLD_PROC=y
# CONFIG_MCA is not set
# CONFIG_VISWS is not set
CONFIG_SYSVIPC=y
CONFIG_BSD_PROCESS_ACCT=y
CONFIG_SYSCTL=y
CONFIG_BINFMT_AOUT=y
CONFIG_BINFMT_ELF=y
# CONFIG_BINFMT_MISC is not set
# CONFIG_BINFMT_JAVA is not set
CONFIG_PARPORT=y
# CONFIG_PARPORT_PC is not set
f CONFIG_APM is not set
# CONFIG_TOSHIBA is not set

#
# Plug and Play support
#
# CONFIG_PNP is not set

#
# Block devices
f
CONFIG_BLK_DEV_FD=y
# CONFIG_BLK_DEV_IDE is not set
I CONFIG_BLK_DEV_HD_ONLY is not set
CONFIG_BLK_DEV_LOOP=y
CONFIG_BLK_DEV_NBD=y
CONFIG_BLK_DEV_MD=y
I CONFIG_AUTODETECT_RAID is not set
I CONFIG_MD_LINEAR is not set
# CONFIG_MD_STRIPED is not set
# CONFIG_MD_MIRRORING is not set
I CONFIG_MD_RAID5 is not set
# CONFIG_MD_TRANSLUCENT is not set
# CONFIG_MD_HSM is not set
```

```
CONFIG_BLK_DEV_RAM=y
CONFIG_BLK_DEV_RAM_SIZE=4096
t CONFIG_BLK_DEV_INITRD is not set
# CONFIG_BLK_DEV_XD is not set
# CONFIG_BLK_DEV_DAC960 is not set
CONFIG_PARIDE_PARPORT=y
# CONFIG_PARIDE is not set
# CONFIG_BLK_DEV_IDE_MODES is not set
t CONFIG_BLK_CPQ_DA is not set
# CONFIG_BLK_CPQ_CISS_DA is not set
# CONFIG_BLK_DEV_HD is not set

#
# Networking options
#
CONFIG_PACKET=y
CONFIG_NETLINK=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK_DEV=y
CONFIG_FIREWALL=y
CONFIG_FILTER=y
CONFIG_UNIX=y
CONFIG_INET=y
CONFIG_IP_MULTICAST=y
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK=y
# CONFIG_IP_MULTIPLE_TABLES is not set
# CONFIG_IP_ROUTE_MULTIPATH is not set
# CONFIG_IP_ROUTE_TOS is not set
# CONFIG_IP_ROUTE_VERBOSE is not set
CONFIG_IP_ROUTE_LARGE_TABLES=y
CONFIG_IP_PNP=y
CONFIG_IP_PNP_DHCP=y
CONFIG_IP_PNP_BOOTP=y
# CONFIG_IP_PNP_RARP is not set
CONFIG_IP_FIREWALL=y
CONFIG_IP_FIREWALL_NETLINK=y
CONFIG_NETLINK_DEV=y
CONFIG_IP_TRANSPARENT_PROXY=y
CONFIG_IP_MASQUERADE=y
CONFIG_IP_MASQUERADE_ICMP=y
CONFIG_IP_MASQUERADE_MOD=y
CONFIG_IP_MASQUERADE_IPAUTOFW=y
CONFIG_IP_MASQUERADE_IPPORTFW=y
CONFIG_IP_MASQUERADE_MFW=y
CONFIG_IP_MASQUERADE_PPTP=y
# CONFIG_DEBUG_IP_MASQUERADE_PPTP is not set
# CONFIG_IP_MASQUERADE_IPSEC is not set
# CONFIG_IP_MASQUERADE_GENERIC is not set
```

```
CONFIG_IP_MASQUERADE_VS=y
CONFIG_IP_MASQUERADE_VS_TAB_BITS=12
CONFIG_IP_MASQUERADE_VS_RR=y
CONFIG_IP_MASQUERADE_VS_WRR=y
CONFIG_IP_MASQUERADE_VS_LC=y
CONFIG_IP_MASQUERADE_VS_WLC=y
# CONFIG_IP_ROUTER is not set
CONFIG_NET_IPIP=y
CONFIG_NET_IPGRE=y
CONFIG_NET_IPGRE_BROADCAST=y
CONFIG_IP_MROUTE=y
CONFIG_IP_PIMSM_V1=y
# CONFIG_IP_PIMSM_V2 is not set
CONFIG_IP_ALIAS=y
# CONFIG_ARPD is not set
CONFIG_SYN_COOKIES=y
CONFIG_INET_RARP=y
CONFIG_SKB_LARGE=y
CONFIG_IPV6=y
CONFIG_IPV6_EUI64=y
CONFIG_IPV6_NO_PB=y
# CONFIG_IPX is not set
# CONFIG_ATALK is not set
# CONFIG_X25 is not set
# CONFIG_LAPB is not set
# CONFIG_BRIDGE is not set
# CONFIG_LLC is not set
# CONFIG_ECONET is not set
# CONFIG_WAN_ROUTER is not set
# CONFIG_NET_FASTROUTE is not set
# CONFIG_NET_HW_FLOWCONTROL is not set
# CONFIG_CPU_IS_SLOW is not set

#
# QoS and/or fair queueing
#
# CONFIG_NET_SCHED is not set

#
# Telephony Support
#
# CONFIG_PHONE is not set
# CONFIG_PHONE_IXJ is not set

#
-# SCSI support
 #
# CONFIG_SCSI is not set

#
```

```
# I2O device support
#
# CONFIG_I20 is not set
# CONFIG_I20_PCI is not set
# CONFIG_I20_BLOCK is not set
# CONFIG_I2O_SCSI is not set

#
# IEEE 1394 (FireWire) support
#
# CONFIG_IEEE1394 is not set

#
# Network device support
#
CONFIG_NETDEVICES=y

#
# ARCnet devices
#
# CONFIG_ARCNET is not ' set
CONFIG_DUMMY=y
# CONFIG_BONDING is not set
# CONFIG_EQUALIZER is not set
# CONFIG_ETHERTAP is not set
f CONFIG_NET_SB1000 is not set

#
# Ethernet (10 or lOOMbit)
#
CONFIG_NET_ETHERNET=y
# CONFIG_NET_VENDOR_3COM is not set
# CONFIG_LANCE is not set
# CONFIG_NET_VEN.DOR_SMC is not set
# CONFIG_NET_VENDOR_RACAL is not set
CONFIG_RTL8139=y
CONFIG_RTL8139TOO=y
# CONFIG_NET_ISA is not set
# CONFIG_NET_EISA is not set
# CONFIG_NET_POCKET is not set

#
# Ethernet (1000 Mbit)
#
f CONFIG_ACENIC is not set
# CONFIG_HAMACHI is not set
# CONFIG_YELLOWFIN is not set
# CONFIG_SK98LIN is not set
# CONFIG_FDDI is not set
# CONFIG_HIPPI is not set
```

```
# CONFIG_PLIP is not set
CONFIG_PPP=y
# CONFIG_SLIP is not set
CONFIG_CIPE=y
# CONFIG_NET_RADIO is not set

t
# Token ring devices
I
# CONFIG_TR is not set
# CONFIG_NET_FC is not set
# CONFIG_RCPCI is not set
# CONFIG_SHAPER is not set

#
# Wan interfaces
#
# CONFIG_HOSTESS_SV11 is not set
# CONFIG_COSA is not set
# CONFIG_SEALEVEL_4021 is not set
I CONFIG_SYNCLINK_SYNCPPP is not set
# CONFIG_LANMEDIA is not set
# CONFIG_COMX is not set
# CONFIG_HDLC is not set
# CONFIG_DLCI is not set
# CONFIG_SBNI is not set

#
# Amateur Radio support
f
# CONFIG_HAMRADIO is not set

I
# IrDA (infrared) support
*
# CONFIG_IRDA is not set

#
# ISDN subsystem
#
# CONFIG_ISDN is not set

. #
# Old CD-ROM drivers (not SCSI, not IDE)
#
# CONFIG_CD_NO_IDESCSI is not set

#
# Character devices
#
```

```
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_SERIAL=y
CONFIG_SERIAL_CONSOLE=y
CONFIG_SERIAL_EXTENDED=y
CONFIG_SERIAL_MANY_PORTS=y
CONFIG_SERIAL_SHARE_IRQ=y
# CONFIG_SERIAL_DETECT_IRQ is not set
CONFIG_SERIAL_MULTIPORT=y
# CONFIG_HUB6 is not set
# CONFIG_SERIAL_NONSTANDARD is not set
CONFIG_UNIX98_PTYS=y
CONFIG_UNIX98_PTY_COUNT=256
# CONFIG_PRINTER is not set
CONFIG_MOUSE=y

#
# Mice
#
# CONFIG_ATIXL_BUSMOUSE is not set
CONFIG_BUSMOUSE=y
t CONFIG_MS_BUSMOUSE is not set
# CONFIG_PSMOUSE is not set
# CONFIG_82C710_MOUSE is not set
# CONFIG_PC110_PAD is not set

#
# Joysticks
I
# CONFIG_JOYSTICK is not set
# CONFIG_QIC02_TAPE is not set
t CONFIG_WATCHDOG is not set
CONFIG_NVRAM=y
CONFIG_RTC=y

#
# I2C support
t
t CONFIG_I2C is not set
# CONFIG_AGP is not set
t CONFIG_DRM is not set

#
t Video For Linux
i
# CONFIG_VIDEO_DEV is not set
# CONFIG_DTLK is not set

#
# Ftape, the floppy tape device driver
```

```
#
i CONFIG_FTAPE is not set
# CONFIG_UNIKEY is not set

#
# USB support
#
f CONFIG_USB is not set

#
#  Filesystems
#
CONFIG_QUOTA=y
CONFIG_AUTOFS_FS=y
# CONFIG_SUPERMOUNT is not set
# CONFIG_ADFS_FS is not set
# CONFIG_AFFS_FS is not set
# CONFIG_HFS_FS is not set
# CONFIG_FAT_FS is not set
# CONFIG_MSDOS_FS is not set
# CONFIG_UMSDOS_FS is not set
# CONFIG_VFAT_FS is not set
CONFIG_ISO9660_FS=y
# CONFIG_JOLIET is not set
# CONFIG_UDF_FS' is not set
# CONFIG_MINIX_FS is not set
# CONFIG_NTFS_FS is not set
# CONFIG_HPFS_FS is not set
CONFIG_PROC_FS=y
CONFIG_DEVPTS_FS=y
# CONFIG_QNX4FS_FS is not set
CONFIG_ROMFS_FS=y
CONFIG_EXT2_FS=y
# CONFIG_SYSV_FS is not set
# CONFIG_UFS_FS is not set
# CONFIG_REISERFS_FS is not set
# CONFIG_EFS_FS is not set

#
# Network File Systems
#
# CONFIG_CODA_FS is not set
CONFIG_NFS_FS=y
CONFIG_ROOT_NFS=y
CONFIG_NFSD=y
CONFIG_NFSD_SUN=y
CONFIG_SUNRPC=y
CONFIG_LOCKD=y
CONFIG_SMB_FS=y
CONFIG_NCP_FS=y
```

```
CONFIG_NCPFS_PACKET_SIGNING=y
CONFIG_NCPFS_IOCTL_LOCKING=y
CONFIG_NCPFS_STRONG=y
CONFIG_NCPFS_NFS_NS=y
# CONFIG_NCPFS_OS2_NS is not set
CONFIG_NCPFS_MOUNT_SUBDIR=y
CONFIG_NCPFS_NLS=y
CONFIG_NCPFS_EXTRAS=y

I
#  Partition Types
f
CONFIG_BSD_DISKLABEL=y
CONFIG_MAC_PARTITION=y
CONFIG_SMD_DISKLABEL=y
CONFIG_SOLARIS_X86_PARTITION=y
CONFIG_UNIXWARE_DISKLABEL=y
CONFIG_NLS=y

#
# Native Language Support
#
CONFIG_NLS_DEFAULT="cp437"
# CONFIG_NLS_CODEPAGE_437 is not set
# CONFIG_NLS_CODEPAGE_737 is not set
# CONFIG_NLS_CODEPAGE_775 is not set
t CONFIG_NLS_CODEPAGE_850 is not set
# CONFIG_NLS_CODEPAGE_852 is not set
# CONFIG_NLS_CODEPAGE_855 is not set
# CONFIG_NLS_CODEPAGE_857 is not set
# CONFIG_NLS_CODEPAGE_860 is not set
# CONFIG_NLS_CODEPAGE_861 is not set
# CONFIG_NLS_CODEPAGE_862 is not set
# CONFIG_NLS_CODEPAGE_863 is not set
# CONFIG_NLS_CODEPAGE_864 is not set
# CONFIG_NLS_CODEPAGE_865 is not set
CONFIG_NLS_CODEPAGE_866=y
# CONFIG_NLS_CODEPAGE_869 is not set
# CONFIG_NLS_CODEPAGE_874 is not set
# CONFIG_NLS_CODEPAGE_932 is not set
# CONFIG_NLS_CODEPAGE_936 is not set
# CONFIG_NLS_CODEPAGE_949 is not set
# CONFIG_NLS_CODEPAGE_950 is not set
CONFIG_NLS_ISO8859_1=y
CONFIG_NLS_IS08859_2=y
# CONFIG_NLS_ISO8859_3 is not set
# CONFIG_NLS_IS08859_4 is not set
CONFIG_NLS_ISO8859_5=y
# CONFIG_NLS_IS08859_6 is not set
I CONFIG_NLS_ IS08859_7 is not set
```

```
# CONFIG_NLS_ISO8859_8 is not set
# CONFIG_NLS_IS08859_9 is not set
# CONFIG_NLS_IS08859_14 is not set
CONFIG_NLS_ISO8859_15=y
CONFIG_NLS_KOI8_R=y

#
# Console drivers
#
CONFIG_VGA_CONSOLE=y
CONFIG_VIDEO_SELECT=y
# CONFIG_MDA_CONSOLE is not set
# CONFIG_FB is not set

#
# Sound
#
# CONFIG_SOUND is not set

#
# Kernel hacking
#
# CONFIG_MAGIC_SYSRQ is not set
```

Листинг В.2. Пример файла конфигурации обычного ядра

```
#
# Automatically generated by make menuconfig: don't edit
#

#
# Code maturity level options
#
CONFIG_EXPERIMENTAL=y

I
# Processor type and features
#
# CONFIG_M386 is not set
# CONFIG_M486 is not set
# CONFIG_M586 is not set
# CONFIG_M586TSC is not set
CONFIG_M686=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X86_INVLPG=y
CONFIG_X8 6_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_TSC=y
CONFIG_X8 6_GOOD_APIC=y
```

```
CONFIG_1GB=y
# CONFIG_2GB is not set
# CONFIG_MATH_EMULATION is not set
CONFIG_MTRR=y
# CONFIG_SMP is not set

#
t Loadable module support
#
CONFIG_MODULES=y
CONFIG_MODVERSIONS=y
CONFIG_KMOD=y

#
t General setup
I
CONFIG_BIGMEM=y
CONFIG_NET=y
CONFIG_PCI=y
# CONFIG_PCI_GOBIOS is not set
# CONFIG_PCI_GODIRECT is not set
CONFIG_PCI_GOANY=y
CONFIG_PCI_BIOS=y
CONFIG_PCI_DIRECT=y
CONFIG_PCI_QUIRKS=y
# CONFIG_PCI_OPTIMIZE is not set
CONFIG_PCI_OLD_PROC=y
# CONFIG_MCA is not set
# CONFIG_VISWS is not set
CONFIG_SYSVIPC=y
CONFIG_BSD_PROCESS_ACCT=y
CONFIG_SYSCTL=y
CONFIG_BINFMT_AOUT=m
CONFIG_BINFMT_ELF=y
CONFIG_BINFMT_MISC=m
CONFIG_BINFMT_JAVA=m
CONFIG_PARPORT=m
CONFIG_PARPORT_PC=m
# CONFIG_PARPORT_OTHER is not set
CONFIG_APM=y
# CONFIG_APM_IGNORE_USER_SUSPEND is not set
# CONFIG_APM_DO_ENABLE is not set
t CONFIG_APM_CPU_IDLE is not set
# CONFIG_APM_DISPLAY_BLANK is not set
# CONFIG_APM_IGNORE_SUSPEND_BOUNCE is not set
# CONFIG_APM_RTC_IS_GMT is not set
# CONFIG_APM_ALLOW_INTS is not set
# CONFIG_APM_REAL_MODE_POWER_OFF is not set
CONFIG_TOSHIBA=m
```

```
#
# Plug and Play support
I
CONFIG_PNP=y
CONFIG_PNP_PARPORT=m

#
# Block devices
#
CONFIG_BLK_DEV_FD=y
CONFIG_BLK_DEV_IDE=y
I CONFIG_BLK_DEV_HD_IDE is not set
CONFIG_BLK_DEV_IDEDISK=y
CONFIG_IDEDISK_MULTI_MODE=y
CONFIG_BLK_DEV_IDECD=y
# CONFIG_BLK_DEV_IDETAPE is not set
CONFIG_BLK_DEV_IDEFLOPPY=m
CONFIG_BLK_DEV_IDESCSI=m
f CONFIG_IDE_TASK_IOCTL_DEBUG is not set
CONFIG_BLK_DEV_CMD640=y
f CONFIG_BLK_DEV_CMD640_ENHANCED is not set
CONFIG_BLK_DEV_RZ1000=y
CONFIG_BLK_DEV_IDEPCI=y
CONFIG_IDEPCI_SHARE_IRQ=y
CONFIG_BLK_DEV_IDEDMA=y
# CONFIG_IDEDMA_AUTO is not set
CONFIG_IDEDMA_NEW_DRIVE_LISTINGS=y
CONFIG_IDEDMA_PCI_EXPERIMENTAL=y
CONFIG_IDEDMA_PCI_WIP=y
# CONFIG_BLK_DEV_OFFBOARD is not set
CONFIG_BLK_DEV_AEC62XX=y
CONFIG_AEC62XX_TUNING=y
CONFIG_BLK_DEV_ALI15X3=y
# CONFIG_WDC_ALI15X3 is not set
CONFIG_BLK_DEV_AMD7409=y
CONFIG_AMD7409_OVERRIDE=y
CONFIG_BLK_DEV_CMD64X=y
CONFIG_CMD64X_RAID=y
CONFIG_BLK_DEV_CY82C693=y
CONFIG_BLK_DEV_CS5530=y
CONFIG_BLK_DEV_HPT34X=y
CONFIG_HPT34X_AUTODMA=y
CONFIG_BLK_DEV_HPT366=y
HPT366_FIP=y
HPT366_MODE3=y
CONFIG_BLK_DEV_PIIX=y
CONFIG_BLK_DEV_OPTI621=y
CONFIG_BLK_DEV_PDC202XX=y
CONFIG_PDC202XX_BURST=y
CONFIG_PDC202XX_MASTER=y
```

```
CONFIG_BLK_DEV_SIS5513=y
CONFIG_BLK_DEV_TRM290=y
CONFIG_BLK_DEV_VIA82CXXX=y
# CONFIG_VIA82CXXX_TUNING is. not set
# CONFIG_IDE_CHIPSETS is not set
CONFIG_IDEDMA_IVB=y
CONFIG_BLK_DEV_LOOP=m
CONFIG_BLK_DEV_NBD=m
CONFIG_BLK_DEV_MD=y
CONFIG_AUTODETECT_RAID=y
CONFIG_MD_LINEAR=m
CONFIG_MD_STRIPED=m
CONFIG_MD_MIRRORING=m
CONFIG_MD_RAID5=m
I CONFIG_MD_TRANSLUCENT is not set
# CONFIG_MD_HSM is not set
CONFIG_BLK_DEV_RAM=y
CONFIG_BLK_DEV_RAM_SIZE=4096
CONFIG_BLK_DEV_INITRD=y
CONFIG_BLK_DEV_XD=m
CONFIG_BLK_DEV_DAC960=m
CONFIG_PARIDE_PARPORT=m
CONFIG_PARIDE=m
CONFIG_PARIDE_PD=m
CONFIG_PARIDE_PCD=m
CONFIG_PARIDE_PF=m
CONFIG_PARIDE_PT=m
CONFIG_PARIDE_PG=m
CONFIG_PARIDE_ATEN=m
CONFIG_PARIDE_BPCK=m
CONFIG_PARIDE_COMM=m
CONFIG_PARIDE_DSTR=m
CONFIG_PARIDE_FIT2=m
CONFIG_PARIDE_FIT3=m
CONFIG_PARIDE_EPAT=m
CONFIG_PARIDE_EPIA=m
CONFIG_PARIDE_FRIQ=m
CONFIG_PARIDE_FRPW=m
CONFIG_PARIDE_KBIC=m
CONFIG_PARIDE_KTTI=m
CONFIG_PARIDE_ON20=m
CONFIG_PARIDE_ON26=m
CONFIG_BLK_DEV_IDE_MODES=y
CONFIG_BLK_CPQ_DA=m
CONFIG_BLK_CPQ_CISS_DA=m
# CONFIG_BLK_DEV_HD is not set

#
# Networking options
#
```

```
CONFIG_PACKET=y
CONFIG_NETLINK=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK_DEV=y
CONFIG_FIREWALL=y
CONFIG_FILTER=y
CONFIG_UNIX=y
CONFIG_INET=y
CONFIG_IP_MULTICAST=y
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK=y
# CONFIG_IP_MULTIPLE_TABLES is not set
f CONFIG_IP_ROUTE_MULTIPATH is not set
# CONFIG_IP_ROUTE_TOS is not set
# CONFIG_IP_ROUTE_VERBOSE is not set
# CONFIG_IP_ROUTE_LARGE_TABLES is not set
# CONFIG_IP_PNP is not set
CONFIG_IP_FIREWALL=y
CONFIG_IP_FIREWALL_NETLINK=y
CONFIG_NETLINK_DEV=y
CONFIG_IP_TRANSPARENT_PROXY=y
CONFIG_IP_MASQUERADE=y
CONFIG_IP_MASQUERADE_ICMP=y
CONFIG_IP_MASQUERADE_MOD=y
CONFIG_IP_MASQUERADE_IPAUTOFW=m
CONFIG_IP_MASQUERADE_IPPORTFW=m
CONFIG_IP_MASQUERADE_MFW=m
CONFIG_IP_MASQUERADE_PPTP=m
# CONFIG_DEBUG_IP_MASQUERADE_PPTP is not set
# CONFIG_IP_MASQUERADE_IPSEC is not set
# CONFIG_IP_MASQUERADE_GENERIC is not set
CONFIG_IP_MASQUERADE_VS=y
CONFIG_IP_MASQUERADE_VS_TAB_BITS=12
CONFIG_IP_MASQUERADE_VS_RR=m
CONFIG_IP_MASQUERADE_VS_WRR=m
CONFIG_IP_MASQUERADE_VS_LC=m
CONFIG_IP_MASQUERADE_VS_WLC=m
# CONFIG_IP_ROUTER is not set
CONFIG_NET_IPIP=m
CONFIG_NET_IPGRE=m
CONFIG_NET_IPGRE_BROADCAST=y
# CONFIG_IP_MROUTE is not set
CONFIG_IP_ALIAS=y
# CONFIG_ARPD is not set
CONFIG_SYN_COOKIES=y
CONFIG_INET_RARP=m
CONFIG_SKB_LARGE=y
CONFIG_IPV6=m
CONFIG_IPV6_EUI64=y
```

```
CONFIG_IPV6_NO_PB=y.
CONFIG_IPX=m
# CONFI,G_IPX_INTERN is not set
# CONFIG_SPX is not set
CONFIG_ATALK=m
# CONFIG_X25 is not set
CONFIG_LAPB=m
# CONFIG_BRIDGE is not set
# CONFIG_LLC is not set
# CONFIG_ECONET is not set
CONFIG_WAN_ROUTER=m
# CONFIG_NET_FASTROUTE is not set
# CONFIG_NET_HW_FLOWCONTROL is not set
# CONFIG_CPU_IS_SLOW is not set

#
# QoS and/or fair queueing
#
# CONFIG_NET_SCHED is not set

#
# Telephony Support
t
CONFIG_PHONE=m
CONFIG_PHONE_IXJ=m

#
# SCSI support
f
CONFIG_SCSI=y
CONFIG_BLK_DEV_SD=y
CONFIG_CHR_DEV_ST=m
CONFIG_BLK_DEV_SR=y
CONFIG_BLK_DEV_SR_VENDOR=y
CONFIG_CHR_DEV_SG=m
# CONFIG_SCSI_MULTI_LUN is not set
CONFIG_SCSI_CONSTANTS=y
CONFIG_SCSI_LOGGING=y

#
# SCSI low-level drivers
#
CONFIG_BLK_DEV_3W_XXXX_RAID=m
CONFIG_SCSI_7000FASST=m
CONFIG_SCSI_ACARD=m
CONFIG_SCSI_AHA152X=m
CONFIG_SCSI_AHA1542=m
CONFIG_SCSI_AHA1740=m
CONFIG_SCSI_AACRAID=m
CONFIG_SCSI_AIC7XXX=m
```

```
# CONFIG_AIC7XXX_TCQ_ON_BY_DEFAULT is not set
CONFIG_AIC7XXX_CMDS_PER_DEVICE=8
CONFIG_AIC7XXX_PROC_STATS=y
CONFIG_AIC7XXX_RESET_DELAY=5
CONFIG_SCSI_IPS=m
CONFIG_SCSI_ADVANSYS=m
CONFIG_SCSI_IN2000=m
CONFIG_SCSI_AM53C974=m
CONFIG_SCSI_MEGARAID=m
CONFIG_SCSI_BUSLOGIC=m
# CONFIG_SCSI_OMIT_FLASHPOINT is not set
CONFIG_SCSI_CPQFCTS=m
CONFIG_SCSI_DTC3280=m
CONFIG_SCSI_DPT_I20=m
CONFIG_SCSI_EATA=m
CONFIG_SCSI_EATA_TAGGED_QUEUE=y
# CONFIG_SCSI_EATA_LINKED_COMMANDS is -not set
CONFIG_SCSI_EATA_MAX_TAGS=16
CONFIG_SCSI_EATA_DMA=m
CONFIG_SCSI_EATA_PIO=m
CONFIG_SCSI_FUTURE_DOMAIN=m
CONFIG_SCSI_GDTH=m
CONFIG_SCSI_GENERIC_NCR5380=m
t CONFIG_SCSI_GENERIC_NCR53C400 is not set
CONFIG_SCSI_G_NCR5380_PORT=y
# CONFIG_SCSI_G_NCR5380_MEM is not set
CONFIG_SCSI_INITIO=m
CONFIG_SCSI_INIA100=m
CONFIG_SCSI_PPA=m
CONFIG_SCSI_IMM=m
# CONFIG_SCSI_IZIP_EPP16 is not set
# CONFIG_SCSI_IZIP_SLOW_CTR is not set
CONFIG_PPSCSI=m
CONFIG_PPSCSI_T348=m
CONFIG_PPSCSI_T358=m
CONFIG_PPSCSI_ONSCSI=m
CONFIG_PPSCSI_SPARCSI=m
CONFIG_PPSCSI_EPSA2=m
CONFIG_PPSCSI_EPST=m
CONFIG_SCSI_NCR53C406A=m
CONFIG_SCSI_SYM53C416=m
CONFIG_SCSI_SIM710=m
CONFIG_SCSI_NCR53C7xx=m
t CONFIG_SCSI_NCR53C7xx_sync is not set
CONFIG_SCSI_NCR53C7xx_FAST=y
CONFIG_SCSI_NCR53C7xx_DISCONNECT=y
CONFIG_SCSI_NCR53C8XX=m
CONFIG_SCSI_SYM53C8XX=m
CONFIG_SCSI_NCR53C8XX_DEFAULT_TAGS=8
CONFIG_SCSI_NCR53C8XX_MAX_TAGS=32
```

```
CONFIG_SCSI_NCR53C8XX_SYNC=20
CONFIG_SCSI_NCR53C8XX_PROFILE=y
# CONFIG_SCSI_NCR53C8XX_IOMAPPED is not set
# CONFIG_SCSI_NCR53C8XX_PQS_PDS is not set
f CONFIG_SCSI_NCR53C8XX_SYMBIOS_COMPAT is not set
CONFIG_SCSI_PAS16=m
CONFIG_SCSI_PCI2000=m
CONFIG_SCSI_PCI2220I=m
CONFIG_SCSI_PSI240I=m
CONFIG_SCSI_QLOGIC_FAS=m
CONFIG_SCSI_QLOGIC_ISP=m
CONFIG_SCSI_QLOGIC_1280=m
CONFIG_SCSI_QLOGIC_FC=m
CONFIG_SCSI_QLOGIC_2x00=m
CONFIG_SCSI_SEAGATE=m
CONFIG_SCSI_DC395x_TRMS1040=m
CONFIG_SCSI_DC390T=m
# CONFIG_SCSI_DC390T_NOGENSUPP is not set
CONFIG_SCSI_T128=m
CONFIG_SCSI_U14_34F=m
# CONFIG_SCSI_U14_34F_LINKED_COMMANDS is not set
CONFIG_SCSI_U14_34F_MAX_TAGS=8
CONFIG_SCSI_ULTRASTOR=m
CONFIG_SCSI_DEBUG=m

#
t I2O device support
#
CONFIG_I2O=m
CONFIG_I2O_PCI=m
CONFIG_I2O_BLOCK=m
CONFIG_I20_SCSI=m

#
# IEEE 1394 (FireWire) support
#
CONFIG_IEEE1394=m
CONFIG_IEEE1394_PCILYNX=m
CONFIG_IEEE1394_PCILYNX_LOCALRAM=y
CONFIG_IEEE1394_PCILYNX_PORTS=y
CONFIG_IEEE1394_AIC5800=m
CONFIG_IEEE1394_OHCI1394=m
CONFIG_IEEE1394_VIDE01394=m
CONFIG_IEEE1394_RAWIO=m
# CONFIG_IEEE1394_VERBOSEDEBUG is not set

t
# Network device support
i
CONFIG_NETDEVICES=y
```

```
#
# ARCnet devices
#
# CONFIG_ARCNET is not set
CONFIG_DUMMY=m
CONFIG_BONDING=m
CONFIG_EQUALIZER=m
CONFIG_ETHERTAP=m
CONFIG_NET_SB1000=m

#
# Ethernet (10 or 100Mbit)
#
CONFIG_NET_ETHERNET=y
CONFIG_NET_VENDOR_3COM=y
CONFIG_EL1=m
CONFIG_EL2=m
CONFIG_ELPLUS=m
CONFIG_EL16=m
CONFIG_EL3=m
CONFIG_3C515=m
CONFIG_BC90X=m
CONFIG_VORTEX=m
CONFIG_LANCE=m
CONFIG_NET_VENDOR_SMC=y
CONFIG_WD80x3=m
CONFIG_ULTRA=m
CONFIG_ULTRA32=m
CONFIG_SMC9194=m
CONFIG_NET_VENDOR_RACAL=y
CONFIG_NI5010=m
CONFIG_NI52=m
CONFIG_NI65=m
CONFIG_RTL8139=m
CONFIG_RTL8139TOO=m
CONFIG_NET_ISA=y
CONFIG_AT1700=m
CONFIG_E2100=m
CONFIG_DEPCA=m
CONFIG_EWRK3=m
CONFIG_EEXPRESS=m
CONFIG_EEXPRESS_PRO=m
CONFIG_FMV18X=m
CONFIG_HPLAN_PLUS=m
CONFIG_HPLAN=m
CONFIG_HP100=m
CONFIG_ETH16I=m
CONFIG_NE2000=m
# CONFIG_SEEQ8005 is not set
CONFIG_SK_G16=y
```

```
CONFIG_NET_EISA=y
CONFIG_PCNET32=m
CONFIG_AC3200=m
CONFIG_APRICOT=m
CONFIG_CS89xO=m
CONFIG_DM9102=m
CONFIG_DE4X5=m
CONFIG_DEC_ELCP=m
CONFIG_DEC_ELCP_OLD=m
CONFIG_DGRS=m
CONFIG_EEXPRESS_PR0100=m
CONFIG_E100=m
CONFIG_EEXPRESS_PRO1000=m
CONFIG_LNE390=m
CONFIG_NE3210=m
CONFIG_NE2K_PCI=m
CONFIG_RL100ATX=m
CONFIG_TLAN=m
CONFIG_VIA_RHINE=m
CONFIG_SIS900=m
CONFIG_ES3210=m
CONFIG_EPIC100=m
# CONFIG_ZNET is not set
CONFIG_NET_POCKET=y
CONFIG_ATP=y
CONFIG_DE600=m
CONFIG_DE620=m

#
# Ethernet (1000 Mbit)
#
CONFIG_ACENIC=m
CONFIG_HAMACHI=m
CONFIG_YELLOWFIN=m
CONFIG_SK98LIN=m
CONFIG_FDDI=y
# CONFIG_DEFXX is not set
CONFIG_SKFP=m
# CONFIG_HIPPI is not set

#
# Appletalk devices
#
CONFIG_LTPC=m
CONFIG_COPS=m
CONFIG_COPS_DAYNA=y
CONFIG_COPS_TANGENT=y
CONFIG_IPDDP=m
CONFIG_IPDDP_ENCAP=y
CONFIG_IPDDP_DECAP=y
```

```
CONFIG_PLIP=m
CONFIG_PPP=m
CONFIG_SLIP=m
CONFIG_SLIP_COMPRESSED=y
CONFIG_SLIP_SMART=y
CONFIG_SLIP_MODE_SLIP6=y
CONFIG_CIPE=m
CONFIG_NET_RADIO=y
CONFIG_STRIP=m
CONFIG_WAVELAN=m
CONFIG_ARLAN=m

#
# Token ring devices
I
CONFIG_TR=y
CONFIG_IBMTR=m
CONFIG_IBMLS=m
CONFIG_IBMOL=m
CONFIG_SKTR=m
CONFIG_NET_FC=y
CONFIG_IPHASE5526=m
CONFIG_RCPCI=m
CONFIG_SHAPER=m

f
# Wan interfaces
#
CONFIG_HOSTESS_SV11=m
CONFIG_COSA=m
CONFIG_SEALEVEL_4021=m
CONFIG_SYNCLINK_SYNCPPP=m
CONFIG_LANMEDIA=m
CONFIG_COMX=m
CONFIG_COMX_HW_COMX=m
CONFIG_COMX_HW_LOCOMX=m
CONFIG_COMX_HW_MIXCOM=m
CONFIG_COMX_PROTO_PPP=m
CONFIG_COMX_PROTO_LAPB=m
CONFIG_COMX_PROTO_FR=m
CONFIG_HDLC=m
CONFIG_N2=m
CONFIG_C101=m
CONFIG_WANXL=m
CONFIG_PC300=m
CONFIG_PC300_X25=y
CONFIG_DLCI=m
CONFIG_DLCI_COUNT=24
CONFIG_DLCI_MAX=8
CONFIG_SDLA=m
```

```
CONFIG_WAN_DRIVERS=y
CONFIG_VENDOR_SANGOMA=m
CONFIG_WANPIPE_CARDS=4
# CONFIG_WANPIPE_FR is not set
CONFIG_WANPIPE_PPP=y
CONFIG_WANPIPE_CHDLC=y
CONFIG_SBNI=m

#
# Amateur Radio support
#
# CONFIG_HAMRADIO is not set

t
# IrDA (infrared) support
#
CONFIG_IRDA=m
CONFIG_IRLAN=m
CONFIG_IRCOMM=m
CONFIG_IRDA_ULTRA=y
CONFIG_IRDA_OPTIONS=y
CONFIG_IRDA_CACHE_LAST_LSAP=y
CONFIG_IRDA_FAST_RR=y
f CONFIG_IRDA_DEBUG is not set
CONFIG_IRDA_COMPRESSION=y
CONFIG_IRDA_DEFLATE=m

#
# Infrared-port device drivers
#
CONFIG_IRTTY_SIR=m
CONFIG_IRPORT_SIR=m
CONFIG_NSC_FIR=m
CONFIG_WINBOND_FIR=m
CONFIG_TOSHIBA_FIR=m
CONFIG_SMC_IRCC_FIR=m
CONFIG_DONGLE=y
CONFIG_ESI_DONGLE=m
CONFIG_ACTISYS_DONGLE=m
CONFIG_TEKRAM_DONGLE=m
CONFIG_GIRBIL_DONGLE=m
CONFIG_LITELINK_DONGLE=m
CONFIG_OLD_BELKIN_DONGLE=m
CONFIG_AIRPORT_DONGLE=m

#
# ISDN subsystem
#
CONFIG_ISDN=m
CONFIG_ISDN_PPP=y
```

```
CONFIG_ISDN_PPP_VJ=y
CONFIG_ISDN_MPP=y
CONFIG_ISDN_AUDIO=y
# CONFIG_ISDN_TTY_FAX is not set

t
# ISDN feature submodules
#
CONFIG_ISDN_DRV_LOOP=m
# CONFIG_ISDN_DIVERSION is not set

#
# Passive ISDN cards
#
CONFIG_ISDN_DRV_HISAX=m
CONFIG_HISAX_EURO=y
CONFIG_DE_AOC=y
# CONFIG_HISAX_NO_SENDCOMPLETE is not set
# CONFIG_HISAX_NO_LLC is not set
# CONFIG_HISAX_NO_KEYPAD is not set
CONFIG_HISAX_1TR6=y
CONFIG_HISAX_NI1=y
CONFIG_HISAX_16_0=y
CONFIG_HISAX_16_3=y
CONFIG_HISAX_TELESPCI=y
CONFIG_HISAX_SOBOX=y
CONFIG_HISAX_AVM_A1=y
CONFIG_HISAX_FRITZPCI=y
CONFIG_HISAX_AVM_A1_PCMCIA=y
CONFIG_HISAX_ELSA=y
CONFIG_HISAX_IX1MICROR2=y
CONFIG_HISAX_DIEHLDIVA=y
CONFIG_HISAX_ASUSCOM=y
CONFIG_HISAX_TELEINT=y
CONFIG_HISAX_HFCS=y
CONFIG_HISAX_SEDLBAUER=y
CONFIG_HISAX_SPORTSTER=y
CONFIG_HISAX_MIC=y
CONFIG_HISAX_NETJET=y
CONFIG_HISAX_NETJET_U=y
CONFIG_HISAX_NICCY=y
CONFIG_HISAX_ISURF=y
CONFIG_HISAX_HSTSAPHIR=y
CONFIG_HISAX_BKM_A4T=y
CONFIG_HISAX_SCT_QUADRO=y
CONFIG_HISAX_GAZEL=y
CONFIG_HISAX_HFC_PCI=y
CONFIG_HISAX_W6692=y
# CONFIG_HISAX_HFC_SX is not set
```

```
#
# Active ISDN cards
#
CONFIG_ISDN_DRV_ICN=m
CONFIG_ISDN_DRV_PCBIT=m
# CONFIG_ISDN_DRV_SC is not set
# CONFIG_ISDN_DRV_ACT2000 is not set
CONFIG_ISDN_DRV_EICON=m
CONFIG_ISDN_DRV_EICON_ISA=y
CONFIG_ISDN_CAPI=m
CONFIG_ISDN_CAPI_MIDDLEWARE=y
CONFIG_ISDN_CAPIFS=y
CONFIG_ISDN_DRV_AVMB1_B1ISA=y
CONFIG_ISDN_DRV_AVMB1_B1PCI=y
CONFIG_ISDN_DRV_AVMB1_B1PCIV4=y
CONFIG_ISDN_DRV_AVMB1_T1ISA=y
CONFIG_ISDN_DRV_AVMB1_B1PCMCIA=y
# CONFIG_ISDN_DRV_AVMB1_T1PCI is not set
# CONFIG_ISDN_DRV_AVMB1_C4 is not set
CONFIG_ISDN_DRV_AVMB1_VERBOSE_REASON=y

i
i Old CD-ROM drivers (not SCSI, not IDE)
#
# CONFIG_CD_NO_IDESCSI is not set

#
# Character devices
#
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_SERIAL=y
CONFIG_SERIAL_CONSOLE=y
CONFIG_SERIAL_EXTENDED=y
CONFIG_SERIAL_MANY_PORTS=y
CONFIG_SERIAL_SHARE_IRQ=y
# CONFIG_SERIAL_DETECT_IRQ is not set
CONFIG_SERIAL_MULTIPORT=y
# CONFIG_HUB6 is not set
CONFIG_SERIAL_NONSTANDARD=y
CONFIG_COMPUTONE=m
CONFIG_ROCKETPORT=m
CONFIG_CYCLADES=m
# CONFIG_CYZ_INTR is not set
CONFIG_DIGIEPCA=m
CONFIG_ESPSERIAL=m
CONFIG_MOXA_INTELLIO=m
CONFIG_MOXA_SMARTIO=m
CONFIG_ISI=m
CONFIG_RISCOM8=m
```

```
CONFIG_SPECIALIX=m
CONFIG_SPECIALIX_RTSCTS=y
CONFIG_SX=m
CONFIG_RIO=m
CONFIG_RIO_OLDPCI=y
CONFIG_STALDRV=y
CONFIG_STALLION=m
CONFIG_ISTALLION=m
CONFIG_SYNCLINK=m
CONFIG_N_HDLC=m
CONFIG_UNIX98_PTYS=y
CONFIG_UNIX98_PTY_COUNT=256
CONFIG_PRINTER=m
CONFIG_PRINTER_READBACK=y
CONFIG_MOUSE=y

#
i Mice
f
CONFIG_ATIXL_BUSMOUSE=m
CONFIG_BUSMOUSE=m
CONFIG_MS_BUSMOUSE=m
CONFIG_PSMOUSE=y
CONFIG_82C710_MOUSE=m
CONFIG_PC110_PAD=m

#
t Joysticks
tt
CONFIG_JOYSTICK=m
CONFIG_JOY_ANALOG=m
CONFIG_JOY_ASSASSIN=m
CONFIG_JOY_GRAVIS=m
CONFIG_JOY_LOGITECH=m
CONFIG_JOY_SIDEWINDER=m
CONFIG_JOY_THRUSTMASTER=m
CONFIG_JOY_CREATIVE=m
CONFIG_JOY_LIGHTNING=m
CONFIG_JOY_PCI=m
CONFIG_JOY_MAGELLAN=m
CONFIG_JOY_SPACEORB=m
CONFIG_JOY_SPACEBALL=m
CONFIG_JOY_WARRIOR=m
CONFIG_JOY_CONSOLE=m
CONFIG_JOY_DB9=m
CONFIG_JOY_TURBOGRAFX=m
# CONFIG_QIC02_TAPE is not set
CONFIG_WATCHDOG=y
```

```
#
I Watchdog Cards
#
# CONFIG_WATCHDOG_NOWAYOUT is not set
CONFIG_WDT=m
# CONFIG_WDT_501 is not set
CONFIG_SOFT_WATCHDOG=m
CONFIG_PCWATCHDOG=m
CONFIG_ACQUIRE_WDT=m
CONFIG_60XX_WDT=m
CONFIG_MIXCOMWD=m
CONFIG_NVRAM=m
CONFIG_RTC=y

#
I I2C support
f
CONFIG_I2C=m
CONFIG_I2C_ALGOBIT=m
CONFIG_I2C_PHILIPSPAR=m
CONFIG_I2C_ELV=m
CONFIG_I2C_VELLEMAN=m
CONFIG_I2C_ALGOPCF=m
CONFIG_I2C_ELEKTOR=m
CONFIG_I2C_MAINBOARD=y
CONFIG_I2C_ALI15X3=m
CONFIG_I2C_HYDRA=m
CONFIG_I2C_PIIX4=m
CONFIG_I2C_VIA=m
CONFIG_I2C_ISA=m
CONFIG_I2C_CHARDEV=m

#
# Hardware sensors support
#
CONFIG_SENSORS=m
CONFIG_SENSORS_ADM1021=m
CONFIG_SENSORS_ADM9240=m
CONFIG_SENSORS_GL518SM=m
CONFIG_SENSORS_LM75=m
CONFIG_SENSORS_LM78=m
CONFIG_SENSORS_LM80=m
CONFIG_SENSORS_SIS5595=m
CONFIG_SENSORS_W83781D=m
CONFIG_SENSORS_OTHER=y
CONFIG_SENSORS_EEPROM=m
CONFIG_SENSORS_LTC1710=m
CONFIG_AGP=m
CONFIG_AGP_INTEL=y
CONFIG_AGP_I810=y
```

```
CONFIG_AGP_VIA=y
CONFIG_AGP_AMD=y
CONFIG_AGP_SIS=y
CONFIG_AGP_ALI=y
CONFIG_DRM=y
CONFIG_DRM_TDFX=m
CONFIG_DRM_GAMMA=m
CONFIG_DRM_R128=m
CONFIG_DRM_I810=m
CONFIG_DRM_MGA=m

#
I Video For Linux
i
CONFIG_VIDEO_DEV=m
CONFIG_RADIO_RTRACK=m
CONFIG_RADIO_RTRACK2=m
CONFIG_RADIO_AZTECH=m
CONFIG_RADIO_CADET=m
CONFIG_RADIO_MIROPCM20=m
CONFIG_RADIO_GEMTEK=m
CONFIG_RADIO_TRUST=m
CONFIG_VIDEO_BT848=m
CONFIG_VIDEO_BWQCAM=m
CONFIG_VIDEO_CQCAM=m
CONFIG_VIDEO_CPIA=m
CONFIG_VIDEO_CPIA_PP=m
CONFIG_VIDEO_CPIA_USB=m
CONFIG_VIDEO_PMS=m
CONFIG_VIDEO_SAA5249=m
CONFIG_RADIO_SF16FMI=m
CONFIG_RADIO_TYPHOON=m
CONFIG_RADIO_TYPHOON_PROC_FS=y
CONFIG_RADIO_ZOLTRIX=m
CONFIG_VIDEO_ZORAN=m
CONFIG_VIDEO_BUZ=m
CONFIG_DTLK=m

#
# Ftape, the floppy tape device driver
#
CONFIG_FTAPE=m
CONFIG_ZFTAPE=m
CONFIG_ZFT_DFLT_BLK_SZ=10240
CONFIG_ZFT_COMPRESSOR=m
CONFIG_FT_NR_BUFFERS=3
# CONFIG_FT_PROC_FS is not set
CONFIG_FT_NORMAL_DEBUG=y
# CONFIG_FT_FULL_DEBUG is not set
# CONFIG_FT_NO_TRACE is not set
```

```
# CONFIG_FT_NO_TRACE_AT_ALL is not set
CONFIG_FT_STD_FDC=y
# CONFIG_FT_MACH2 is not set
# CONFIG_FT_PROBE_FC10 is not set
t CONFIG_FT_ALT_FDC is not set
CONFIG_FT_FDC_THR=8
CONFIG_FT_FDC_MAX_RATE=2000
CONFIG_FT_ALPHA_CLOCK=0
# CONFIG_UNIKEY is not set

#
# USB support
#
CONFIG_USB=m
# CONFIG_USB_DEBUG is not set
CONFIG_USB_DEVICEFS=y
CONFIG_USB_BANDWIDTH=y
CONFIG_USB_UHCI=m
CONFIG_USB_UHCI_ALT=m
CONFIG_USB_OHCI=m
CONFIG_USB_PRINTER=m
CONFIG_USB_SCANNER=m
CONFIG_USB_AUDIO=m
CONFIG_USB_ACM=m
CONFIG_USB_SERIAL=m
CONFIG_USB_SERIAL_GENERIC=y
CONFIG_USB_SERIAL_VISOR=m
CONFIG_USB_SERIAL_WHITEHEAT=m
CONFIG_USB_SERIAL_FTDI_SIO=m
CONFIG_USB_SERIAL_KEYSPAN_PDA=m
CONFIG_USB_SERIAL_KEYSPAN=m
CONFIG_USB_SERIAL_KEYSPAN_USA28=y
CONFIG_USB_SERIAL_KEYSPAN_USA28X=y
CONFIG_USB_SERIAL_KEYSPAN_USA19=y
CONFIG_USB_SERIAL_KEYSPAN_USA18X=y
CONFIG_USB_SERIAL_KEYSPAN_USA19W=y
CONFIG_USB_SERIAL_DIGI_ACCELEPORT=m
CONFIG_USB_SERIAL_OMNINET=m
CONFIG_USB_SERIAL_DEBUG=y
CONFIG_USB_IBMCAM=m
CONFIG_USB_OV511=m
CONFIG_USB_DC2XX=m
CONFIG_USB_MDC800=m
CONFIG_USB_STORAGE=m
# CONFIG_USB_STORAGE_DEBUG is not set
CONFIG_USB_DABUSB=m
CONFIG_USB_PLUSB=m
CONFIG_USB_PEGASUS=m
CONFIG_USB_RI0500=m
CONFIG_USB_DSBR=m
CONFIG_USB_BLUETOOTH=m
```

```
CONFIG_USB_KAWETH=m
CONFIG_USB_HID=m
CONFIG_USB_KBD=m
CONFIG_USB_MOUSE=m
CONFIG_USB_WACOM=m
CONFIG_USB_WMFORCE=m
CONFIG_INPUT_KEYBDEV=m
CONFIG_INPUT_MOUSEDEV=m
CONFIG_INPUT_MOUSEDEV_SCREEN_X=1024
CONFIG_INPUT_MOUSEDEV_SCREEN_Y=768
CONFIG_INPUT_JOYDEV=m
CONFIG_INPUT_EVDEV=m

#
# Filesystems
#
CONFIG_QUOTA=y
CONFIG_AUTOFS_FS=m
CONFIG_SUPERMOUNT=m
# CONFIG_ADFS_FS is not set
# CONFIG_AFFS_FS is not set
CONFIG_HFS_FS=m
CONFIG_FAT_FS=m
CONFIG_MSDOS_FS=m
# CONFIG_UMSDOS_FS is not set
CONFIG_VFAT_FS=m
CONFIG_IS09660_FS=y
CONFIG_JOLIET=y
CONFIG_UDF_FS=m
# CONFIG_UDF_RW is not set
CONFIG_MINIX_FS=m
CONFIG_NTFS_FS=m
# CONFIG_NTFS_RW is not set
CONFIG_HPFS_FS=m
CONFIG_PROC_FS=y
CONFIG_DEVPTS_FS=y
# CONFIG_QNX4FS_FS is not set
CONFIG_ROMFS_FS=m
CONFIG_EXT2_FS=y
CONFIG_SYSV_FS=m
CONFIG_UFS_FS=m
# CONFIG_UFS_FS_WRITE is not set
CONFIG_REISERFS_FS=m
# CONFIG_REISERFS_CHECK is not set
CONFIG_EFS_FS=m
CONFIG_SGI_PARTITION=y

#
# Network File Systems
i
CONFIG_CODA_FS=m
```

```
CONFIG_NFS_FS=m
CONFIG_NFSD=m
CONFIG_NFSD_SUN=y
CONFIG_SUNRPC=m
CONFIG_LOCKD=m
CONFIG_SMB_FS=m
CONFIG_NCP_FS=m
CONFIG_NCPFS_PACKET_SIGNING=y
CONFIG_NCPFS_IOCTL_LOCKING=y
CONFIG_NCPFS_STRONG=y
CONFIG_NCPFS_NFS_NS=y
CONFIG_NCPFS_OS2_NS=y
CONFIG_NCPFS_SMALLDOS=y
CONFIG_NCPFS_MOUNT_SUBDIR=y
CONFIG_NCPFS_NLS=y
CONFIG_NCPFS_EXTRAS=y

#
# Partition Types
#
CONFIG_BSD_DISKLABEL=y
CONFIG_MAC_PARTITION=y
CONFIG_SMD_DISKLABEL=y
CONFIG_SOLARIS_X86_PARTITION=y
CONFIG_UNIXWARE_DISKLABEL=y
CONFIG_NLS=y

#
# Native Language Support
#
CONFIG_NLS_DEFAULT="cp437"
CONFIG_NLS_CODEPAGE_437=m
CONFIG_NLS_CODEPAGE_737=m
CONFIG_NLS_CODEPAGE_775=m
CONFIG_NLS_CODEPAGE_850=m
CONFIG_NLS_CODEPAGE_852=m
CONFIG_NLS_CODEPAGE_855=m
CONFIG_NLS_CODEPAGE_857=m
CONFIG_NLS_CODEPAGE_860=m
CONFIG_NLS_CODEPAGE_861=m
CONFIG_NLS_CODEPAGE_862=m
CONFIG_NLS_CODEPAGE_863=m
CONFIG_NLS_CODEPAGE_864=m
CONFIG_NLS_CODEPAGE_865=m
CONFIG_NLS_CODEPAGE_866=m
CONFIG_NLS_CODEPAGE_869=m
CONFIG_NLS_CODEPAGE_874=m
CONFIG_NLS_CODEPAGE_932=m
CONFIG_NLS_CODEPAGE_936=m
CONFIG_NLS_CODEPAGE_949=m
```

```
CONFIG_NLS_CODEPAGE_950=m
CONFIG_NLS_ISO8859_1=m
CONFIG_NLS_ISO8859_2=m
CONFIG_NLS_ISO8859_3=m
CONFIG_NLS_ISO8859_4=m
CONFIG_NLS_ISO8859_5=m
CONFIG_NLS_ISO8859_6=m
CONFIG_NLS_ISO8859_7=m
CONFIG_NLS_ISO8859_8=m
CONFIG_NLS_ISO8859_9=m
CONFIG_NLS_ISO8859_14=m
CONFIG_NLS_ISO8859_15=m
CONFIG_NLS_KOI8_R=m

t
# Console drivers
#
CONFIG_VGA_CONSOLE=y
CONFIG_VIDEO_SELECT=y
CONFIG_MDA_CONSOLE=m
CONFIG_FB=y
CONFIG_DUMMY_CONSOLE=y
# CONFIG_UNICON is not set
CONFIG_FB_PM2=m
CONFIG_FB_ATY=m
CONFIG_FB_VESA=y
# CONFIG_FB_VGA16 is not set
CONFIG_VIDEO_SELECT=y
CONFIG_FB_MATROX=m
CONFIG_FB_MATROX_MILLENIUM=y
CONFIG_FB_MATROX_MYSTIQUE=y
CONFIG_FB_MATROX_G100=y
CONFIG_FB_MATROX_MULTIHEAD=y
# CONFIG_FB_ATY128 is not set
# CONFIG_FB_VIRTUAL is not set
# CONFIG_FBCON_ADVANCED is not set
CONFIG_FBCON_CFB8=y
CONFIG_FBCON_CFB16=y
CONFIG_FBCON_CFB24=y
CONFIG_FBCON_CFB32=y
# CONFIG_FBCON_FONTWIDTH8_ONLY is not set
tt CONFIG_FBCON_FONTS is not set
CONFIG_FONT_8x8=y
CONFIG_FONT_8x16=y

#
# Sound
#
CONFIG_SOUND=m
CONFIG_SOUND_CMPCI=m
```

```
CONFIG_SOUND_CMPCI_FM=y
CONFIG_SOUND_CMPCI_MIDI=y
CONFIG_SOUND_CS4281=m
CONFIG_SOUND_FUSION=m
CONFIG_SOUND_EMU10K1=m
CONFIG_SOUND_ES1370=m
CONFIG_SOUND_ES1371=m
CONFIG_SOUND_MAESTRO=m
CONFIG_SOUND_ESSSOLO1=m
CONFIG_SOUND_ICH=m
CONFIG_SOUND_SONICVIBES=m
CONFIG_SOUND_TRIDENT=m
CONFIG_SOUND_MSNDCLAS=m
# CONFIG_MSNDCLAS_HAVE_BOOT is not set
CONFIG_MSNDCLAS_INIT_FILE="/etc/sound/msndinit.bin"
CONFIG_MSNDCLAS_PERM_FILE="/etc/sound/msndperm.bin"
CONFIG_SOUND_MSNDPIN=m
# CONFIG_MSNDPIN_HAVE_BOOT is not set
CONFIG_MSNDPIN_INIT_FILE="/etc/sound/pndspini.bin"
CONFIG_MSNDPIN_PERM_FILE="/etc/sound/pndsperm.bin"
CONFIG_SOUND_VIA82CXXX=m
CONFIG_SOUND_OSS=m
CONFIG_SOUND_PAS=m
CONFIG_SOUND_SB=m
CONFIG_SOUND_GUS=m
CONFIG_GUS16=y
CONFIG_GUSMAX=y
CONFIG_SOUND_MPU401=m
CONFIG_SOUND_PSS=m
# CONFIG_PSS_MIXER is not set
CONFIG_SOUND_MSS=m
CONFIG_SOUND_SSCAPE=m
CONFIG_SOUND_TRIX=m
CONFIG_SOUND_MAD16=m
CONFIG_MAD16_OLDCARD=y
# CONFIG_SOUND_WAVEFRONT is not set
CONFIG_SOUND_CS4232=m
CONFIG_SOUND_OPL3SA2=m
CONFIG_SOUND_MAUI=m
CONFIG_SOUND_SGALAXY=m
CONFIG_SOUND_AD1816=m
CONFIG_SOUND_OPL3SA1=m
# CONFIG_SOUND_SOFTOSS is not set
CONFIG_SOUND_YM3812=m
CONFIG_SOUND_VMIDI=m
CONFIG_SOUND_UART6850=m
CONFIG_SOUND_NM256=m
CONFIG_SOUND_YMPCI=m
```

```
#
# Additional low level sound drivers
#
CONFIG_LOWLEVEL_SOUND=y
CONFIG_ACI_MIXER=m
CONFIG_VIDEO_MSP3400=m
CONFIG_AWE32_SYNTH=m
CONFIG_AEDSP16=m
CONFIG_AEDSP,16_BASE=220
CONFIG_MPU_BASE=330
CONFIG_SC6600=y
CONFIG_SC6600_JOY=y
CONFIG_SC6600_CDROM=4
CONFIG_SC6600_CDROMBASE=0
CONFIG_AEDSP16_SBPRO=y
CONFIG_AEDSP16_BASE=220
CONFIG_AEDSP16_SB_IRQ=5
CONFIG_AEDSP16_SB_DMA=0
CONFIG_AEDSP16_MPU401=y
CONFIG_AEDSP16_MPU_IRQ=5

#
# Kernel hacking
I
CONFIG_MAGIC_SYSRQ=y
```

Приложение Г. Ссылки

Дистрибутивы и ядра Linux

Дистрибутив	Официальный Web-сайт
Red Hat Linux	http://www.redhat.com
Mandrake Linux	http://www.linux-mandrake.com/ru
ASP Linux	http://www.asplinux.ru
ALT Linux	http://www.altlinux.ru
KSI Linux	http://www.ksi-linux.com
Black Cat Linux	http://www.blackcatlinux.com
Open Linux (Caldera)	http://www.calderasystems.com
Yellow Dog Linux (для Macintosh)	http://www.terrasoftsolutions.com
SuSE Linux	http://www.suse.com
Debian Linux	http://www.debian.org
Infomagic	http://www.infomagic.com
LinuxPPC (версия для MacPowerPC)	http://www.linuxppc.com
Turbo Linux (Pacific Hi-Tech)	http://www.turbolinux.com
Slackware Linux	http://www.slackware.com
Ядро Linux	http://www.kernel.org

Документация по Linux

Название ресурса	Web-сайт
Linux.Ru	http://www.linux.ru
Linux.Ru.Net	http://linux.ru.net
Linux RSP Web Site (Russian Security Project)	http://www.linuxrsp.ru
Linux.Org.Ru	http://www.linux.org.ru
Рубрика «Линуксоид» (Софтерра)	http://www.softerra.ru/freeos/
Linux World Kiev	http://linux.kiev.ua
Корпоративное использование Linux	http://linux.cn.ua
Linuxoid — документация и программы для Linux	http://www.linuxoid.ru
Различная документация по Linux	http://www.nevod.ru/linux
Безопасность в Internet	http://www.atlas.net.ru
SQUID: зона особого внимания	http://squid.opennet.ru/
Виртуальная энциклопедия «Linux по-русски»	http://linux-ve.chat.ru
Советы по использованию GIMP	http://gimp.linux.ru.net
SAG (руководство системного администратора)	http://hibase.cs.hut/fi/~liw/linux/sag
FTP-узел группы LDP (Linux Documentation Project)	ftp://metalab.unc.edu/pub/linux/docs
Web-узел группы LDP	http://metalab.unc.edu/LDP/
Библиотека почты	http://www.mailinfo.ru/
Страничка В. Водолазского	http://come.to/vodolaz/
Denis Kolisnichenko's Web Site	http://dkws.narod.ru
Клуб разработчиков PHP	http://phpclub.net
Dago.Org	http://www.dago.org

Рабочие столы

Рабочий стол	Web-сайт
K Desktop (KDE)	http://www.kde.org
GNOME	http://www.gnome.org
Менеджер окон Enlighment	http://www.enlightment.org
Менеджер окон FVWM	http://www.fvwm.org
Менеджер окон Window Maker	http://www.windowmaker.org
Менеджер окон Afterstep	http://www.afterstep.org
Менеджер окон Blackbox	http://www.blackbox.org
XFce: Графический менеджер	http://www.xfce.org/
Информация о X11	http://www.x11.org
Темы для KDE и Gnome	http://www.themes.org
GNU-версия системы X Window для Linux	http://www.xfree86.org

Архивы и ресурсы ПО

Архив	Web-сайт
Архив ПО	http://linuxwww.db.erau.edu
Библиотека RPM-пакетов	http://rpmfind.net
Новый софт для Linux	http://freshmeat.com
GNU-архив	http://www.gnu.org
Игры	http://www.linuxgames.org
Игры	http://www.happypenguin.org
Игра Quake	http://www.linuxquake.org
Драйверы Open Sound Systems	http://www.opensound.com
Драйверы для принтеров HP	http://hpinkjet.sourceforge.net/
Java-апплеты для Linux	http://www.blackdown.org
Различные ссылки	http://www.linuxlinks.com

Серверы

Название	Web-сайт
Web-сервер Russian Apache	http://www.apache.ru
Web-сервер Apache	http://www.apache.org
FTP server ProFTPD	http://www.proftpd.org
Internet Software Consortium: BIND, INN, DHCP	http://www.ics.org
BIND — Сервер Доменных Имен	http://www.isc.org/products/BIND/
Sendmail	http://www.sendmail.org
Альтернативный почтовик QMAIL	http://www.qmail.org
Прокси-сервер Squid	http://www.squid.org
Сервер Samba	http://www.samba.org
POP3 сервер Qpopper	http://www.eudora.com/free/qpop.html
Postfix — почтовый сервер (SMTP)	http://www.postfix.org/start.html
MOSIX — программный модуль для реализации кластеров на Linux	http://www.mosix.cs.huji.ac.il/txt_distributionhtml
Zope — система создания и поддержки высокопроизводительных серверов	http://www.zope.org
Comanche — утилита для конфигурирования web-сервера Apache	http://www.covalent.net/projects/comanche/
ASPSeek 1.1.3 — поисковая машина	http://www.aspseek.org
PPP — Point To Point Protocol	ftp://ftp.linuxcare.com.au/pub/ppp/
K12 Linux Terminal Server Project	http://k12os.org/

Безопасность

Название	Web-сайт
PAM: подключаемые модули аутентификации	http://www.kernel.org/pub/linux/libs/pam/index.html
CFS: криптографическая файловая система	ftp://ftp.research.att.com/dist/mab
TCFS: прозрачная криптографическая файловая система	http://edu-gw.dia.unisa.it/tcfs
Проект Generic Graphic Interface	http://synergy.caltech.edu/~ggi/
SATAN: Security Administrators Tool for Analyzing Networks	http://www.trouble.org/~zen/satan/satan.html
PGP: Криптография с открытым ключом	http://www.pgp.com
Secure Shell	http://www.cs.hut.fi/ssh/
Журналируемая файловая система от IBM - JFS	http://oss.software.ibm.com/developer/opensource/jfs/
ReiserFS 3.6.25: Журналируемая файловая система	http://devlinux.com/pub/namesys/
SAINT — программа для выявления брешей в защите системы	http://www.wwdsi.com/saint/
Безопасность в Internet	http://www.atlas.net.ru
Libsafe — бесплатное ПО защиты от атак под Linux	http://www.avaya.ru/

Программное *обеспечение для Linux*

Ресурс	Адрес
FTP-узел компании Red Hat	ftp://ftp.redhat.com
Обновления пакетов Red Hat Linux	ftp://updates.redhat.com
Пакеты программ	ftp://contrib.redhat.com
FTP-узел Open Linux	ftp://ftp.calderasystems.com
FTP-узел SuSE Linux	ftp://ftp.suse.com
FTP-узел Debian Linux	ftp://ftp.debian.org
FTP-узел LinuxPPC	ftp://ftp.linuxppc.com
FTP-узел Turbo Linux	ftp://ftp.turbolinux.com
Обучающее программное обеспечение ShoolForge	http://schoolforge.net/
Обучающее программное обеспечение Seul	http://richtech.ca/seul/

Информационные узлы

Название	Web-сайт
Linux Weekly News	http://www.lwn.com
Linux.com	http://www.linux.com
Linux Today	http://www.linuxtoday.com
Linux Power	http://www.linuxpower.org
Linux Focus	http://www.linuxfocus.org
Linux World	http://www.linuxworld.org
Linux Mail	http://www.linuxmail.org
Linux Journal	http://www.linuxjournal.org
Linux Gazette	http://www.linuxgazette.org
Linux Online	http://www.linux.org
Linux International	http://www.li.org
Linux European	http://www.uk.linux.org
Последние новости по ядру Linux	http://www.kernelnotes.org
Linux-форум	http://slashdot.org
Linux News	http://www.linuxnews.ru
Linux Web Site Watcher	http://webwatcher.org
Linux.Ru.Net	http://linux.ru.net
Linux-Online	http://www.linux-online.ru
Linux.Ru	http://www.linux.ru

Группы новостей

comp.os.linux.announce.....объявления о Linux-разработках.
comp.os.linux.development.apps..... для программистов, разрабатывающих Linux-приложения.
comp.os.linux.development.system .... то же для системных программистов.
comp.os.linux.hardware.....информация о железках, совместимых с Linux.
comp.os.linux.admin.....вопросы системного администрирования.
comp.os.linux.misc.....специальные вопросы.
comp.os.linux.setup.....вопросы по инсталляции Linux.
comp.os.linux.networking.....Linux и сеть.

СУБД и офис

Название	Web-сайт
СУБД Oracle	http://www.oracle.com
СУБД Sybase	http://www.sybase.com
СУБД IBM - DB2	http://www.software.ibm.com/data/db2/
Informix for Linux	http://www.informix.com/linux
Ingress II	http://www.cai.com/products/ingres.htm
СУБД AdabasD	http://www.softwareag.com
MySQL	http://www.mysql.com
GNU SQL	http://www.ispras.ru/~kml/gss
InterBase SQL Server	http://www.borland.com
PostgreSQL	http://www.postgresql.org
Интерфейс Falgship для файлов баз данных xBase	http://www.fship.com/free.html
Персональная СУБД Gaby для Gnome	http://gaby.netpedia.net

Название	Web-сайт
K Office	http://koffice.kde.org
Corel (WordPerfect, Corel Linux)	http://linux.corel.com
Star Office	http://www.stardivision.com
Gnome Workshop Project	http://www.gnome.org/gw.html
OpenOffice — бинарные файлы	http://www.openoffice.org/dev_docs/source/get_binaries.html
Бразуер Mozilla	http://mozilla.org/releases/
Opera — простой и быстрый Web-браузер	http://www.opera.com/download/linux.html
Netscape Communicator	http://www.netscape.com
Galeon: Браузер под GNOME	http://galeon.sourceforge.net/
Amaya 4.2.1: Графический браузер от консорциума W3C	http://www.w3.org/Amaya/
Links: текстовый браузер	http://artax.karlin.mff.cuni.cz/~mikulas/links/
Lynx: текстовый браузер	http://lynx.browser.org
Quanta+: HTML-редактор	http://quanta.sourceforge.net/
Balsa: Почтовый клиент с графическим интерфейсом	http://www.balsa.net/
Evolution: Графический почтовый клиент	http://www.helixcode.com/apps/evolution.php3
Sylpheed — графический почтовый агент	http://sylpheed.good-day.net/
VIM — текстовый редактор	http://www.vim.org
XEmacs — текстовый редактор и система разработки приложений	http://www.xemacs.org/

Программирование под Linux

Название	Web-сайт
Ресурсы для программирования в Linux	http://www.linuxprogramming.org
Продукты, разработанные с использованием Tk/Tc	http://www.scriptics.com
Java-ресурсы	http://java.sun.com
Сценарии Perl	http://www.perl.com
Разработка приложений для GNOME	http://developers.gnome.org
Разработка приложений для KDE	http://developer.kde.org
OpenProjects Network	http://www.openprojects.nu
Free Pascal for Linux	http://www.freepascal.org
GTK+ — библиотека для создания графических интерфейсов	http://www.gtk.org/
Язык программирования Python	http://www.python.org/
Клуб разработчиков PHP	http://phpclub.net
PHP: Язык программирования для Web	http://www.php.net

Создание загрузочных дисков

Название	Web(FTP)-узел
Bootkit	ftp://sunsite.unc.edu/pub/Linux/system/Recovery/Bootkit-vw.tar.gz
Rescue Shell Scripts	ftp://sunsite.unc.edu/pub/Linux/system/Recovery/rescue.tgz
Cat Rescue	ftp://gd.cs.csufresno.edu/pub/sun4bin/src/CatRescue100.tgz
SAR — Search and Rescue	http://www.icce.rug.nl/karel/programs/SAR.html
SAR — Search and Rescue	ftp://ftp.icce.rug.nl/pub/unix/SAR-vw.tar.gz (vvv - номер версии)
Yard	http://www.cs.umass.edu/~fawcett/yard.html

Другие программы

Название	Web-сайт
Pine: Программа для чтения почты и новостей	http://www.washington.edu/pine/
Fetchmail: Утилита для получения почты	http://www.tuxedo.org/-esr/fetchmail/
Zope: Сервер и набор утилит для создания и администрирования динамических сайтов	http://www.zope.org
GNU Parted: Утилита для работы с разделами диска	http://www.gnu.org/software/parted/parted.html
Licq: Клон ICQ	http://licq.sourceforge.net
Modutils: Программы для управления модулями ядра	ftp://ftp.ocs.com.aU/pub/modutils/v2.4/
Linuxconf: Утилита всестороннего конфигурирования системы	http://www.solucorp.qc.ca/linuxconf/
NcFTP: FTP-client	http://www.ncftp.com/ncftp/
FreeAmp: проигрыватель аудиофайлов	http://www.freeamp.org/
LILO: Linux LOader	ftp://sd.dynhost.com/pub/linux/lilo/
WINE: Среда для запуска WIN32, WIN16 и DOS программ	ftp://metalab.unc.edu/pub/Linux/ALPHA/wine/development/
VMWare — виртуальная машина	http://www.vmware.com
Всевозможные эмуляторы (dos, win)	http://www.bochs.com
OpenGUI: Высокоуровневая графическая библиотека	http://www.tutok.sk/fastgl/
Downloader: Программа с графическим интерфейсом для загрузки, догрузки и не только, файлов	http://www.krasu.ru/soft/chuchelo

Издательство
Наука и Техника

Россия: Санкт-Петербург,
пр. Обуховской обороны, 107
(812)-567-70-25, (812)-567-70-26
E-mail: nit@mail.wplus.net
для писем: 193029 СПб. а/я 44,
ООО «Наука и Техника»

Украина: Киев
(044)-516-38-66
E-mail: nit@ambernet.kiev.ua
для писем: 02166 Киев,
ул. Курнатовка, 9/2.1
«Наука и Техника»

Новая серия "Секреты мастерства"

Книги этой серии ориентированы на подготовленного читателя и представляют собой отличные руководства, превосходные практические и справочные пособия. Детальное рассмотрение материала позволит вам изучить тонкости, далеко выходящие за рамки общих сведений. Авторы не только описывают те или иные технологии, но и делятся своим профессиональным опытом их использования. Читая книги серии "Секреты мастерства", вы приобретаете уникальные знания и повышаете свой профессиональный уровень.

Секреты Мастерства

WEB-страница:
www.pubnit.com
ИНТЕРНЕТ-МАГАЗИН:
www.nit.com.ru