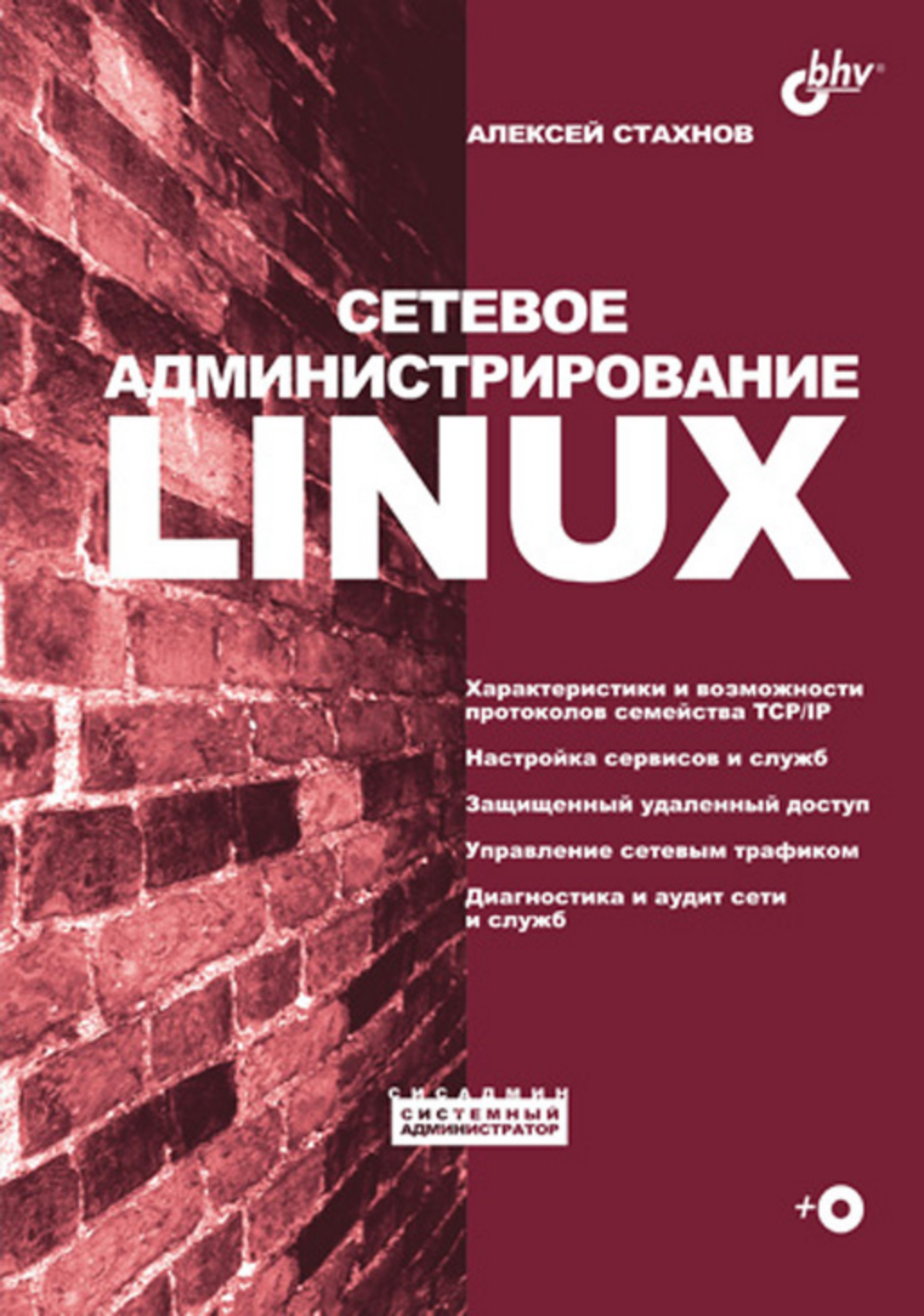

bhv
АЛЕКСЕЙ СТАХНОВ
СЕТЕВОЕ АДМИНИСТРИРОВАНИЕ
LINUX
Характеристики и возможности протоколов семейства TCP/IP
Настройка сервисов и служб
Защищенный удаленный доступ
Управление сетевым трафиком
Диагностика и аудит сети и служб
СИСАДМИН
СИСТЕМНЫЙ АДМИНИСТРАТОР
+

Алексей Стахнов

СЕТЕВОЕ АДМИНИСТРИРОВАНИЕ LINUX

Санкт-Петербург
«БХВ-Петербург»
2004

УДК 681.3.06
ББК 32.973-018.2
С78

Стахнов А. А.

С78 Сетевое администрирование Linux. — СПб.: БХВ-Петербург, 2004. — 480 с.: ил.

ISBN 5-94157-277-8

В книге представлены теоретические и практические знания, позволяющие хорошо понимать процессы, происходящие в сети. Рассматриваются сетевые модели, протоколы, адреса, службы, конфигурирование сетевых интерфейсов, настройка серверов FTP, Proxy, INN, Apache, Samba, Mars, обсуждается сетевая файловая система, взаимодействие Linux с другими операционными системами. Описывается конфигурирование локальной сети: сетевые принтеры, шлюз в Интернет, настройка Firewall, учет трафика и т. п. Приведено множество программ, помогающих обслуживать сеть и заботиться о ее безопасности. Рассказано, как создать, настроить и обеспечить надежное функционирование сервера небольшой локальной сети, способного выполнять большинство типовых задач. На прилагаемом компакт-диске находятся последние версии программного обеспечения, рассмотренного в книге.

Для системных администраторов

УДК 681.3.06
ББК 32.973-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Григорий Добин*
Редактор	*Екатерина Капалыгина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Оформление серии	*Via Design*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 27.02.04.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 38,7.
Тираж 4000 экз. Заказ №
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-277-8

© Стахнов А. А., 2004
© Оформление, издательство "БХВ-Петербург", 2004

Содержание

Введение **15**

Благодарности 15
Почему написана эта книга 16
Для кого написана эта книга 16
Структура книги 16
Как со мной связаться 18

ЧАСТЬ I. СЕТЕВЫЕ ПРОТОКОЛЫ И КОНФИГУРИРОВАНИЕ **19**

Глава 1. Сетевые протоколы **21**

Модели сетевых взаимодействий 21
Терминология 21
Модель взаимодействия открытых систем (OSI) 23
Модель сетевого взаимодействия TCP/IP 25
Сопоставление сетевых моделей OSI и TCP/IP 25
Сетевые протоколы 25
Семейство протоколов TCP/IP 26
Протоколы межсетевого уровня (Интернет) 27
Протокол IP 27
Формат пакета IPv4 27
Протокол IPv6 29
Адресация в IPv6 30
Сетевые пакеты 30
Маршрутизация пакетов 31
Протоколы маршрутизации 31
Адресация в TCP/IP 32
Протокол адресации ARP/RARP 34
Протокол ICMP 34
Протоколы транспортного уровня 37
Протокол TCP 38
Протокол UDP 39
Протоколы уровня приложений 39
Протокол FTP 39
Протокол SMTP 40
Протокол Telnet 40
Сетевая файловая система NFS 40
Протокол IPX 40
Протокол AppleTalk 41
Протокол NetBIOS 41
Протокол DECnet 41
Литература и ссылки 41

Глава 2. Настройка сети TCP/IP **43**

Конфигурирование сетевых интерфейсов 43
Настройка локального интерфейса lo 44

Настройка Ethernet-карты (eth0) 44
Конфигурирование статических маршрутов и маршрута по умолчанию....45
Использование DHCP 46
Статический ARP 46
Настройка DNS 47
Протокол PPP 49
Общая информация 49
Свойства протокола PPP 50
Составляющие PPP....51
Функционирование протокола PPP 51
Поддерживаемое оборудование 51
Структура пакета протокола PPP 52
PPP-протокол управления соединением (LCP) 53
Сокращения, используемые при описании протокола PPP....53
Стандарты, описывающие протокол PPP 55
Протокол SLIP/CSLIP....56
Протокол SLIP 56
Протокол CSLIP 57
Процесс init....57
Конфигурационный файл init (/etc/inittab) 59
Основные конфигурационные файлы....63
Файл rc.sysinit 64
Файл rc....65
Файл rc.local 70
Другие файлы, влияющие на процесс загрузки 70
Средства тестирования сети и сетевых настроек....71
Утилита ifconfig 71
Утилита hostname 71
Утилита ping 72
Утилита tracetoute 72
Утилита arp....72
Утилита netstat....72
Утилита TCPdump....73
Литература и ссылки 73

Глава 3. Настройка модемного соединения 75

Начальные установки....75
Настройка модема и последовательного порта 76
Связь с провайдером 77
Схема организации подключения локальной сети 77
Организация связи по коммутируемому соединению....78
Настройка программ....78
Настройка связи с провайдером....78
Команды pppd 80
Настройка diald 83
Создание скрипта соединения: /etc/diald/connect 84
Настройка основной конфигурации: /etc/diald.conf....85
Настройка правил тайм-аутов: /etc/diald/standard.filter 87
Комплексное тестирование....87
Настройка сервера входящих звонков (Dial-in) 88
Настройка mgetty 88
Настройка pppd....89
Настройка Callback-сервера....90
Конфигурация Callback-сервера 90

Конфигурация клиентов 91
Конфигурирование Linux-клиента 91
Конфигурирование клиента MS Windows....92
Литература и ссылки 92

ЧАСТЬ II. СЕТЕВЫЕ СЛУЖБЫ 95

Глава 4. DHCP 97

DHCP-протокол....97
Архитектура и формат сообщений 97
Режимы выдачи IP-адресов....98
Параметры конфигурации (поле *options*) 100
Недостатки DHCP 100
DHCP-сервер 101
Файл dhcpd.conf 101
Файл dhcpd.leases 104
Пример файла dhcpd.conf....105
DHCP-клиент 106
Файл dhclient.conf 106
Файл dhclient.leases 108
Литература и ссылки 109

Глава 5. DNS 110

Настройка сетевых параметров 111
Файл host.conf 111
Файл /etc/hosts....111
Файл /etc/resolf.conf....112
Настройка кэширующего сервера 112
Файл /etc/named.conf....112
Файл /etc/127.0.0 114
Запуск named....115
Настройка полнофункционального DNS-сервера 116
Файл /etc/named.conf....116
Файл /etc/named/ivan.petrov....117
Файл /etc/192.168.0 118
Некоторые тонкости....119
Записи ресурсов (RR) службы DNS....119
Реверсная зона 121
Два сервера DNS....121
Иерархические поддомены 121
Вторичные DNS-серверы 121
Используйте серверы кэширования....121
Инструменты....121
Литература и ссылки 122

Глава 6. Почта 123

Протокол SMTP....124
Протокол POP3....124
Протокол IMAP....125
Формат почтового сообщения....125
Спецификация MIME....126
Поле *MIME-Version*....126
Поле *Content-Type* 127
Поле *Content-Transfer-Encoding*....127

Программное обеспечение 128
Спецификация S/MIME 128
PGP, GPG 128
Программа sendmail 129
Принцип работы 129
Настройка программы 130
Тестирование отправки почты sendmail 131
Тестирование обслуживания по протоколу SMTP 131
Тестирование обслуживания по протоколу POP3 135
Программа Postfix 137
Конфигурационные файлы 138
Литература и ссылки 138

Глава 7. Сетевая информационная система NIS (NIS+) и ее конфигурирование. LDAP 140

NIS 140
Как работает NIS 140
Программа-сервер ypserv 141
NIS+ 141
Как работает NIS+ 142
LDAP 142
Установка LDAP-сервера 143
Настройка LDAP-сервера 143
Формат конфигурационного файла 143
Ключи командной строки 149
База данных LDAP 150
Механизмы баз данных LDAP, объекты и атрибуты 150
Создание и поддержание базы данных 152
Утилиты 154
Slapindex 154
Slapcat 154
Ldapsearch 155
Ldapdelete 155
Ldapmodify 156
Ldapadd 156
Kldap 156
GQ 156
Взаимодействие программ с LDAP 156
Литература и ссылки 158

Глава 8. FTP 159

Протокол FTP 159
Представление данных 159
Тип файла 159
Управление форматом 160
Структура 160
Режим передачи 160
Управляющие команды FTP 161
Ответы на управляющие FTP-команды 161
Управление соединением 163
Программное обеспечение 164
Пакет wu-ftp 164
Команды 164

Конфигурирование сервера ... 166
Файл ftpaccess ... 166
Файл ftpservers ... 172
Файл ftpconversions ... 172
Файл ftpgroups ... 173
Файл ftphosts ... 173
Файл ftpusers ... 173
Параметры запуска программ, входящих в пакет ... 173
Программа ftpd ... 173
Программа ftpwho ... 174
Программа ftpcount ... 174
Программа ftpshut ... 174
Программа ftprestart ... 175
Программа ckconfig ... 175
Формат файла журнала xferlog ... 175
Безопасность ... 176
Литература и ссылки ... 177

Глава 9. NNTP. Сервер новостей INN ... 178

Протокол NNTP ... 178
Основные команды протокола NNTP ... 181
ARTICLE ... 181
BODY ... 181
HEAD ... 181
STAT ... 181
GROUP ggg ... 182
HELP ... 182
IHAVE <message-id> ... 182
LAST ... 182
LIST ... 182
NEWGROUPS date time [GMT] [<distributions>] ... 183
NEWNEWS newsgroups date time [GMT] [<distribution>] ... 183
NEXT ... 183
POST ... 183
QUIT ... 184
SLAVE ... 184
Сервер новостей INN ... 184
Работа пакета INN ... 184
Управляющие сообщения ... 184
Настройка системы INN ... 185
Файл active ... 195
Файлы базы данных и журналы ... 196
Настройка списка получаемых групп новостей ... 197
Журналирование пакета INN ... 200
Программы пакета INN ... 201
Литература и ссылки ... 202

Глава 10. Web-сервер Apache ... 204

Конфигурация ... 205
Используемые обозначения ... 205
Права доступа и свойства объекта ... 206
Общие характеристики сервера ... 208
Виртуальные серверы ... 210

Преобразование адресов ... 211
Преобразование HTTP-заголовков ... 211
Безопасность ... 212
Индекс каталога ... 212
Перекодировка (русификация) ... 213
Файл access.conf ... 216
Файл srm.conf ... 217
Файл httpd.conf ... 217
Настройка виртуальных серверов в файле httpd.conf ... 217
Литература и ссылки ... 219

Глава 11. Proxy-сервер ... 220

Squid ... 221
Протокол ICP ... 221
Cache digest ... 221
Иерархия кэшей ... 222
Алгоритм получения запрошенного объекта ... 222
Конфигурирование пакета Squid ... 222
Сетевые параметры ... 222
Соседи ... 223
Размер кэша ... 224
Имена и размеры файлов ... 224
Параметры внешних программ ... 225
Тонкая настройка кэша ... 226
Время ожидания ... 227
ACL ... 228
Права доступа ... 229
Параметры администрирования ... 229
Параметры для работы в режиме ускорителя HTTP-сервера ... 229
Разное ... 230
Пример конфигурации Squid ... 232
Создание иерархии Proxy-серверов ... 233
Transparent proxy ... 234
Ключи запуска Squid ... 235
Файлы журналов Squid ... 236
Файл access.log ... 236
Файл store.log ... 237
Файл useragent.log ... 238
Нестандартные применения ... 238
Борьба с баннерами ... 238
Разделение внешнего канала ... 239
Обработка статистики ... 240
Программа Squid Cache and Web Utilities (SARG) ... 241
Программа MRTG ... 241
Литература и ссылки ... 241

Глава 12. Синхронизация времени через сеть, настройка временной зоны ... 242

Сетевой протокол времени ... 242
Классы обслуживания ... 243
Обеспечение достоверности данных ... 243
Формат NTP-пакета ... 244
Рекомендуемая конфигурация ... 244
Стандарты ... 245

Сервер xntpd ... 245
Конфигурация сервера ... 245
Класс *symmetric* ... 246
Класс *procedure-call* ... 246
Класс *multicast* ... 246
Общие параметры ... 247
Обеспечение безопасности сервера ... 249
Программы и утилиты, относящиеся к службе точного времени ... 250
ntpdate ... 250
ntpq ... 250
ntptrace ... 250
xntpd ... 251
xntpdc ... 251
Публичные NTP-серверы ... 251
Клиентские программы для синхронизации времени ... 251
UNIX/Linux ... 252
Apple ... 252
Windows ... 252
Литература и ссылки ... 252

ЧАСТЬ III. СЕТЕВЫЕ РЕСУРСЫ. ВЗАИМОДЕЙСТВИЕ С ДРУГИМИ ОПЕРАЦИОННЫМИ СИСТЕМАМИ ... 253

Глава 13. NFS — сетевая файловая система ... 255

Установка и настройка NFS-сервера ... 255
Установка и настройка NFS-клиента ... 256
Опции монтирования ... 257
rsize ... 257
wsize ... 257
soft ... 258
hard ... 258
timeo=n ... 258
retrans=n ... 258
Безопасность NFS ... 258
Безопасность клиента ... 258
Безопасность сервера ... 259
Литература и ссылки ... 259

Глава 14. Сервер Samba для клиентов Windows ... 260

Файл конфигурации smb.conf ... 262
Секция *[global]* ... 267
Секция *[homes]* ... 270
Секция *[comm]* ... 270
Секция *[tmp]* ... 271
Пароли пользователей ... 271
Добавление пользователей Samba ... 272
Принтеры ... 273
Использование ресурсов Samba ... 273
Конфигурирование Samba в качестве первичного контроллера домена ... 275
Утилиты ... 277
SWAT ... 277
Webmin ... 278
Ksamba ... 278
SambaSentinel ... 278
Литература и ссылки ... 278

Глава 15. Mars — клиентам Novell 280

Термины, используемые в тексте 280
Linux и IPX 282
Файлы в каталоге /proc, относящиеся к IPX 282
Linux-утилиты IPX 282
IPX-клиент 283
Настройка сетевого программного обеспечения IPX 283
Проверка конфигурации 283
Монтирование сервера или тома Novell 283
Посылка сообщения пользователю Novell 283
IPX-сервер 284
Пакет Mars_nwe 284
Пакет Lwared 290
IPX-маршрутизатор 291
Настройка Linux как клиента печати сервера Novell 292
Настройка Linux как сервера печати Novell 292
Пользовательские и административные команды ncpfs 292
Команды пользователя 293
Утилиты администрирования 293
Туннелирование IPX через IP 294
Настройка 294
Литература и ссылки 295

ЧАСТЬ IV. НА СЛУЖБЕ 297

Глава 16. Firewall 299

Типы брандмауэров 300
Брандмауэр с фильтрацией пакетов 301
Политика организации брандмауэра 302
Фильтрация сетевых пакетов 303
Фильтрация входящих пакетов 303
Фильтрация исходящих пакетов 306
Защита локальных служб 307
Программа ipchains 307
Опции ipchains 309
Символьные константы 310
Создание правил фильтрации 311
Удаление существующих правил 311
Определение политики по умолчанию 312
Разрешение прохождения пакетов через интерфейс обратной петли 312
Запрет прохождения пакетов с фальсифицированными адресами 313
Фильтрация ICMP-сообщений 315
Сообщения об ошибках и управляющие сообщения 316
Противодействие smurf-атакам 319
Разрешение функционирования служб 319
Запрет доступа с "неблагонадежных" узлов 325
Поддержка обмена в локальной сети 325
Разрешение доступа к внутреннему сетевому интерфейсу брандмауэра 325
Выбор конфигурации для пользующейся доверием локальной сети 325
Организация доступа из локальной сети к брандмауэру бастионного типа 326
Перенаправление трафика 326
Разрешение доступа в Интернет из локальной сети: IP-перенаправление и маскировка 327
Организация демилитаризованной зоны 329
Защита подсетей с помощью брандмауэров 329

Отладка брандмауэра ... 330
Общие рекомендации по отладке брандмауэра ... 330
Отображение списка правил брандмауэра ... 332
Утилиты ... 332
Iptables ... 332
Порядок движения транзитных пакетов ... 334
Порядок движения пакетов для локальной программы ... 335
Порядок движения пакетов от локальной программы ... 336
Таблица mangle ... 336
Таблица nat ... 337
Таблица filter ... 337
Построение правил для iptables ... 337
Команды ipchains ... 338
Критерии проверки пакетов ... 339
Общие критерии ... 340
TCP-критерии ... 341
UDP-критерии ... 342
ICMP-критерии ... 343
Специальные критерии ... 343
Действия и переходы ... 345
Действие ACCEPT ... 346
Действие DNAT ... 346
Действие DROP ... 346
Действие LOG ... 346
Действие MARK ... 347
Действие MASQUERADE ... 347
Действие MIRROR ... 347
Действие QUEUE ... 347
Действие REDIRECT ... 347
Действие REJECT ... 347
Действие RETURN ... 347
Действие SNAT ... 348
Действие TOS ... 348
Действие TTL ... 348
Действие ULOG ... 348
Утилиты iptables ... 348
Iptables-save ... 348
Iptables-restore ... 349
Литература и ссылки ... 349

Глава 17. Сетевые принтеры ... 350

Способы вывода на принтер ... 350
Система печати CUPS ... 351
Программный пакет LPD ... 351
Настройка LPD ... 353
Учет ресурсов ... 354
Настройка сетевого принтера ... 355
Использование принт-сервера ... 355
Печать на Ethernet-принтер ... 357
Литература и ссылки ... 357

Глава 18. Организация шлюза в Интернет для локальной сети ... 359

Начальные установки ... 360
Связь с провайдером ... 360
Схема организации подключения локальной сети ... 361

Организация связи по коммутируемому соединению 361
Настройка программ 361
Настройка связи с провайдером 362
Настройка diald 364
Создание скрипта соединения: /etc/diald/connect 365
Настройка основной конфигурации: /etc/diald.conf 367
Настройка правил тайм-аутов: /etc/diald/standard.filter 368
Комплексное тестирование 368
Организация связи по выделенному каналу 369
Настройка связи с провайдером 369
Комплексное тестирование 370
Защита локальной сети 371
Установка Proxy-сервера 371
Transparent Proxy 371
Борьба с баннерами 372
Разделение внешнего канала (ограничение трафика) 372
Мониторинг загрузки каналов 373
Программа MRTG 373
Конфигурирование MRTG 373
Программа RRDtool 377
Подсчет трафика 377
Литература и ссылки 378

Глава 19. Учет сетевого трафика 380

Простой учет трафика 380
Учет трафика при помощи net-acct 385
Nacctab 385
Nacctpeering 387
Литература и ссылки 388

Глава 20. Виртуальные частные сети 389

Протокол IPSec 390
VPN-сервер FreeS/WAN 391
Ipsec.conf 392
Ipsec.secrets 393
MS Windows NT VPN (PPTP) 394
Linux PPTP-сервер 395
Linux PPTP-клиент 396
Литература и ссылки 396

Глава 21. Бездисковые компьютеры 397

Что такое бездисковый компьютер 397
Преимущества использования бездискового компьютера 397
Недостатки использования бездискового компьютера 398
Области применения 399
Процесс загрузки бездискового компьютера 399
Предварительные действия 400
Установка и настройка программного обеспечения на сервере 401
Linux-клиент 401
Создание загрузочного ПЗУ (загрузочной дискеты) 402
Настройка сервера 403
Конфигурация клиента 404
Windows-клиенты 404
Установка и настройка программного обеспечения на клиенте 405

Создание загрузочного образа дискеты ... 407
Загрузка бездисковой машины ... 407
Оптимизация бездисковой загрузки ... 408
Литература и ссылки ... 411

ЧАСТЬ V. УТИЛИТЫ АДМИНИСТРИРОВАНИЯ И ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ ... 413

Глава 22. Доступ к удаленным компьютерам ... 415

Telnet ... 415
Протокол Telnet ... 415
Команды Telnet ... 417
Программа-клиент telnet ... 418
Программа-сервер telnetd ... 419
Применение Telnet и безопасность ... 419
Семейство r-команд ... 420
Команда *rlogin* ... 420
Команда *rsh* ... 420
Команда *rcp* ... 420
Команда *rsync* ... 420
Команда *rdist* ... 420
Применение r-команд и безопасность ... 421
SSH и OpenSSH ... 421
Принцип работы SSH ... 421
OpenSSH ... 422
Конфигурирование OpenSSH ... 422
Ключи запуска сервера SSH ... 428
Ключи запуска клиента SSH ... 428
Программы, входящие в пакет OpenSSH ... 430
Программа ssh-keygen ... 430
Программа ssh-agent ... 431
Программа ssh-add ... 431
Программа sftp ... 431
Программа scp ... 432
Программа ssh-keyscan ... 433
Литература и ссылки ... 434

Глава 23. Обеспечение безопасности и администрирование сети ... 435

Расширенное управление доступом к файлам ... 435
Установка Linux ACL ... 436
Установка и изменение прав доступа ... 437
Шифрование трафика ... 438
Stunnel ... 439
Установка ... 439
Организация шифрованного туннеля ... 439
Stunnel и приложения, поддерживающие SSL ... 440
Сертификаты ... 440
Утилиты сканирования и защиты сети ... 441
SATAN ... 441
Portsentry ... 442
Установка и настройка ... 442
Запуск ... 444

Сетевая статистика ... 444
NeTraMet ... 444
Ключи запуска NeTraMet ... 445
Ключи запуска NeMaC ... 445
Протоколирование ... 445
Демон syslogd ... 446
Параметры запуска ... 446
Файл конфигурации ... 446
Сетевое протоколирование ... 448
Демон klogd ... 448
Защита системы после взлома ... 449
Rootkit ... 449
Обнаружение rootkit ... 451
Сканирование портов ... 451
Использование RPM ... 451
Сканер для rootkit ... 452
После обнаружения ... 452
LIDS ... 453
Установка ... 453
Конфигурирование LIDS ... 455
Способности ... 455
Правила доступа ... 458
Tripwire ... 459
Portsentry ... 460
LogSentry ... 460
AIDE ... 460
RSBAC ... 460
Security-Enhanced Linux ... 461
Литература и ссылки ... 461

Приложение 1. Литература ... 463

Приложение 2. Ссылки ... 466

Приложение 3. Описание компакт-диска ... 473

Предметный указатель ... 477

Введение

Традиционным элементом практически любой книги является введение. Любой человек, взяв незнакомую книгу в руки, первым делом интересуется тремя вещами: аннотацией, введением и оглавлением книги. Позвольте представить вам введение моей книги.

Благодарности

Хотелось бы сказать о людях, благодаря которым эта книга в конце концов была создана.

Огромное спасибо моей жене Светлане и дочке Наталье за проявленное терпение, поддержку и понимание — мало людей согласятся видеть мужа и отца на протяжении многих месяцев спиной к окружающей действительности и лицом к монитору. Спасибо всем остальным членам моей семьи — без их чуткости и поддержки мне было бы намного тяжелее работать.

Отдельное спасибо Юрию Осьмеркину — это он меня привел в мир Linux и консультировал по множеству вопросов, связанных с материалом книги.

Я благодарен коллективу издательства "БХВ-Петербург" за веру в молодых авторов и терпение в работе с ними. Особо хочется отметить следующих людей: Екатерину Капалыгину, моего редактора, благодаря ее стараниям книга приобрела единый стиль подачи материала; Евгения Рыбакова — за решение общих проблем; и других специалистов, создавших книгу в том виде, в котором читатель увидит ее в магазинах.

Я благодарен сотням и тысячам энтузиастов, плодами работы которых я воспользовался при написании книги, — составителям и переводчикам разнообразной документации, FAQ, How To и различных статей, авторам программ и просто их пользователям.

Почему написана эта книга

Достаточно сложный вопрос. Здесь переплелись и меркантильный интерес, и честолюбие, желание попробовать себя в другой области, попытка побороть свою неуверенность и лень, и не в последнюю очередь — хотелось сделать книгу для наших реалий и нашей специфики. Не секрет, что большинство переводной литературы неадекватно для нашей полунищей действительности. Часто можно встретить несколько "раздражающие" для глаза администратора бюджетной организации советы типа "в качестве маршрутизатора мы рекомендуем использовать устройство фирмы Cisco со следующими параметрами...". Конечно, с точки зрения надежности, простоты в обслуживании и тому подобных вещей такой совет верен. А с точки зрения банального бюджета какой-нибудь государственной конторы — заплатить 4—5 тысяч американских долларов за "железку" размером с кирпич — полный абсурд. Поэтому для наших реалий нужна книга, описывающая построение сетевой и программной инфраструктуры, позволяющей решать большинство типовых задач. Помимо этого, одной из причин для создания книги явилось желание систематизировать и углубить свои собственные знания об операционной системе Linux и ее приложениях.

Для кого написана эта книга

Прежде чем создавать какое-то произведение, автор всегда должен определить своего потенциального читателя. Каким же я его вижу? Это должен быть человек, увлекающийся информационными технологиями, который обладает достаточно приличным багажом знаний в области программного обеспечения (как правило, операционная система Windows), почти наверняка тем или иным образом связанный с администрированием (по крайней мере, как администратор своего собственного персонального компьютера), которому интересно возиться с программным обеспечением и который собирается перейти, или недавно это сделал, к использованию операционной среды Linux. При этом уровень книг серии "для чайников" или "сделай все за 21 день" его заведомо не устраивает, поскольку ему необходимо четко представлять себе возможности операционной системы, ее структуру, решаемые с ее помощью прикладные задачи, наиболее популярное программное обеспечение, его установка, настройка и использование.

Вот так я представляю себе читателя книги.

Структура книги

Книга разбита на пять частей плюс приложения. Рассмотрим, что в них описывается и для кого они предназначены.

Часть I представляет интерес для новичков в мире Linux. В ней содержится обзор протоколов семейства TCP/IP, процесс настройки и отладки сетевых интерфейсов, а также настройка модемного соединения. Этот раздел является вводным (базисным), поскольку дальнейшее изложение материала подразумевает знание специфики протоколов TCP/IP и настроенной сети. Он будет интересен в первую очередь новичкам и "продвинутым" пользователям, поскольку администраторы со стажем должны знать данную тему "на зубок".

Часть II представляет интерес как для новичков, так и для опытных пользователей операционной системы Linux, поскольку именно здесь рассматриваются вопросы конфигурирования сетевых служб. В этой части описывается конфигурирование DHCP, DNS, почтового сервера, службы LDAP, FTP, NNTP, Apache, Proxy-сервера и NTP-сервера. Именно эта часть позволит вам создать полнофункциональный сервер, способный выполнить около 90% задач типичного сервера небольшой организации. Вторую часть я старался сделать доступной для понимания начинающему пользователю. Она содержит большой объем информации в достаточно сжатом виде и требует размышлений и экспериментов от читателя.

В *части III* я опять возвращаюсь к настройке сетевых сервисов. Поскольку мы за плюрализм и демократию, наша сеть не является гомогенной средой и волей-неволей приходится взаимодействовать с различными операционными системами. В этой части мы ознакомимся с NFS (сетевой файловой системой UNIX), научимся предоставлять и получать доступ к каталогам операционных систем Windows и Novell.

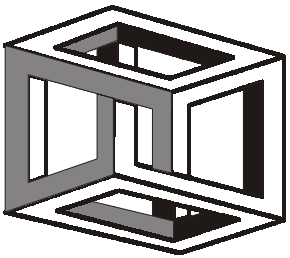
Предыдущие части книги были подготовительным этапом для *части IV*. Она предназначена больше для опытных пользователей, т. к. я хотел, чтобы моя книга служила вам верой и правдой в качестве справочного пособия долгое время, и вы периодически возвращались к ней для решения специфических задач, возникающих в вашей работе. Здесь вы найдете описание основных приложений, используемых для организации НОРМАЛЬНОГО функционирования сети организации, подключенной к Интернету. В этой части рассмотрена защита сети от нежелательного воздействия, организация виртуальных частных сетей, учета сетевого трафика, настройка сетевых принтеров, изготовление бездисковых компьютеров и организация шлюза в Интернете.

Часть V посвящена администрированию системы. Здесь рассмотрен удаленный безопасный доступ к хостам, а также утилиты для администрирования и мониторинга сети.

В *приложениях* находится список рекомендуемой литературы, небольшая коллекция ссылок, тем или иным образом касающихся Linux и программ для этой операционной системы, а также список наиболее часто применяемых сетевых портов и программ, их использующих.

Как со мной связаться

Те читатели, которые хотят внести свои предложения или уточнения по содержанию данной книги, поделиться интересными идеями, темами и т. п., могут воспользоваться электронным адресом **alst@farlep.net**. Я постараюсь ответить на все письма. Также можно воспользоваться моим сайтом **www.alst.od.ua**.



ЧАСТЬ I

Сетевые протоколы и конфигурирование

Эта часть является вводной. Как в хорошем детективе, прежде чем приступать к поиску преступника, нужно осмотреть место происшествия. Продолжая аналогию — прежде чем углубляться в дебри специфического программного обеспечения, необходимо узнать фундамент, на котором оно стоит, те общие моменты, которые используются большинством программ.

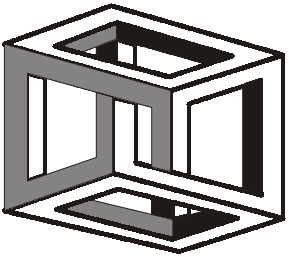
Данная часть интересна в первую очередь новичкам и опытным пользователям, поскольку администраторы со стажем должны хорошо знать эту тему. В ней содержится обзор протоколов семейства TCP/IP, процесс настройки и отладки сетевых интерфейсов, настройка модемного соединения, поскольку дальнейшее изложение материала подразумевает знание специфики протоколов TCP/IP и настроенной сети.

Глава 1. Сетевые протоколы

Глава 2. Настройка сети TCP/IP

Глава 3. Настройка модемного соединения

ГЛАВА 1



Сетевые протоколы

В данной главе будут рассмотрены базовые понятия, сетевые модели и протоколы, используемые в сетях. На фундаменте, заложенном этой главой, выстроена вся книга, поэтому рекомендую начинающим ознакомиться с ней, а более опытным полистать, освежить свои знания.

Модели сетевых взаимодействий

Как и любая сложная система, сеть опирается на стандарты, без которых невозможно ее нормальное функционирование. За последние двадцать лет было создано множество концепций сетевых взаимодействий, однако наибольшее распространение получили всего две:

- модель взаимодействия открытых систем (OSI);
- модель сетевого взаимодействия TCP/IP.

Терминология

Для облегчения понимания содержимого этой главы, приведем основные термины (табл. 1.1).

Таблица 1.1. *Базовые сетевые термины*

Термин	Определение
Датаграмма	Пакет данных. Обозначает единицу информации при сетевом обмене
DNS (Domain Name Service, служба доменных имен)	Специально выделенные компьютеры, которые производят поиск соответствия символического имени хоста и цифрового адреса хоста

Таблица 1.1 *(продолжение)*

Термин	Определение
Интернет	Глобальная компьютерная сеть, основанная на семействе протоколов TCP/IP
FTP (File Transfer Protocol, протокол передачи файлов)	Протокол используется для приема и передачи файлов между двумя компьютерами
IP (Internet Protocol, протокол Интернет)	Основа семейства протоколов TCP/IP. Практически любой протокол из этого семейства базируется на протоколе IP
ICMP (Internet Control Message Protocol, протокол управляющих сообщений в стеке протоколов IP)	Используется для передачи управляющих сообщений протокола IP
NFS (Network File System, сетевая файловая система)	Система виртуальных дисков, позволяющая клиентским компьютерам использовать каталоги сервера в качестве диска
NIC (Network Information Center, сетевой информационный центр)	Организация, которая отвечает за администрирование и раздачу сетевых адресов и имен
Узел (Node, Host)	Компьютер в сети. Название применимо как к клиенту, так и к серверу
OSI (Open System Interconnection, взаимодействие открытых систем)	Модель взаимодействия открытых систем
RFC (Request For Comments, запрос для пояснений)	Стандарты протоколов Интернета и их взаимодействия
RIP (Routing Information Protocol, протокол маршрутизации информации)	Протокол, используемый для обмена информацией между маршрутизаторами
SMTP (Simple Mail Transfer Protocol, простой протокол передачи электронной почты)	Используется для обмена электронной почтой
SNMP (Simple Network Management Protocol, простой протокол управления сетью)	Используется для управления сетевыми устройствами
TCP (Transmission Control Protocol, протокол управления передачей)	Используется для надежной передачи данных
Telnet	Протокол, осуществляющий удаленное сетевое подключение к компьютеру, эмулирующее терминал

Таблица 1.1 *(окончание)*

Термин	Определение
UDP (User Datagram Protocol, протокол пользовательских датаграмм)	Используется для обмена блоками информации без установки соединения

Модель взаимодействия открытых систем (OSI)

Еще в 1983 году Международная организация по стандартизации (International Organization for Standardization, ISO) разработала стандарт взаимодействия открытых систем (Open System Interconnection, OSI). В результате получилась семиуровневая модель:

1. Физический уровень (Physical Level).
2. Уровень данных (Data Link Level).
3. Сетевой уровень (Network Level).
4. Транспортный уровень (Transport Level).
5. Уровень сессии (Session Level).
6. Уровень представления (Presentation Level).
7. Уровень приложения (Application Level).

Первый уровень самый "приземленный", последующие — все более и более абстрагируются от особенностей физической среды передачи информации.

Каждый уровень модели OSI решает свои задачи, использует сервисы, предоставляемые предыдущим уровнем и, в свою очередь, предоставляет сервисы следующему уровню. Согласно этой модели, уровни не могут "перескакивать" через соседей, например, транспортный уровень не может непосредственно пользоваться сервисом физического уровня, он обязан пройти по цепочке: Сетевой уровень → Уровень данных → Физический уровень. В табл. 1.2 приведено описание уровней сетевой модели OSI.

Таблица 1.2. *Уровни сетевой модели OSI*

Уровень	Название	Описание
1	Физический уровень	Отвечает за физическое подключение компьютера к сети. Определяет уровни напряжения, параметры кабеля, разъемы, распайку проводов и т. п.
2	Уровень данных	Физически подготавливает данные для передачи (разбивая их на кадры определенной структуры) и преобразует обратно во время приема (восстанавливая из кадров)

Таблица 1.2 *(окончание)*

Уровень	Название	Описание
3	Сетевой уровень	Маршрутизирует данные в сети
4	Транспортный уровень	Обеспечивает последовательность и целостность передачи данных
5	Уровень сессии	Устанавливает и завершает коммуникационные сессии
6	Уровень представления	Выполняет преобразование данных и обеспечивает передачу данных в универсальном формате
7	Уровень приложения	Осуществляет интерфейс между приложением и процессом сетевого взаимодействия

На каждом уровне блоки информации имеют собственное название (табл. 1.3).

Таблица 1.3. *Название блока информации в модели*

Уровень	Название уровня	Название блока информации
1	Физический уровень	Бит
2	Уровень данных	Кадр (пакет)
3	Сетевой уровень	Датаграмма
4	Транспортный уровень	Сегмент
5, 6, 7	Уровень приложения	Сообщение

Несмотря на то, что OSI является международным стандартом и на его основе правительство США выпустило спецификации GOSIP (Government Open Systems Interconnection Profile, Государственный регламент взаимодействия открытых систем), у производителей программного обеспечения стандарт OSI широкой поддержки не получил. Это объясняется несколькими причинами:

- волокита в принятии стандарта;
- его "оторванность" от реалий;
- наличие большого числа уровней трудно для реализации и приводит к потере производительности;
- широчайшее распространение протокола TCP/IP, и нежелание потребителей отказываться от него.

В результате, спецификации OSI сегодня — это, в основном, страницы в учебнике, в реальной жизни они не применяются.

Модель сетевого взаимодействия TCP/IP

Архитектура семейства протоколов TCP/IP (Transmission Control Protocol/ Internet Protocol, протокол управления передачей/интернет-протокол) основана на представлении, что коммуникационная инфраструктура содержит три вида объектов: процессы, хосты и сети.

Основываясь на этих трех объектах, разработчики выбрали четырехуровневую модель:

1. Уровень сетевого интерфейса (Network Interface Layer).
2. Уровень межсетевого интерфейса — Интернета (Internet Layer).
3. Транспортный уровень (Host-to-Host Layer).
4. Уровень приложений/процессов (Application/Process Layer).

Сопоставление сетевых моделей OSI и TCP/IP

Нетрудно заметить, что модель TCP/IP отличается от модели OSI. В табл. 1.4 показано соответствие модели TCP/IP и модели OSI.

Таблица 1.4. *Соответствие модели TCP/IP и модели OSI*

TCP/IP	OSI
Уровень приложений	Уровень приложений
	Уровень представления
	Уровень сеанса
Транспортный уровень	Транспортный уровень
Межсетевой уровень (Интернет)	Сетевой уровень
Уровень сетевого интерфейса	Уровень канала данных
	Физический уровень

Как видно из таблицы, уровень сетевого интерфейса модели TCP/IP соответствует сразу двум уровням модели OSI, а уровень приложений модели TCP/IP — трем уровням модели OSI.

Сетевые протоколы

В данном разделе мы рассмотрим различные сетевые протоколы, используемые в современной компьютерной индустрии. Пожалуй, это основная часть сетевых моделей (аппаратная часть все-таки не настолько важна для функционирования сети). Также здесь будут рассмотрены протоколы транспортного уровня, на которые опираются протоколы уровня приложений.

Семейство протоколов TCP/IP

Существует несколько протоколов вида TCP/IP. Их можно объединить в семейство. Перечислим его содержимое:

- межсетевой протокол (Internet Protocol — IP, протокол Интернета) соответствует уровню интернет-модели TCP/IP. Отвечает за передачу данных с одного хоста на другой;
- межсетевой протокол управления сообщениями (Internet Control Message Protocol, ICMP) отвечает за низкоуровневую поддержку протокола IP, включая подтверждение получения сообщения, генерирование сообщений об ошибках и многое другое;
- протокол преобразования адресов (Address Resolution Protocol, ARP) выполняет преобразование логических сетевых адресов в аппаратные MAC-адреса (Media Access Control, управление средой доступа). Соответствует уровню сетевого интерфейса;
- реверсный протокол преобразования адресов (Reverse Address Resolution Protocol, RARP) выполняет преобразование аппаратных MAC-адресов в логические сетевые адреса. Соответствует уровню сетевого интерфейса;
- протокол пользовательских датаграмм (User Datagram Protocol, UDP) обеспечивает пересылку данных без проверки с помощью протокола IP;
- протокол управления передачей (Transmission Control Protocol, TCP) обеспечивает пересылку данных (с созданием сессии и проверкой передачи данных) с помощью протокола IP;
- множество протоколов уровня приложений (FTP, Telnet, IMAP, SMTP и др.).

Протоколы семейства TCP/IP можно представить в виде схемы, которая отображена в табл. 1.5.

Таблица 1.5. *Схема семейства протоколов TCP/IP*

Уровень приложений	FTP	SMTP	NFS	SNMP
Транспортный уровень	TCP		UDP	
Межсетевой уровень (Интернет)	IP		ARP/RARP	ICMP
Уровень сетевого интерфейса	Ethernet, FDDI, ATM			
	Витая пара, коаксиальный кабель, оптический кабель и т. п.			

Протоколы межсетевого уровня (Интернет)

Протоколы межсетевого уровня (Интернет) являются базовыми в семействе протоколов TCP/IP. Перечислим их названия: TCP/IP, ARP/RARP и ICMP.

Протокол IP

Первоначальный стандарт IP разработан в конце семидесятых годов и не был рассчитан на огромное количество хостов, характерное для современного Интернета. Поэтому в настоящее время утвержден новый стандарт IP (в литературе часто старый стандарт встречается как IPv4, а новый — как IPv6). Однако массового применения он пока не нашел из-за огромного количества программных и аппаратных средств, не способных работать с IPv6, поэтому в дальнейшем содержимое книги, в основном, преломляется через призму протокола IPv4.

Формат пакета IPv4

Пакет IP состоит из заголовка и поля данных. *Заголовок* пакета имеет следующие поля:

- поле *Номер версии* (VERS) указывает версию протокола IP. Сейчас повсеместно используется версия 4 и готовится переход на версию 6;
- поле *Длина заголовка* (HLEN) пакета IP. Занимает 4 бита и указывает значение длины заголовка, измеренное в 32-битовых словах. Обычно заголовок имеет длину в 20 байт (пять 32-битовых слов), но при увеличении объема служебной информации она может быть увеличена за счет использования дополнительных байт в поле *Резерв* (IP OPTIONS);
- поле *Тип сервиса* (SERVICE TYPE) занимает 1 байт и задает приоритетность пакета и вид критерия выбора маршрута. Первые три бита этого поля образуют подполе приоритета пакета (PRECEDENCE). Приоритет может иметь значения от 0 (нормальный пакет) до 7 (пакет управляющей информации). Поле *Тип сервиса* содержит также три бита, определяющие критерий выбора маршрута. Установленный бит D (delay) говорит о том, что маршрут должен выбираться для минимизации задержки доставки данного пакета, бит T — для максимизации пропускной способности, а бит R — для максимизации надежности доставки;
- поле *Общая длина* (TOTAL LENGTH) занимает 2 байта и указывает общую длину пакета с учетом заголовка и поля данных;
- поле *Идентификатор пакета* (IDENTIFICATION) занимает 2 байта и используется для распознавания пакетов, образовавшихся путем фрагментации исходного пакета. Все фрагменты должны иметь одинаковое значение этого поля;

- поле *Флаги* (FLAGS) занимает 3 бита, оно указывает на возможность фрагментации пакета (установленный бит Do not Fragment, DF — запрещает маршрутизатору фрагментировать данный пакет), а также на то, является ли данный пакет промежуточным или последним фрагментом исходного пакета (установленный бит More Fragments, MF — говорит о том, что пакет переносит промежуточный фрагмент);
- поле *Смещение фрагмента* (FRAGMENT OFFSET) занимает 13 бит, оно используется для указания в байтах смещения поля данных этого пакета от начала общего поля данных исходного пакета, подвергнутого фрагментации. Используется при сборке/разборке фрагментов пакетов при передачах их между сетями с различными величинами максимальной длины пакета;
- поле *Время жизни* (TIME TO LIVE) занимает 1 байт и указывает предельный срок, в течение которого пакет может перемещаться по сети. Время жизни данного пакета измеряется в секундах и задается источником передачи средствами протокола IP. На шлюзах и в других узлах сети по истечении каждой секунды из текущего времени жизни вычитается единица, единица вычитается также при каждой транзитной передаче (даже если не прошла секунда). По истечении времени жизни пакет аннулируется;
- поле *Идентификатор протокола верхнего уровня* (PROTOCOL) занимает 1 байт и указывает, какому протоколу верхнего уровня принадлежит пакет (например, это могут быть протоколы TCP, UDP или RIP);
- поле *Контрольная сумма* (HEADER CHECKSUM) занимает 2 байта, она рассчитывается по всему заголовку;
- поля *Адрес источника* (SOURCE IP ADDRESS) и *Адрес назначения* (DESTINATION IP ADDRESS) имеют одинаковую длину (32 бита) и структуру;
- поле *Резерв* (IP OPTIONS) является необязательным и используется обычно только при отладке сети. Это поле состоит из нескольких подполей, каждое из которых может быть одного из восьми предопределенных типов. Так как число подполей может быть произвольным, то в конце поля *Резерв* должно быть добавлено несколько байт для выравнивания заголовка пакета по 32-битовой границе.

Максимальная длина *поля данных* пакета ограничена разрядностью поля, определяющего эту величину, и составляет 65 535 байт, однако при передаче по сетям различного типа длина пакета выбирается с учетом максимальной длины пакета протокола нижнего уровня, несущего IP-пакеты. В большинстве типов локальных и глобальных сетей определяется такое понятие, как максимальный размер поля данных кадра, в который должен разместить свой пакет протокол IP. Эту величину обычно называют максимальной единицей транспортировки (Maximum Transfer Unit, MTU). К при-

меру, сети Ethernet имеют значение MTU, равное 1500 байт, сети FDDI — 4096 байт.

IP-маршрутизаторы не собирают фрагменты пакетов в более крупные пакеты, даже если на пути встречается сеть, допускающая такое укрупнение. Это связано с тем, что отдельные фрагменты сообщения могут перемещаться в интерсети по различным маршрутам.

Когда пришел первый фрагмент пакета, узел назначения запускает таймер, который определяет максимально допустимое время ожидания прихода остальных фрагментов этого пакета. Если время истекает раньше прибытия последнего фрагмента, то все полученные к этому моменту фрагменты пакета отбрасываются, а в узел, пославший исходный пакет, с помощью протокола ICMP направляется сообщение об ошибке.

Протокол IPv6

Основные причины, из-за которых разрабатывался IPv6:

- протокол IPv4 создали в конце 70-х годов, и вполне логично, что он плохо учитывает особенности современной инфраструктуры и нагрузок на сеть;
- появление приложений, использующих Интернет для передачи данных в реальном времени (звук, видео). Эти приложения чувствительны к задержкам передачи пакетов. Их особенностью является передача очень больших объемов информации. А в IPv4 не предусмотрено специального механизма резервирования полосы пропускания или механизма приоритетов;
- бурное расширение сети Интернет. Наиболее очевидным следствием такого развития стало почти полное истощение адресного пространства Интернета, определяемого полем адреса IP в четыре байта. Конечно, были разработаны механизмы компенсации нехватки адресов, однако это кардинально не решило проблему.

Основное предложение по модернизации протокола IP было разработано группой IETF (Internet Engineering Task Force, группа решения задач Интернета). Согласно предложенному, в протоколе IPv6 остаются неизменными основные принципы IPv4, такие как: датаграммный метод работы, фрагментация пакетов, разрешение отправителю задавать максимальное число хопов (хоп — количество пересылок пакета от одного сетевого интерфейса к другому, иногда называется временем жизни пакета) для своих пакетов. Однако в деталях реализации протокола IPv6 имеются существенные отличия от IPv4. Эти отличия коротко можно описать следующим образом:

- использование более длинных адресов. Новый размер адреса — наиболее заметное отличие IPv6 от IPv4. Версия 6 использует 128-битовые адреса (16 байт);

- гибкий формат заголовка. Вместо заголовка с фиксированными полями фиксированного размера (за исключением поля Резерв), IPv6 использует базовый заголовок фиксированного формата плюс набор необязательных заголовков различного формата;
- поддержка резервирования пропускной способности;
- поддержка расширяемости протокола. Это одно из наиболее значительных изменений в подходе к построению протокола — от полностью детализированного описания протокола к протоколу, который разрешает поддержку дополнительных функций.

Адресация в IPv6

Адреса в IPv6 имеют длину 128 бит или 16 байт. Версия 6 обобщает специальные типы адресов версии 4 в следующих типах адресов:

- Unicast — индивидуальный адрес. Определяет отдельный узел — компьютер или порт маршрутизатора. Пакет должен быть доставлен узлу по кратчайшему маршруту;
- Cluster — адрес кластера. Обозначает группу узлов, которые имеют общий адресный префикс (например, присоединенных к одной физической сети). Пакет должен быть маршрутизирован группе узлов по кратчайшему пути, а затем доставлен только одному из членов группы (например, ближайшему узлу);
- Multicast — адрес набора узлов, возможно в различных физических сетях. Копии пакета должны быть доставлены каждому узлу набора, используя аппаратные возможности групповой или широковещательной доставки, если это осуществимо.

Как и в версии IPv4, адреса в версии IPv6 делятся на классы, в зависимости от значения нескольких старших битов адреса.

Большая часть классов зарезервирована для будущего применения. Наиболее интересным для практического использования является класс, предназначенный для провайдеров услуг Интернета, названный Provider-Assigned Unicast.

Для обеспечения совместимости со схемой адресации версии IPv4 в версии IPv6 есть класс адресов, имеющих 0000 0000 в старших битах адреса. Младшие 4 байта адреса этого класса должны содержать адрес IPv4. Маршрутизаторы, поддерживающие обе версии адресов, должны обеспечивать трансляцию при передаче пакета из сети, поддерживающей адресацию IPv4, в сеть, поддерживающую адресацию IPv6, и наоборот.

Сетевые пакеты

Как уже упоминалось, информация по сети передается определенными порциями — пакетами. Причем, на каждом уровне пакет имеет свой размер

и структуру. В результате в пакет нижнего уровня вкладывается пакет следующего уровня и т. д. Так же понятно, что чем более высокого уровня пакет, тем меньше информации он может содержать в себе. Размеры пакетов ограничиваются как особенностями аппаратуры, так и требованиями протоколов.

Маршрутизация пакетов

Маршрутизация — механизм передачи пакетов между сетями. При маршрутизации пакетов решается задача, как за наименьшее время, по кратчайшему пути, с минимальной стоимостью доставить пакет. Как правило, в совокупности решить эту задачу не представляется возможным. Поэтому протоколы маршрутизации пакетов должны иметь возможность задавать различные правила и стратегии маршрутизации. К примеру, доставить пакет с максимальной скоростью или с минимальной стоимостью.

Протоколы маршрутизации

Протоколы маршрутизации разделяются на протокол внутреннего шлюза (Interior Gateway Protocol, IGP) и протокол внешнего шлюза (Exterior Gateway Protocol, EGP).

Протокол внутреннего шлюза управляет маршрутизацией в пределах сети или группы сетей одного владельца, носящей название "автономная система". Внутри автономных систем имеется только список сетей, входящих в автономную систему, и известны точки взаимодействия с внешним миром.

Протокол внешнего шлюза отвечает за маршрутизацию между автономными системами.

На сегодняшний день широко используются следующие протоколы маршрутизации:

- RIP (Routing Information Protocol) — протокол данных маршрутизации. Устаревший протокол. Тем не менее, достаточно широко распространен благодаря утилите routed, которая является стандартной программой для операционных систем UNIX-семейства;
- OSPF (Open Shortest Path First) — протокол выбора кратчайшего пути. Протокол промышленного уровня. Рассчитан на крупные сети со сложной топологией. Более гибок, чем протокол RIP, однако по сравнению с ним сложнее в администрировании;
- IGRP (Interior Gateway Routing Protocol) — протокол маршрутизации внутреннего шлюза. Используется маршрутизаторами CISCO. По всей видимости, скоро сойдет со сцены;
- EGP (Exterior Gateway Protocol) — протокол внешнего шлюза. Старый протокол времен зарождения Интернета. Практически вытеснен протоколом BGP;

- BGP (Border Gateway Protocol) — протокол граничного шлюза. В отличие от EGP, поддерживает сложную топологию сети. Имеет возможность широкой настройки стратегии маршрутизации;
- DVMRP (Vector Multicast Routing Protocol) — протокол групповой маршрутизации по вектору расстояния;

RIP, OSPF и IGRP — внутренние протоколы; EGP и BGP — внешние протоколы.

Адресация в TCP/IP

Каждый компьютер в сети IP имеет адреса трех уровней:

1. *Локальный адрес узла* определяется технологией (например, Ethernet), с помощью которой построена отдельная сеть, имеющая в своем составе данный узел. Для узлов, входящих в локальные сети, — это MAC-адрес сетевого адаптера. MAC-адреса назначаются производителями оборудования и являются (теоретически) уникальными адресами, т. к. управляются централизованно, однако большинство производителей Ethernet-карт предоставляют утилиту для переназначения MAC-адреса. Для всех существующих технологий локальных сетей MAC-адрес имеет 6-байтовый формат: старшие 3 байта — идентификатор фирмы производителя, а младшие 3 байта назначаются уникальным образом самим производителем.
2. *IP-адрес* состоит из 4 байт (стандарт IPv4) или 16 байт (стандарт IPv6). Этот адрес используется на сетевом уровне. Он назначается администратором во время конфигурирования сети. IP-адрес состоит из двух частей: номера сети и номера узла. Номер сети может быть выбран администратором произвольно либо назначен по рекомендации специального подразделения Интернета (Network Information Center, NIC), если сеть должна работать как составная часть Интернета.
3. *Символьный идентификатор-имя*, например, tosser.mail.ru. Этот адрес назначается администратором и состоит из нескольких частей, например, имени машины, имени фирмы, имени домена. Такой идентификатор-имя используется на прикладном уровне, например, в протоколе FTP.

IP-адрес состоит из двух частей: сетевой части и адреса хоста. На основании сетевой части адреса принимается решение о сетевой маршрутизации. Адрес хоста однозначно определяет сетевое устройство, которое, в большинстве случаев, совпадает с хостом (здесь не обойтись без исключений — некоторые компьютеры имеют несколько IP-адресов). Стандартно IP-адреса записываются как десятичные числа (по одному на каждый байт адреса), разделенные точками. К примеру, 192.168.74.2. Однако не все сетевые адреса могут использоваться для назначения их компьютерам. Исключениями являются адреса 0.0.0.0, 127.0.0.1, 255.255.255.255 и некоторые другие. Существует несколько классов сетевых адресов (табл. 1.6).

Таблица 1.6. *Распределение сетевых адресов по классам сетей*

Класс	Первый байт	Формат	Комментарии
A	1—126	C.M.M.M	Очень крупные сети, как правило — корпорации или большие государственные учреждения
B	128—191	C.C.M.M	Крупные сети — крупные фирмы, большие интернет-провайдеры
C	192—223	C.C.C.M	Обычная сеть на 254 компьютера
D	224—239	—	Как правило, подсети, выдаваемые провайдерами клиентам
E	240—254	—	Экспериментальные адреса

В колонке "Формат" буквы C обозначают сетевую часть адреса, а буквы M — его компьютерную часть.

Выделением IP-адресов занимается служба регистрации информационного центра InterNIC, но если необходимо получить 4—5 IP-адресов, то их вполне может предоставить любой интернет-провайдер. Однако не все адреса предназначены для доступа из Интернета. Существует группа адресов, используемых только в локальных сетях:

- 10.0.0.0 — 10.255.255.255
- 172.16.0.0 — 172.31.255.255
- 192.168.0.0 — 192.168.255.255

Как можно видеть, это адреса классов A, B и C соответственно.

Несколько IP-адресов имеют специальное значение:

- адрес, в котором сетевая часть содержит нули, соответствует хосту в локальной сети. Например, 0.0.0.145 соответствует рабочей станции 145 в локальной сети, а адрес 0.0.0.0 — текущему хосту;
- сеть с адресом 127.X.X.X — является фиктивной сетью, не имеющей никаких аппаратных сетевых интерфейсов и состоящей только из локального компьютера. Адрес 127.0.0.1 всегда обозначает текущую машину и имеет символическое имя localhost;
- адрес, содержащий в какой-либо части число 255, обозначает широковещательный адрес. Например, пакет, посланный по адресу 192.168.3.255, будет отослан всем компьютерам в сети 192.168.3, а пакет, посланный по адресу 255.255.255.255, отправится по *всем* компьютерам Интернета.

Символьные имена Интернета имеют следующую структуру:

Имя_компьютера.домен3уровня.домен2уровня.домен1уровня.

Например: **www.rambler.ru**, **www.yahoo.com**, **www.alst.odessa.ua**.

Домены первого уровня стандартизированы и обычно состоят из двух или трех букв латинского алфавита. В последнее время вводятся домены первого уровня, состоящие из более чем трех букв, но пока массового распространения они не получили. Как правило, домен первого уровня может иметь имя типа com, org, net, mil или двухсимвольного названия страны, за которой закреплен домен: ru — Россия, ua — Украина, uk — Великобритания. Относительно имени домена второго уровня строгих правил нет. Для доменов первого уровня типа com, домен второго уровня обычно носит имя компании, фирмы или торговой марки. Для домена страны правила именования несколько другие. В частности, для России имя домена второго уровня определяется покупателем — к примеру, exler.ru, а для Украины имя домена второго уровня — это либо название областного центра (odessa.ua), либо имя типа com, org, net, mil. Похожая ситуация наблюдается и в других странах: Швеция, Франция, Германия имена доменов второго уровня жестко не закрепляют, а Великобритания, Тайвань, Япония — закрепляют.

Протокол адресации ARP/RARP

Несмотря на то, что адресация IP-пакетов осуществляется при помощи IP-адресов, при передаче данных с компьютера на компьютер необходимо использовать аппаратные MAC-адреса (конечно, кроме тех случаев, когда используется соединение типа "точка-точка"). Для определения соответствия аппаратных MAC-адресов IP-адресам служит протокол ARP (Address Resolution Protocol, протокол преобразования адресов). Он применяется в сетях любых типов, использующих широковещательный режим. ARP можно применять только в пределах одной сети. Однако это не мешает передавать пакет через несколько сетей, просто при прохождении пакетом маршрутизатора, он определяет новый MAC-адрес приемника. Каждый компьютер в сети создает таблицу ARP, которая содержит последние запросы.

Иногда аппаратные адреса необходимо транслировать в IP-адреса. Для этого используется протокол RARP (Reverse Address Resolution Protocol, обратный протокол преобразования адресов).

Протокол ICMP

Протокол ICMP (Internet Control Message Protocol, межсетевой протокол управления сообщениями) отвечает за низкоуровневую поддержку протокола IP, включая подтверждение получения сообщения, генерацию сообщения об ошибках и многое другое. В какой-то мере может использоваться для маршрутизации пакетов.

ICMP-сообщения посылаются с помощью стандартного IP-заголовка. Первый октет в поле данных датаграммы — это поле типа ICMP-сообщения. Значение этого поля определяет формат всех остальных данных в датаграмме. Затем идет октет, содержащий код, поясняющий сообщение, и двухбайтовая контрольная сумма.

В табл. 1.7 приведены типы и коды ICMP-сообщений.

Таблица 1.7. *Типы и коды ICMP-сообщений*

Тип	Код	Описание кода	Описание типа
0	0		Ответ на эхо-сообщение
3	0	Невозможно передать датаграмму на локальную сеть, где находится адресат	Сообщение о проблемах при передаче пакетов Шлюз может послать сообщения с кодами 0, 1, 4 и 5. Хост может послать сообщения с кодами 2 и 3
	1	Невозможно передать датаграмму на хост, являющийся адресатом	
	2	Нельзя воспользоваться указанным протоколом	
	3	Нельзя передать данные на указанный порт	
	4	Для передачи датаграммы по сети требуется фрагментация, но выставлен флаг DF (запрет фрагментации пакетов)	
	5	Сбой в маршрутизации при отправлении	
4	0		Сообщение для приостановки отправителя. Шлюз может удалить датаграммы, если у него нет места в буфере для постановки этих датаграмм в очередь на отправление по маршруту следования к адресату. Если шлюз удаляет датаграмму, то он должен послать сообщение для приостановки хосту, отправившему данную датаграмму. Сообщение о приостановке может послать также сам адресат, если датаграммы приходят слишком быстро, чтобы успеть их обработать. Сообщение о приостановке является запросом для хоста уменьшить скорость посылки данных на этот адрес. Шлюз или хост может посылать сообщение о приостановке еще до достижения предельной пропускной способности и не ждать, пока этот предел будет пройден. И шлюз и хост могут отправить сообщение с кодом 0

Таблица 1.7 *(продолжение)*

Тип	Код	Описание кода	Описание типа
5	0	Переадресация датаграмм для сети	Сообщение о переадресации
	1	Переадресация датаграмм для хоста	
	2	Переадресация датаграмм для типа услуг и сети	
	3	Переадресация датаграмм для типа услуг и хоста	
8	0		Эхо-сообщение и сообщение в ответ на эхо. Данные из эхо-сообщения должны быть переданы в ответ на это сообщение. Идентификатор и номер очереди может использоваться отправителем эхо-сообщения с целью идентификации приходящих пакетов. Компьютер, отозвавшийся на это сообщение, возвращает в своем ответе те же значения для идентификатора и номера очереди, что были в исходном эхо-сообщении. Как шлюз, так и хост могут возвращать сообщение с кодом 0
11	0	При передаче превышено время жизни	Сообщение о превышении контрольного времени. Шлюз может послать сообщение с кодом 0, а хост — с кодом 1
	1	Превышено контрольное время при сборке фрагментов датаграммы	
12	0	Указатель показывает ошибку	Сообщение о проблемах с параметром. Если шлюз или хост, обрабатывающий датаграмму, обнаруживает проблему с обработкой параметров заголовка, и это не позволяет завершить ее обработку, то он должен удалить датаграмму. Одной из причин этого могут быть неправильные аргументы в опции. Указатель определяет октет в заголовке исходной датаграммы, где была обнаружена ошибка. Код 0 сообщения может приходить как от шлюза, так и от хоста

Таблица 1.7 *(окончание)*

Тип	Код	Описание кода	Описание типа
13	0		Сообщение со штампом времени. Данные из сообщения возвращаются вместе с ответом, при этом в них добавляется еще один штамп времени. *Штамп времени* — это 32 бита, где записано время в миллисекундах, прошедшее после полуночи по единому времени (Union Time, UT). *Штамп времени отправления* — это время, которое отправитель фиксировал последний раз перед посылкой сообщения. *Штамп времени получения* — это время, когда исходное сообщение впервые увидел получатель первоначального сообщения. *Штамп времени передачи* — это время, которое фиксировал в последний раз компьютер, отправляющий ответное сообщение. И шлюз и хост могут возвращать сообщения с кодом 0
14	0		Сообщение с ответом на штамп времени
15	0		Запрос информации. Данное сообщение может быть послано, когда в IP-заголовке в полях отправителя и получателя записаны нули. В ответ должен быть послан IP-модуль с полностью заданными адресами. Данное сообщение является способом, с помощью которого хост может определить номер сети, куда он подключен. И хост и шлюз могут возвращать сообщения с кодом 0
16	0		Ответное сообщение с информацией

Протоколы транспортного уровня

Протоколы транспортного уровня базируются на протоколе IP. Существуют два протокола транспортного уровня: TCP и UDP. Они обеспечивают передачу данных с заданными характеристиками между источником и приемником данных. Эти протоколы вводят новый уровень адресации, так называемый номер порта (port number), который определяет, какому процессу на хосте передаются данные. Номера портов занимают два байта. Существует

список соответствия номеров портов приложениям, определенный в RFC1700 (Request For Comments, запрос для пояснений; данные документы описывают стандарты протоколов Интернета и их взаимодействия). Некоторые зарезервированные порты приведены в табл. 1.8.

Таблица 1.8. *Сервисы и закрепленные за ними порты*

№ порта	Сервис	Описание
7	Echo	Echo
20	FTP-data	Передача данных
21	FTP	Управляющие команды
23	Telnet	Удаленный доступ в систему
25	SMTP	Протокол электронной почты
53	Domain	Сервер доменных имен DNS
80	HTTP	Сервер WWW
110	POP3	Протокол электронной почты
119	NNTP	Телеконференции
123	NTP	Синхронизация времени
161	SNMP	Протокол управления сетевыми устройствами
179	BGP	Маршрутизация

Протокол TCP

Протокол TCP поддерживает надежную передачу данных с предварительным установлением связи между источником и приемником информации. На базе этого протокола реализована большая часть протоколов уровня приложений.

Протокол TCP имеет следующие характеристики, обуславливающее его широкое использование:

- перед началом передачи данных протокол создает канал между источником и приемником информации путем передачи запроса на начало сеанса и получением ответа. По окончании передачи данных сеанс должен быть явно завершен путем передачи соответствующего запроса;
- доставка данных является надежной. Перед отправкой следующего пакета источник информации должен получить подтверждение о приеме предыдущего пакета от приемника информации;
- возможность управления потоком данных;
- возможность доставки экстренных данных.

Эти особенности позволяют программам, использующим протокол TCP, не заботиться об организации надежной передачи данных. С другой стороны, использование этого протокола приводит к уменьшению скорости передачи данных.

Протокол UDP

Протокол UDP обеспечивает логический канал между источником и приемником данных без предварительного установления связи. То есть пакеты, передаваемые по протоколу UDP, не зависят друг от друга, и никакого подтверждения доставки пакета протоколом не предусматривается. Это сильно напоминает бросание бутылки с запиской в море — авось дойдет. Поэтому программы, использующие данный протокол, должны сами организовывать проверку факта доставки информации. Однако благодаря своей простоте протокол UDP может при нормальных условиях передать гораздо больше информации, чем парный ему протокол TCP.

В качестве примера приведем несколько приложений, использующих протокол UDP:

- сервер DNS;
- программы, использующие протокол синхронизации времени NTP;
- программы, использующие протокол удаленной загрузки BOOTP.

Для всех перечисленных программ предполагается, что в случае утери пакета необходимые действия (повторная посылка пакета, выдача сообщения и тому подобные действия) осуществляются самими программами. В случае необходимости гарантированной доставки данных используется протокол TCP.

Протоколы уровня приложений

Последний, четвертый уровень — уровень приложений. К сожалению, почти каждый разработчик программ, использующих протокол уровня приложения, изобретает свой протокол или модифицирует уже существующие, хотя имеется определенный набор протоколов, описанный в соответствующих RFC. В зависимости от используемого протокола транспортного уровня протоколы уровня приложений либо полагаются на надежную доставку данных (протокол TCP), либо придумывают свой способ контроля достоверности данных (при использовании протокола UDP).

Протокол FTP

Протокол передачи файлов используется для организации и приема файлов. Позволяет просматривать каталоги и файлы, переименовывать их, удалять

и т. п. При пересылке файлов контролирует их целостность. Существует "младший брат" протокола FTP — TFTP, который намного проще в реализации, и, в основном, используется для загрузки информации на бездисковые рабочие станции.

Протокол SMTP

Простой протокол передачи почтовых сообщений позволяет работать с электронной почтой. Благодаря тому, что все команды — обычные английские слова, можно с помощью программы telnet подключиться на порт 25 (SMTP) и передавать соответствующие команды с консоли.

Протокол Telnet

Протокол предназначен для удаленного доступа в систему. К примеру, можно с домашнего компьютера через Интернет зайти на рабочий компьютер и выполнять на нем любые команды (запускать программы, редактировать файлы и т. п.). Используется, в основном, для удаленного администрирования системы. Считается небезопасным для операционной системы, т. к. при входе в систему логин (имя пользователя) и пароль передаются в открытом виде. Повсеместно заменяется на протокол SSH.

Сетевая файловая система NFS

Протокол, разработанный фирмой Sun, предназначен для использования дисков и каталогов сервера рабочими станциями в качестве "псевдодисков". Возник очень давно, когда винчестер в сто мегабайт стоил весьма дорого, и использование одного диска, распределяемого через сеть, приводило к существенной экономии денежных средств. Сегодня использование протокола NFS постепенно сходит на нет, одно из немногих мест его применения — бездисковые рабочие станции, использующие NFS в качестве "своей" файловой системы.

Протокол IPX

Протокол обмена пакетами между сетями (Internet Packet Exchange, IPX) разработан фирмой Novell для своего программного продукта NetWare. Однако, начиная с четвертой версии своей операционной системы, фирма Novell стала внедрять поддержку протокола TCP/IP, а в пятой версии протокол TCP/IP стал практически "родным" для NetWare. Тем не менее, протокол IPX еще достаточно широко используется.

Протокол IPX произошел от протокола межсетевых датаграмм (Internet Datagram Protocol, IDP), разработанного в научно-исследовательском цен-

тре Xerox. Протокол IPX реализует механизм сокетов с негарантированной доставкой датаграмм. Поверх протокола IPX может функционировать большое количество протоколов, в том числе:

- протокол данных маршрутизации (Routing Information Protocol, RIP);
- протокол обмена нумерованными пакетами (Sequenced Packet Exchange, SPX), гарантированная доставка;
- протокол Echo;
- протокол сообщений об ошибках;
- протокол обмена пакетами (Packet Exchange Protocol, PEP);
- протокол сервисных объявлений (Service Advertisement Protocol, SAP).

Существует программное обеспечение под Linux (Mars), выполняющее функции сервера NetWare, и программное обеспечение, выступающее клиентом для серверов NetWare. Также есть программное обеспечение под Linux, позволяющее маршрутизировать пакеты IPX.

Протокол AppleTalk

Протокол AppleTalk используется в сетях фирмы Apple. Реально, помимо компьютеров Apple, он не используется нигде. В операционной системе Linux существует поддержка этого протокола, что позволяет взаимодействовать с компьютерами Apple.

Протокол NetBIOS

Протокол, используемый фирмой Microsoft в своих продуктах для оргнизации одноранговых сетей. В последнее время продукты Microsoft по умолчанию используют протоколы TCP/IP.

Протокол DECnet

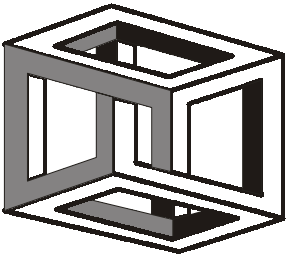
Группа сетевых продуктов фирмы DEC. Поддержка в операционной системе Linux этого протокола существует. Однако маловероятно, что вы столкнетесь с этим протоколом, поскольку фирмы DEC уже нет, а ее мини-компьютеры у нас не распространены.

Литература и ссылки

- Анонимный автор (опытный взломщик). Максимальная безопасность в Linux. — ДиаСофт, 2000.
- Брежнев А. Ф., Смелянский Р. Л. Семейство протоколов TCP/IP.

- RFC792 — Протокол ICMP.
- **gazette.linux.ru.net/rus/articles/povest_ob_ip.html** — Повесть об IP-адресации. Песин И.
- **msk.nestor.minsk.by/kg/2003/05/kg30503.html** — Компьютерная газета Linux в Сети. Сага первая. Подготовил X-Stranger.
- **www.rfc-editor.org** — сайт, посвященный RFC.

ГЛАВА 2



Настройка сети TCP/IP

В этой главе мы рассмотрим в общих чертах вопросы, связанные с конфигурированием разнообразных сетевых устройств и протоколов, необходимых для нормального функционирования в локальной сети, организации модемного соединения и маршрутизатора. Глава в основном является обзорной, поскольку большинство вопросов подробно освещаются в отдельных главах. Начнем же мы с сетевых интерфейсов.

Конфигурирование сетевых интерфейсов

Сетевым интерфейсом можно назвать любое устройство, которое подключается к компьютеру и может устанавливать соединение, производить обмен данными с одним или несколькими хостами. Как правило, каждый интерфейс имеет IP-адрес (а иногда и несколько), с помощью которого можно однозначно его идентифицировать в сети. Также можно присвоить один (или несколько) символических имен как интерфейсу, так и IP-адресу, закрепленному за этим интерфейсом.

Каждый компьютер с ОС Linux имеет виртуальный сетевой интерфейс с символическим именем lo (loopback) и IP-адресом 127.0.0.1. Этот интерфейс должен обязательно присутствовать, иначе множество программ откажутся работать либо будут неверно функционировать.

Для простоты будем считать, что у нас есть сетевая карта, которая подключена к локальной сети. В принципе, большинство сетевых карт опознаются и автоматически устанавливаются во всех современных дистрибутивах, но рассчитывать необходимо на худший вариант. В Linux Ethernet-сетевые карты имеют символическое имя ethX, где X — номер интерфейса (0, 1, 2, ...).

Для каждого сетевого устройства вы должны подключить драйвер (на этапе сборки ядра или динамически) и настроить интерфейс.

Для динамического подключения драйвера необходимо подгрузить модуль ядра, отвечающий за взаимодействие с данным сетевым устройством (например, сетевой картой) и передать ему параметры устройства. Сделать это можно с помощью команды `insmod`, вызов которой осуществляется следующим образом:

```
insmod [-fkmpsxXv] [-o имя_устройства] файл_драйвера
```

Например, для сетевой карты можно выполнить команду следующего вида:

```
insmod -o eth0 /lib/modules/2.4.20/net/rt139.o
```

После подключения драйверов необходимо настроить интерфейс — присвоить IP-адрес и установить нужные значения для других параметров сетевого подключения (DNS, шлюз и т. п.). Наиболее часто для этого используется программа ifconfig (interface configuration).

При запуске ifconfig без аргументов вы получите список активных интерфейсов и их параметры.

Настройка локального интерфейса lo

Этот интерфейс используется для связи программ IP-клиентов с IP-серверами, запущенными на том же компьютере. Для его настройки необходимо выполнить команду

```
ifconfig lo 127.0.0.1
```

А для проверки его работоспособности — используем утилиту ping:

```
ping 127.0.0.1
```

Настройка Ethernet-карты (eth0)

Мы подобрали правильный драйвер для сетевой карты, запустили его. Теперь необходимо закрепить за этой сетевой картой IP-адрес и некоторые дополнительные параметры. Это можно сделать, выполнив программу ifconfig со следующими параметрами (не забывайте, IP-адрес и маска подсети индивидуальны для каждого случая):

```
ifconfig eth0 192.168.0.10 netmask 255.255.255.0 up
```

В некоторых случаях необходимо явно указать прерывание, адрес ввода/вывода или тип физического соединения (коаксиал, витая пара, оптика). Для этого необходимо воспользоваться параметрами командной строки `ifconfig`, такими как `irq`, `io_addr`, `media`.

Помимо этого у команды `ifconfig` есть опции, которые задают различные характеристики интерфейса, например, максимальное число байтов, которое он может передать за один раз (MTU), широковещательный адрес

и т. д. Более подробную информацию смотрите в руководстве по программе ifconfig.

После этого проверяем работоспособность сетевого соединения командой `ping`, задавая в качестве параметра IP-адрес хоста, находящегося в сети и заведомо работоспособного.

После этого необходимо сделать так, чтобы драйверы сетевой карты и настройки интерфейса загружались при старте системы. Для этого нужно добавить соответствующие команды в скрипт инициализации.

Конфигурирование статических маршрутов и маршрута по умолчанию

Установив драйвер сетевого интерфейса, задав IP-адрес и маску подсети, мы еще не закончили с сетевыми настройками. Нам необходимо каким-то образом показать системе, куда должны уходить сетевые пакеты, не предназначенные для сети, заданной при конфигурировании маской подсети. Иными словами, мы должны определить маршрут прохождения пакетов с помощью специальных правил. Данные маршрутизации пакетов хранятся в специальной таблице маршрутизации ядра. Маршруты могут быть *статическими* (задаваемыми при старте системы) и *динамическими*. Для записи правила статической маршрутизации используется программа route. А динамическая маршрутизация выполняется процессом-демоном routed или gated, который отслеживает и модифицирует таблицу маршрутизации на основе сообщений от других хостов сети.

Динамическая маршрутизация используется в том случае, если у вас сложная, меняющаяся структура сети и одна и та же машина может быть доступна по различным интерфейсам.

Для хоста, подключаемого к небольшой локальной сети, в большинстве случаев достаточно статической маршрутизации. Посмотреть таблицу маршрутизации можно, используя программу netstat либо выполнив команду `route` без параметров.

Для того чтобы добавить правило маршрутизации воспользуйтесь командой

`route [-f] операция [-тип] адресат шлюз [dev] интерфейс`

где:

- *операция* — может принимать одно из двух значений: `add` — добавление маршрута или `delete` — удаление;
- *адресат* — может быть IP-адресом хоста, IP-адресом сети или ключевым словом `default`;
- *шлюз* — IP-адрес хоста, на который пересылаются пакеты. `-f` удаляет из таблицы данные обо всех шлюзах;
- *тип* — принимает значения `net` или `host`.

Как правило, бывает необходимо настроить маршрутизацию со следующими правилами (для lo):

```
route add -net 127.0.0.1 lo
```

Для eth0:

```
route add -net 192.168.0.0 netmask 255.255.255.0 eth0
```

и

```
route add default gw 192.168.0.1 eth0
```

Итак, мы задали следующие правила: все пакеты, относящиеся к сети 127, отправляются на интерфейс lo; все пакеты, относящиеся к сети 192.168.0, отправляются на интерфейс eth0; а все пакеты, не предназначенные для сетей 127 и 192.168.0, пересылаются на хост с IP-адресом 192.168.0.1, и дальнейшую их судьбу определяют правила маршрутизации этого хоста.

После того как вы настроили и проверили маршрутизацию, необходимо добавить правила маршрутизации в скрипты, загружаемые при старте операционной системы.

Использование DHCP

Ручное конфигурирование сетевых интерфейсов достаточно утомительно, особенно если у вас большая сеть. Существует возможность автоматически назначать IP-адреса для сетевых интерфейсов, а также передавать хостам адреса DNS серверов, шлюзов, задавать сетевые маски и многое другое. Все это позволяет делать протокол DHCP, подробное описание и конфигурирование которого находится в *главе 4*.

Статический ARP

ARP — протокол преобразования адресов (Address Resolution Protocol) — выполняет преобразование логических сетевых адресов в аппаратные MAC-адреса (Media Access Control). Ядро Linux содержит таблицу ARP, в которой находятся IP-адреса и соответствующие им MAC-адреса. Как правило, эти пары добавляются в таблицу динамически, однако можно сделать таблицу статической и внести данные самостоятельно. Для чего это нужно? Допустим, только некоторые хосты вашей локальной сети имеют право обращаться к какому-то хосту. Но для пользователей некоторых операционных систем (например, Windows 9*x*) не составит труда поменять IP-адрес сетевого интерфейса. В случае с динамическим ARP новая пара IP-MAC-адреса просто заменит старую, в случае же со статическим ARP "левый" интерфейс не получит доступа к хосту.

Для работы с таблицей ARP предназначена утилита arp. Например:

```
arp -a
   IP address      HW type                 HW address
   191.72.1.3      10Mbps Ethernet         00:00:C0:5A:42:C1
   191.72.1.2      10Mbps Ethernet         00:00:C0:90:B3:42
   191.72.2.4      10Mbps Ethernet         00:00:C0:04:69:AA
```

Показывает пары IP-MAC-адресов и тип сети.

Настройка DNS

Как мы уже знаем, каждый хост, подключенный к сети, работающей по протоколу TCP/IP, идентифицируется своим IP-адресом. IP-адрес представляет собой комбинацию четырех чисел, определяющих конкретную сеть и конкретный хост-компьютер в этой сети. IP-адреса трудно запомнить, поэтому для идентификации хоста вместо его IP-адреса можно пользоваться символическим (доменным) именем. *Доменное имя* состоит из двух частей: имени хоста и имени домена. *Имя хоста* идентифицирует его в сети, а *домен* обозначает сеть, частью которой этот хост является. Комбинация имени хоста и имени домена — это уникальное имя, по которому можно однозначно обратиться к хосту. Однако хост в сети можно идентифицировать на уровне TCP/IP только по его IP-адресу, даже если он имеет доменное имя. Для разрешения этого противоречия первоначально использовался файл /etc/hosts, который имеет следующий вид:

```
# IP            local       fully qualified domain name
127.0.0.1       localhost
#
195.18.0.44     main      main.test.com
192.168.0.1     main1
192.168.0.2      boss
```

Файл /etc/networks определяет символические имена сетей:

```
home-net      192.168.0.0
wine-net      191.72.2.0
```

А в файле /etc/hostname записывается имя хоста.

Однако с увеличением количества хостов становилось трудно, а позднее и невозможно поддерживать в актуальном состоянии эти файлы плюс проблемы синхронизации копирования. Для борьбы с этим явлением и предназначена служба DNS (Domain Network Service, доменная система имен).

Служба DNS преобразует символическое имя хоста в его IP-адрес и наоборот: IP-адрес в символическое имя. Служба DNS представляет собой дерево серверов с иерархической структурой поиска.

Запросы на серверы имен посылают особые программы, которые называют *резолверами* (resolver), или определителями. Эта программа предназначена для получения адресов с серверов имен. Для того чтобы мы могли пользоваться доменными именами, необходимо правильно сконфигурировать компьютер при помощи файлов /etc/host.conf и /etc/resolv.conf.

Типичный файл host.conf имеет следующий вид:

```
order hosts, bind
```

Эта запись определяет, что для определения IP-адреса по символическому имени сперва просматривается файл hosts, а если адрес не найден, используется служба DNS. Вообще же в файле host.conf могут быть следующие записи:

- `order`:
  - `hosts` — проверяется наличие имени в локальном файле /etc/host;
  - `bind` — запрашивается адрес у сервера имен DNS;
  - `nis` — для получения адреса используется база данных центра сетевой информации;
  - `alert` — проверяет наличие в локальной системе адресов удаленных узлов, пытающихся получить к ней доступ; устанавливается и отменяется ключевыми словами `on` и `off`;
- `nospoof` — подтверждает правильность адресов удаленных узлов, пытающихся получить доступ к локальной системе;
- `trim` — удаляет имя домена из полного имени и проверяет наличие только имени хоста; позволяет использовать вместо IP-адреса не полное имя *`хост.домен.расширение`*, а только имя хоста, указанное в файле hosts;
- `multi` — позволяет хосту иметь несколько IP-адресов в локальном файле hosts; включается и выключается ключевыми словами `on` и `off`.

Каждая опция может иметь несколько полей, отделенных друг от друга пробелами или знаками табуляции. Для ввода комментария в начале строки нужно ставить знак #. Опции указывают определителю, каким сервисом пользоваться. Важное значение имеет порядок следования опций. Определитель начинает обработку с первой из указанных опций и переходит по очереди к следующим. Для того чтобы программа-определитель могла выполнять свою задачу, ей должен быть предоставлен доступ к серверам доменных имен. В файле resolv.conf содержатся адреса серверов имен, к которым имеет доступ данный хост. В этом файле можно создавать три типа записей, каждая из которых предваряется одним из трех ключевых слов:

- `domain`;
- `nameserver`;
- `search`.

В записи `domain` вводится доменное имя локальной системы. В записи `search` приводится список доменов на тот случай, если задается только имя хоста. Если к какой-либо системе пользователь обращается часто, он может ввести имя ее домена в запись `search`, а затем использовать в качестве адреса только хост-имя. После записей `search` идут записи `nameserver`, если таковые имеются. Для каждого сервера имен, к которому имеет доступ данная система, вводится ключевое слово `nameserver` и IP-адрес. Таких серверов может быть несколько, и порядок их следования в списке определяет очередность опроса серверов.

Пример файла resolv.conf приведен в листинге 2.1.

Листинг 2.1. Файла resolv.conf

```
resolv.conf file
domain test.com
search list.com
nameserver 195.66.200.100
nameserver 211.18.5.43
```

Организация DNS-сервера подробно рассмотрена в *главе 5*.

Протокол PPP

Последние несколько лет протокол PPP стал стандартом де-факто (de-facto) для организации соединения по коммутируемым каналам и выделенным линиям. Поэтому для нормальной работы модемного соединения и извлечения из него максимальной пользы необходимо иметь понятие о протоколе PPP. Итак, PPP — это интернет-стандарт по передаче IP-пакетов по последовательным линиям. PPP поддерживает синхронные и асинхронные линии.

Общая информация

Point-to-Point Protocol (PPP, протокол "точка-точка") разработан для инкапсуляции протоколов вида "point-to-point IP". Помимо этого, целями создания протокола PPP было упрощение выдачи и управления IP-адресами, асинхронной и синхронной инкапсуляцией, смешиванием сетевых протоколов (network protocol multiplexing), конфигурированием и тестированием качества связи, обнаружением ошибок и опциями для установления таких особенностей сетевого уровня, как настройка адресов и установка сжатия данных. Для поддержки вышеперечисленных качеств, PPP должен предоставлять управление по расширенному протоколу Link Control Protocol (LCP,

протокол управления соединением) и семейству протоколов Network Control Protocols (NCPs, протоколы управления сетью), которые используются для установления параметров связи. На сегодняшний день PPP поддерживает не только IP, но и другие протоколы, включая IPX и DECNet.

Свойства протокола PPP

В табл. 2.1 приведены основные возможности, реализованные протоколом PPP. Однако следует учитывать, что используемое программное обеспечение может не полностью реализовывать эти возможности, поэтому рекомендуется предварительно ознакомиться с описанием применяемых программ, особенно в гетерогенной среде.

Таблица 2.1. *Основные возможности, реализуемые протоколом PPP*

Свойство	Описание
Demand on dial (дозвон по запросу)	Подключение PPP-интерфейса и набор телефонного номера по приходу пакета. Отключение интерфейса PPP после некоторого периода отсутствия активности
Redial	Подключение PPP-интерфейса, который потом не будет отключен и будет всегда сохранять в своем распоряжении подключенный канал
Campling	См. Redial
Scripting	Настройки через серию сообщений или промежуточных соединений для установления PPP-соединения — больше похоже на последовательности, используемые для установления связи по UUCP
Parallel	Конфигурирование нескольких PPP-линий для одного и того же подключения к хосту в целях равномерного разделения трафика между ними (в процессе стандартизации)
Filtering	Выборка, при каких пакетах имеет смысл начинать прозвон по линии, а при каких нет, отталкиваясь в принятии решения от IP- или TCP-типа пакета или TOS (Type of Service). К примеру, игнорировать все ICMP-пакеты
Header Compression (сжатие заговка)	Сжатие TCP-заголовка в соответствии с RFC1144
Server	Принятие входящих PPP-соединений, которые могут также требовать дополнительной маршрутизации
Tunneling	Построение виртуальных сетей по PPP-соединению через TCP-поток существующей IP-сети

Таблица 2.1 *(окончание)*

Свойство	Описание
Extra escaping	Байт-ориентированные символы, не входящие в стандартный набор символов, используемый при установлении связи, они могут быть сконфигурированы отдельно, но также не пересекаться с теми, что используются при установлении связи

Как видите, протокол имеет большие возможности, и не удивительно, что некоторые из них не реализованы в полной мере.

Составляющие PPP

PPP предоставляет возможность передачи датаграмм по последовательным point-to-point-линиям и имеет три составляющие:

- метод предоставления инкапсуляции датаграмм по последовательным PPP-линиям с использованием HDLC (High-Level Data Link Control, высокоуровневое управление данными соединения) — протокол для упаковки датаграмм по PPP средствами связи;
- расширенный протокол LCP для установления, конфигурирования и тестирования физического соединения;
- семейство протоколов NCP для установления и управления другими сетевыми протоколами. Это позволяет протоколу PPP поддерживать одновременно несколько сетевых протоколов.

Функционирование протокола PPP

В момент установления связи через PPP-соединение, PPP-демон вначале шлет пакеты LCP для конфигурирования и тестирования линии связи. После того как связь и дополнительные возможности будут установлены посредством протокола LCP, PPP-демон посылает NCP-фреймы для изменения и настройки одного или более сетевых протоколов. По окончании процесса настройки сетевые пакеты могут передаваться через установленное соединение. Оно будет оставаться активным до тех пор, пока специальные LCP- или NCP-пакеты не закроют соединение, или до тех пор, пока не произойдет какое-нибудь внешнее событие, которое приведет к потере соединения, например, сработает таймер отсутствия активности или разорвется модемное соединение.

Поддерживаемое оборудование

Протокол PPP адаптирован для работы с любым DTE/DCE-интерфейсом, включая RS-232, RS-422, RS-423, CITT V.35. Помимо этих интерфейсов

протокол может работать практически на любом оборудовании, единственное требование — наличие дуплексного режима.

Структура пакета протокола PPP

Протокол PPP использует принципы, терминологию и структуру пакетов, описанных в стандартах ISO, касающихся HDLC:

- ISO 3309-1984/PDAD1 "Addendum 1: Start/stop transmission" — описывает процедуру начала и завершения передачи данных;
- ISO 3309-1979 — описывает структуру пакетов HDLC для использования в синхронных системах;
- ISO 3309:1984/PDAD1 — описывает предложения по изменениям в ISO 3309-1979, которые позволяют использовать асинхронные системы.

На рис. 2.1 изображен формат пакета протокола PPP.

Величина поля, байт	1	1	1	2	Переменный	2 или 4
Назначение	Флаг	Адрес	Управление	Протокол	Данные	Контрольная сумма

Рис. 2.1. Структура пакета протокола PPP

Рассмотрим значения полей пакета протокола PPP:

- `Флаг` — один байт, обозначающий начало или конец пакета. Поле флага содержит двоичную последовательность 01111110;
- `Адрес` — один байт, содержащий двоичную последовательность 11111111, стандартный широковещательный адрес. PPP не поддерживает индивидуальную адресацию станций;
- `Управление` — один байт, содержащий двоичную последовательность: 00000011, который посылается для передачи пользовательских данных в неразделенных пакетах;
- `Протокол` — 2 байта определяют протокол, упакованный в пакете протокола PPP. Значения протоколов можно узнать в соответствующем RFC;
- `Данные` — 0 или больше байтов, составляющих датаграмму протокола, указанного в поле `Протокол`. Конец информационного поля определяется нахождением заканчивающей последовательности и 2-байтовой последовательности в поле контрольной суммы. По умолчанию максимальная длина поля данных 1500 байт. Однако во время установления сеанса программы pppd могут договориться использовать другое значение поля данных;

- `Контрольная сумма` — обычно 16 бит. Однако при установлении соединения pppd могут договориться об использовании 32-битовой контрольной суммы.

PPP-протокол управления соединением (LCP)

PPP-протокол управления соединением (LCP) предоставляет методы для установления, конфигурирования, поддержания и тестирования PPP-соединения. Протокол LCP выполняет несколько функций.

- Конфигурирование и установление связи. Перед передачей какой-либо информации (к примеру, пакет IP) протокол LCP должен открыть соединение и произвести начальный обмен параметрами настройки. Этот этап заканчивается, когда пакет о подтверждении произведенной настройки будет послан и принят обратно.
- Определение качества связи. Протокол LCP позволяет (но эту возможность зачастую не используют) добавить фазу тестирования канала связи. Тестирование канала связи должно происходить сразу же за конфигурированием и установлением связи. Во время определения качества связи проверяется, способно ли соединение с достаточным качеством транспортировать какой-либо сетевой протокол.
- Установление настроек сетевого протокола. После того как протокол LCP закончит определение параметров связи, сетевые протоколы должны быть независимо друг от друга настроены соответствующими протоколами NCP, которыми могут в любой момент времени начать или прекратить пользоваться.
- Окончание связи. Протокол LCP может в любое время прервать установленную связь. Это может произойти по требованию пользователя или из-за какого-нибудь события, к примеру потери несущей, или по истечении допустимого периода времени неиспользования канала.

Существует три типа LCP-пакетов:

- пакеты установления — используются для установления и настройки связи;
- пакеты прерывания — используются для прерывания установленной связи;
- пакеты сохранения связи — используются для управления и диагностики связи.

Сокращения, используемые при описании протокола PPP

В табл. 2.2 приведены некоторые аббревиатуры, используемые при описании протокола PPP. Расшифровка содержит как английское значение, так и русский перевод.

Таблица 2.2. *Аббревиатуры, используемые при описании протокола PPP*

Аббревиатура	Расшифровка
ack	ACKnowlegement — получено
AO	Active Open [state diagram] — соединение активно
C	Close [state diagram] — соединение закрыто
CHAP	Challenge-Handshake Authentication Protocol (RFC1334) — протокол аутентификации
D	Lower Layer Down [state diagram] — нижний уровень отсутствует
DES	Data Encryption Protocol — протокол шифрования данных
DNA	Digital Network Architecture — архитектура цифровых сетей
IETF	Internet Engineering Task Force — организация, непосредственно отвечающая за разработку протоколов и архитектуры сети Интернет
FCS	Frame Check Sequence [X.25] — проверочная последовательность кадра
LCP	Link Control Protocol — протокол управления соединением
LQR	Link Quality Report — отчет о качестве соединения
MD4	MD4 digital signature algorithm — протокол цифровой подписи
MD5	MD5 digital signature algorithm — протокол цифровой подписи
MRU	Maximum Receive Unit — максимальная величина принимаемого кадра
MTU	Maximum Transmission Unit — максимальная величина передаваемого кадра
nak	Negative AcKnowledgement — негативный ответ
NCP	Network Control Protocol — протокол сетевого управления
PAP	Password Authentication Protocol (RFC1334) — протокол аутентификации
PDU	Protocol Data Unit — пакет
PO	Passive Open — пассивное соединение
PPP	Point-to-Point Protocol — протокол "точка-точка"
RCA	Receive Configure-Ack — принят конфигурационный запрос
RCJ	Receive Code-Reject — принят код отклонения
RCN	Receive Configure-Nak или Reject — принят код отклонения
RCR+	Receive good Configure-Request [state diagram] — принят нормальный запрос конфигурации

Таблица 2.2 *(окончание)*

Аббревиатура	Расшифровка
RER	Receive Echo-Request — принят эхо-запрос
RTA	Receive Terminate-Ack [state diagram] — принят запрос на разрыв соединения
RUC	Receive unknown code [state diagram] — принят неизвестный код
sca	Send Configure-Ack [state diagram] — послан конфигурационный запрос
scj	Send Code-Reject [state diagram] — послан код отклонения
scn	Send Configure-Nak or -Reject [state diagram] — послан код отклонения
ST-II	Stream Protocol — потоковый протокол
TO+	Timeout with counter > 0 [state diagram] — счетчик тайм-аута больше, чем ноль
TO-	Timeout with counter expired [state diagram] — счетчик тайм-аута превысил предел
VJ	Van Jacobson (RFC1144 header compression algorithm) — алгоритм компрессии заголовка пакетов PPP
XNS	Xerox Network Services — сетевые службы Xerox

Стандарты, описывающие протокол PPP

Существует несколько стандартов (RFC) протокола PPP. Они приведены в табл. 2.3.

Таблица 2.3. *Стандарты протокола PPP*

Номер RFC	Название
1144	Compressing TCP/IP headers for low-speed serial links — Сжатие заголовков пакетов для низкоскоростных последовательных соединений
1220	Point-to-Point Protocol extensions for bridging — Расширение протокола PPP
1332	PPP Internet Protocol Control Protocol (IPCP) — Управляющий протокол IP
1333	PPP link quality monitoring — Контроль качества соединения PPP
1334	PPP authentication protocols — Протоколы аутентификации PPP

Таблица 2.3 *(окончание)*

Номер RFC	Название
1547	Requirements for an Internet Standard Point-to-Point Protocol — Требования для интернет-стандарта PPP
1552	The PPP Internetwork Packet Exchange Control Protocol (IPXCP) — Управляющий протокол обмена пакетами для разнородных сетей
1570	PPP LCP Extensions — Расширения протокола LCP
1598	PPP in X.25 — Использование протокола PPP в сетях X.25
1618	PPP over ISDN — Использование протокола PPP поверх протокола ISDN
1619	PPP over SONET/SDH — Использование протокола PPP поверх протокола SONET/SDH
1638	PPP Bridging Control Protocol (BCP) — Протокол управления PPP
1661	The Point-to-Point Protocol (PPP) — Протокол "точка-точка"
1662	PPP in HDLC-like Framing — PPP в HDLC-подобных кадрах
1663	PPP Reliable Transmission — Надежная передача PPP
1717	The PPP Multilink Protocol (MP) — Многопоточный PPP-протокол

Практическое использование протокола PPP описано в *главе 3*.

Протокол SLIP/CSLIP

Как и протокол PPP, SLIP и CSLIP — это протоколы, адаптирующие IP для работы на последовательных линиях. Основная функция SLIP — организовать пересылку IP-пакетов по последовательной линии, которая не предусматривает деления пересылаемой информации на какие-либо отдельные блоки и пересылает все данные единым непрерывным потоком.

Первым стандартом де-факто, позволяющим устройствам, соединенным последовательной линией связи, работать по протоколам TCP/IP, был протокол SLIP (Serial Lines Internet Protocol, интернет-протокол для последовательных соединений), созданный в начале 80-х годов.

Протокол SLIP

Протокол SLIP использует специальные символы для ограничения кадра данных в последовательном канале. Для того чтобы распознать границы SLIP-кадров, передаваемых по последовательной линии связи, протокол SLIP предусматривает использование специального символа `END`, значение которого в шестнадцатеричном представлении равно `C0`. Для того чтобы не

возникало конфликтов, байт данных со значением, равным значению символа END, заменяется составной двухбайтовой последовательностью, состоящей из специального символа ESC (DB) и кода DC. Если же байт данных имеет тот же код, что и символ ESC, то он заменяется двухбайтовой последовательностью, состоящей из собственно символа ESC и кода DD. После последнего байта пакета передается символ END.

Хотя в спецификации протокола SLIP не определена максимальная длина передаваемого SLIP-кадра, реальный его размер определяется длиной IP-пакета и не должен превышать 1006 байт. Данное ограничение связано с первой реализацией протокола SLIP в соответствующем драйвере для Berkley Unix, и его соблюдение необходимо для поддержки совместимости разных реализаций SLIP.

Другой недостаток SLIP — отсутствие индикации типа протокола, пакет которого инкапсулируется в пакет SLIP. Поэтому через последовательную линию по протоколу SLIP можно передавать трафик лишь одного сетевого протокола.

Но особенно большим недостатком протокола SLIP является то, что в нем НЕ предусмотрено процедуры коррекции ошибок.

Протокол CSLIP

Низкая пропускная способность последовательных линий вынуждает сокращать время передачи пакетов, уменьшая объем содержащейся в них служебной информации. Эта задача решается с помощью протокола CSLIP (Compressed SLIP), поддерживающего сжатие заголовков пакетов. На низких скоростях передачи данных эта разница заметна только при работе с пакетами, несущими малые объемы информации. На больших же скоростях CSLIP дает меньший выигрыш и почти ничего не дает для пакетов с большими объемами данных, например, FTP-пакетов.

CSLIP для пересылки пакета использует информацию из предыдущего пакета, таким образом, передача искаженных пакетов приводит к большим потерям времени, чем обычный SLIP.

Протокол PPP лишен этих недостатков, поэтому SLIP/CSLIP сейчас практически не используется.

Процесс init

После того как ядро Linux полностью загрузилось, считало конфигурационные параметры и настроило оборудование (по крайней мере то, которое упоминалось в конфигурационных параметрах, и то, драйверы которого присутствуют в ядре), оно приступает к монтированию разделов жесткого диска. Монтирование всегда начинается с корневой файловой системы. Как

только корневая файловая система окажется загружена и смонтирована, будет выведено сообщение:

```
VFS: Mounted root (ext2 filesystem) readonly
```

В этой точке система находит на корневой файловой системе программу init и выполняет ее.

Процесс init — это программа, ответственная за продолжение процедуры загрузки и перевод операционной системы из начального состояния, возникающего после загрузки ядра, в стандартное состояние. Во время этого процесса init выполняет множество операций, необходимых для нормального функционирования операционной системы: монтирование и проверку файловых систем, запуск различных служб и т. п. Список производимых действий помимо конфигурации системы зависит от так называемого уровня выполнения (run level).

Каждый уровень выполнения однозначно (по крайней мере, в пределах дистрибутива) определяет перечень действий, выполняемых процессом init, и конфигурацию запущенных процессов. К сожалению (а может, и к счастью), нет четкого разделения на уровни выполнения, их количество и действия, выполняемые на каждом уровне. Например, в некоторых UNIX-системах уровней выполнения всего два. Некоторые дистрибутивы Linux таким же образом конфигурируют свою операционную систему (например, в дистрибутиве Slackware два уровня выполнения). В других дистрибутивах (Red Hat Linux) уровней выполнения восемь. Поскольку эта книга базируется на дистрибутиве Red Hat, дальнейшее описание на нем и основано.

В операционной системе Linux существует восемь уровней выполнения:

- 0 — останов системы;
- 1 — однопользовательский режим для специальных случаев администрирования. Отсутствует поддержка сети, практически нет сервисов;
- 2 — многопользовательский режим без поддержки сети;
- 3 — многопользовательский режим с поддержкой сети;
- 4 — использование не регламентировано;
- 5 — обычно по умолчанию стартует X Window System;
- 6 — перезагрузка системы;
- S или s — практически то же, что и однопользовательский режим, но уровень выполнения S. В основном используется в скриптах.

Как можно заметить, существует определенное логическое нарушение в следовании уровней выполнения. Было бы более логично нулевой уровень выполнения вставить перед шестым. Однако здесь проявили себя исторические традиции — как повелось много лет назад в UNIX, так ради совместимости и остается.

К сожалению, не существует единого мнения, как использовать уровни со второго по пятый. По большей части это определяется идеологами дистрибутива или пристрастиями системного администратора.

Конфигурационный файл init (/etc/inittab)

Как всякая программа, после старта init сразу считывает свой конфигурационный файл /etc/inittab. Это обычный текстовый файл, состоящий из отдельных строк. Если строка начинается со знака # (стандартный признак комментария в конфигурационных файлах и скриптах) или пуста, она игнорируется. Все остальные строки состоят из 4 разделенных двоеточиями полей, имеющих вид:

```
id:runlevels:action:process
```

где:

- `id` — идентификатор строки. Выбирается произвольно, но в файле не может быть двух строк с одинаковыми идентификаторами. Если конфигурационный файл модифицируется достаточно часто, то имеет смысл использовать неписаное правило нумерации строк в Basic — номера строкам назначать кратно пяти или десяти;
- `runlevels` — уровни выполнения, на которых эта строка будет задействована. Уровни задаются цифрами (без разделителей);
- `process` — команда, которая должна быть запущена;
- `action` — действие. В этом поле стоит ключевое слово, которое определяет, что должен делать процесс init, пока выполняется (или после выполнения) команда, заданная полем `process`:
  - `wait` — ожидать завершения процесса. Соответственно, пока не закончится данный процесс, init не запускает никаких других процессов. Как правило, такого типа процессы используются для разнообразных проверочных действий (проверка и восстановление файловых систем), а также для запуска различных служб (демонов);
  - `once` — выполнять процесс только один раз;
  - `respawn` — перезапустить процесс в случае его "смерти". Актуально для некоторых служб, которые должны постоянно присутствовать в системе;
  - `off` — игнорировать данный элемент. Можно использовать при отладке конфигурационного файла;
  - `boot` — процесс должен быть выполнен при загрузке операционной системы, поле уровней выполнения при этом игнорируется;
  - `bootwait` — то же, что и предыдущая опция, но init должен ожидать окончания работы процесса;

- `initdefault` — указывает init, в какой уровень выполнения необходимо перейти системе после загрузки;
- `sysinit` — процесс должен быть выполнен во время загрузки операционной системы до выполнения любой строки с `boot` или `bootwait`;
- `powerwait` — позволяет процессу init остановить систему при пропадании электроэнергии. Применение этого ключевого слова предполагает, что используется источник бесперебойного питания (UPS), имеющий специальный интерфейс, с помощью которого UPS может посылать в компьютер и принимать из него различные управляющие сигналы (например, "нет питания", "выключить источник бесперебойного питания", "аккумуляторы разряжены" и т. п.), а также программное обеспечение, которое отслеживает состояние источника бесперебойного питания и информирует init о том, что питание отключилось;
- `ctrlaltdel` — разрешает init перезагрузить систему, когда пользователь нажимает комбинацию клавиш <Ctrl>+<Alt>+<Del>. Однако системный администратор может определить действия по комбинации клавиш <Ctrl>+<Alt>+<Del>, например, игнорировать нажатие этой комбинации.

Этот список не является исчерпывающим. Подробную информацию о файле inittab можно узнать из man-страниц init, inittab.

В качестве примера приведем файл inittab (листинг 2.2), который находится в только что установленной системе Red Hat 7.1.

Листинг 2.2. Файл inittab

```
# inittab    Этот файл описывает как процесс init должен настроить
#  операционную систему в соответствующем уровне выполнения
#
# Author:  Miquel van Smoorenburg, <miquels@drinkel.nl.mugnet.org>
#       Modified for RHS Linux by Marc Ewing and Donnie Barnes
#

# Default runlevel. The runlevels used by RHS are:
#   0 — halt (Do NOT set initdefault to this)
#   1 — Single user mode
#   2 — Multiuser, without NFS (The same as 3, if you do not have
#       networking)
#   3 — Full multiuser mode
#   4 — unused
#   5 — X11
#   6 — reboot (Do NOT set initdefault to this)
```

```
#
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

# Things to run in every runlevel.
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# When our UPS tells us power has failed, assume we have a few
# minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting
Down"

# If power was restored before the shutdown kicked in, cancel it.
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

# Run xdm in runlevel 5
# xdm is now a separate service
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Сразу после запуска процесс init считывает свой конфигурационный файл /etc/inittab и производит его разбор. Сначала он определяет, какой уровень по умолчанию установлен в системе. Как видно из приведенного конфигурационного файла — `id:3:initdefault` — уровень выполнения, в котором будет функционировать операционная система после загрузки, равен трем (т. е. предполагается многопользовательский режим с поддержкой сетевых функций). Дистрибутив Red Hat по умолчанию предлагает установить вход в систему в графическом режиме — пятый уровень выполнения.

Затем процесс init принимает к сведению строки, содержащие специальные команды, такие как:

```
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"
```

После этого процесс init выполняет команду, которую необходимо выполнить при старте системы, но перед тем как перейти к какому-нибудь уровню выполнения. Эта команда содержится в строке с ключевым словом `sysinit`:

```
si::sysinit:/etc/rc.d/rc.sysinit
```

После этого процесс init выполняет скрипты, которые должны исполняться в любом уровне выполнения:

```
ud::once:/sbin/update
```

а затем команды, соответствующие уровню, заданному по умолчанию:

```
l3:3:wait:/etc/rc.d/rc 3
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
```

Как можно заметить, есть несколько строк, запускающих скрипт rc, которые отличаются только уровнем выполнения и аргументом командной строки, передаваемой в этот скрипт. После выполнения скрипта rc процесс init осуществляет запуск шести виртуальных консолей (процессов mingetty или, в более старом варианте, — getty), что дает пользователям возможность регистрироваться в системе с терминалов (или виртуальных консолей, поскольку терминал вы вряд ли где-нибудь встретите). Для переключения между виртуальными консолями необходимо нажимать комбинацию клавиш <Alt>+<F*№ виртуальной_консоли*>. После инициализации виртуальных консолей можно считать, что система полностью перешла в соответствующий

уровень выполнения, загрузка завершилась и операционная система ожидает регистрации пользователя.

По окончании загрузки init продолжает функционировать в фоновом режиме. Поэтому с помощью команды `telinit`, которая взаимодействует с процессом init, можно произвести перевод системы с одного уровня выполнения на другой или указать init перечитать свой конфигурационный файл.

Когда пользователь останавливает систему (командой `shutdown`, `halt`, `poweroff` или `reboot`), процесс init завершает все исполняющиеся процессы, размонтирует все файловые системы и останавливает процессор или производит перезагрузку системы.

Основные конфигурационные файлы

Таким образом, в итоге рассмотрения предыдущего раздела мы установили, что процесс init выполняет три основных действия:

- запускает скрипт rc.sysinit из каталога /etc/rc.d;
- запускает скрипт rc из того же каталога /etc/rc.d с опцией, равной уровню выполнения (обычно третий или пятый уровни выполнения);
- запускает процессы getty.

В каталоге /etc находится каталог rc.d, содержимое которого непосредственно касается процесса загрузки системы. Вот оно:

/init.d
/rc0.d
/rc1.d
/rc2.d
/rc3.d
/rc4.d
/rc5.d
/rc6.d
rc
rc.local
rc.sysinit

Опираясь на предыдущую информацию, нетрудно заметить, что существует семь каталогов для каждого уровня выполнения, каталог /init.d и три исполняемых файла, два из которых нам уже знакомы: rc и rc.sysinit. Третий файл rc.local вызывается по окончании исполнения файла rc и предназначен для команд, добавляемых администратором для запуска в процессе начальной загрузки. Редактировать файл rc не возбраняется, однако вероятность ошибки в файле, содержащем сотню-другую строк, очень велика, поэтому настоятельно рекомендуется использовать только файл rc.local.

Файл rc.sysinit

Вернемся к процессу загрузки. Файл rc.sysinit предназначен для выполнения начальных действий, необходимых для корректного функционирования операционной системы. Далее приведен список действий, выполняемых скриптом rc.sysinit. Конечно, он зависит от дистрибутива и от конфигурации системы, но в большей части он неизменен.

Действия скрипта:

- установка путей;
- установка имени хоста;
- чтение конфигурационных данных из /etc/sysconfig/network;
- вывод баннера;
- монтирование файловой системы /proc;
- конфигурирование параметров ядра системы, используя файл /etc /sysctl.conf;
- установка системных часов, используя конфигурацию из файла /etc /sysconfig/clock;
- установка параметров клавиатуры консоли программой loadkeys в соответствии с файлами /etc/sysconfig/console/default.kmap или /etc/sysconfig /keyboard;
- загрузка системного шрифта из /etc/sysconfig/i18n и файлов с расширением pcf.gz или gz из каталогов /etc/sysconfig/console, /usr/lib/kbd /consolefonts или /lib/kbd/consolefonts;
- активация области подкачки;
- инициализация USB-контроллера;
- запуск программы fsck для корневой системы, при обнаружении серьезных проблем выполняется немедленная перезагрузка;
- старт PNP-устройств в соответствии с файлом /etc/isapnp.conf;
- перемонтирование корневой файловой системы в режим чтения/записи;
- перенастройка таблицы монтирования /etc/mtab;
- проверка квот для корневой файловой системы;
- проверка необходимости загрузки модулей, нахождение зависимостей, загрузка и конфигурирование модулей;
- подключение RAID-устройств;
- запуск fsck для других систем;
- монтирование локальных файловых систем;
- включение механизма квот;

- удаление триггерных файлов загрузки;
- очистка каталогов /var/lock и /var/run;
- очистка файлов /var/run/utmp и /var/run/utmpx;
- удаление файлов-защелок из каталога /tmp;
- включение подкачки;
- инициализация последовательных устройств, используя скрипт /etc/rc.d /rc.serial;
- загрузка модулей для SCSI-стримера;
- генерация файла заголовка для определения загружаемого ядра командой /sbin/mkkerneldoth;
- установка ссылки /boot/System.map;
- проверка использования интерактивного режима загрузки и, в случае необходимости, создание файла /var/run/confirm.

Запуск проверки файловой системы командой `fsck` может быть принудительно отключен при наличии файла /fastboot, а также включен при наличии /forcefsck. Создать эти файлы можно выполнением команды `shutdown` с соответствующими ключами. Однако не рекомендуется злоупотреблять данными возможностями.

Sysctl позволяет зафиксировать ряд параметров и обеспечить (через /etc/sysctl.conf) их установку после перезагрузки. Вид /etc/sysctl.conf сразу после инсталляции системы приведен в листинге 2.3.

Листинг 2.3. Файл sysctl.conf

```
# Disables packet forwarding
net.ipv4.ip_forward = 0
# Enables source route verification
net.ipv4.conf.all.rp_filter = 1
# Disables the magic-sysrq key
kernel.sysrq = 0
```

Файл rc

Прежде чем приступить к разбору скрипта rc, необходимо упомянуть о каталогах /rcX.d и /init.d. Уточним еще раз — иерархия /rcX.d характерна для дистрибутивов Red Hat и базирующихся на нем, в других дистрибутивах и в UNIX-системах их может и не быть. Эти каталоги играют исключительную роль в процессе загрузки, поскольку они содержат основные скрипты, необходимые для организации процесса загрузки.

Подкаталог /init.d содержит по одному скрипту для каждой из служб, установленных в системе (sendmail, HTTP, Samba, FTP и т. п.). Этот скрипт отвечает за запуск, остановку или перезагрузку соответствующей службы. В каталоге /rcX.d размещаются ссылки на файлы скриптов, как правило, находящиеся в каталоге /etc/rc.d/init.d. Названия этих ссылок имеют имена, начинающиеся либо с буквы K, либо с буквы S, после которой идет двухзначное число и имя соответствующей службы. Буквы S и K — первые буквы слов start и kill, соответственно. Из этого следует, что файл, начинающийся с буквы S, отвечает за старт соответствующего процесса, а файл, начинающийся с буквы K, отвечает за остановку соответствующего процесса. Цифры, идущие после S или K в именах ссылок, задают порядок запуска скриптов.

Заглянем в файл rc. Первым делом скрипт пытается определить текущий уровень выполнения и уровень выполнения, в который необходимо перевести систему. После этого он проверяет, нажимал ли пользователь клавишу <I> для перехода в режим пошаговой загрузки процессов. Затем скрипт останавливает запущенные на предыдущем уровне выполнения процессы, отсутствующие на новом уровне выполнения, а потом запускает необходимые службы для нового уровня выполнения. Как правило, одна и та же служба нужна на нескольких уровнях. Поэтому не имеет смысла при переходе с одного уровня выполнения на другой останавливать и тут же запускать данную службу. В Linux для этой цели используются специальные флаги. В качестве флагов служат файлы в каталоге /var/lock/subsys/${subsys} или /var/lock/subsys/${subsys}.init, где subsys — имя соответствующей службы. Если файлов нет, то данный процесс считается незапущенным (запуск S-файла имеет смысл), а если есть — запущенным (запуск K-файла имеет смысл). Так же для программы linuxconf создается специальный флаг /var/run/runlevel.dir, из которого можно узнать текущий уровень выполнения системы.

Для управления набором доступных служб в текущем уровне выполнения можно использовать программу конфигурирования linuxconf, программу ntsysv (рис. 2.2), /usr/sbin/setup или графическую программу control-panel (рис. 2.3).

Можно сконфигурировать набор доступных сервисов и вручную. Для запрета старта какого-либо сервиса достаточно просто удалить соответствующую ссылку (SXXlalala) из необходимого каталога /rcX.d, а для разрешения — создать соответствующую ссылку в нужном каталоге /rcX.d. Однако не следует забывать помимо стартовой ссылки создавать стоповую, иначе возможны проблемы, когда система некорректно завершит функционирование сервиса, для которого забыли создать стоповую ссылку. А как же корректно установить порядковый номер у соответствующей ссылки? Конечно, можно чисто эмпирически подобрать номер, исходя из функций, выполняемых сервисом. Но давайте заглянем в любой файл в каталоге /etc/rc.d/init.d/, к примеру, в файл anacron (листинг 2.4).

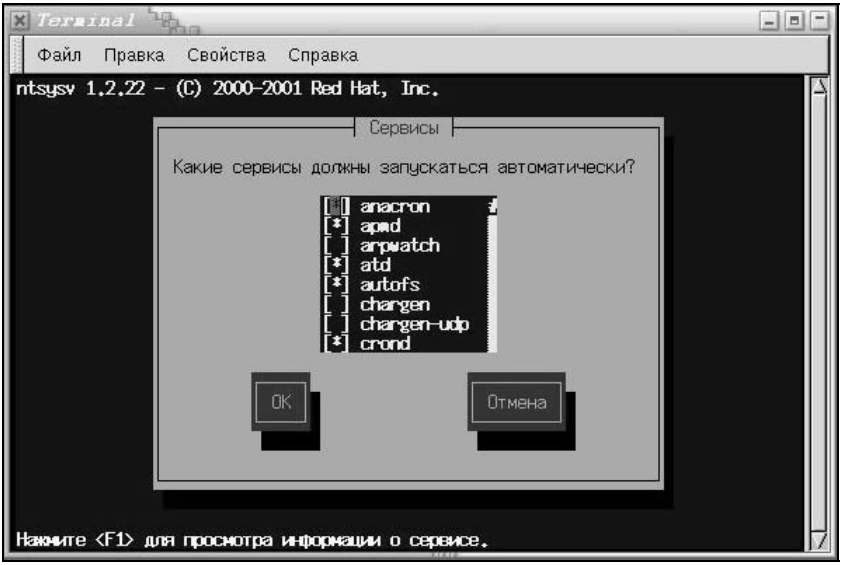


Рис. 2.2. Программа ntsysv

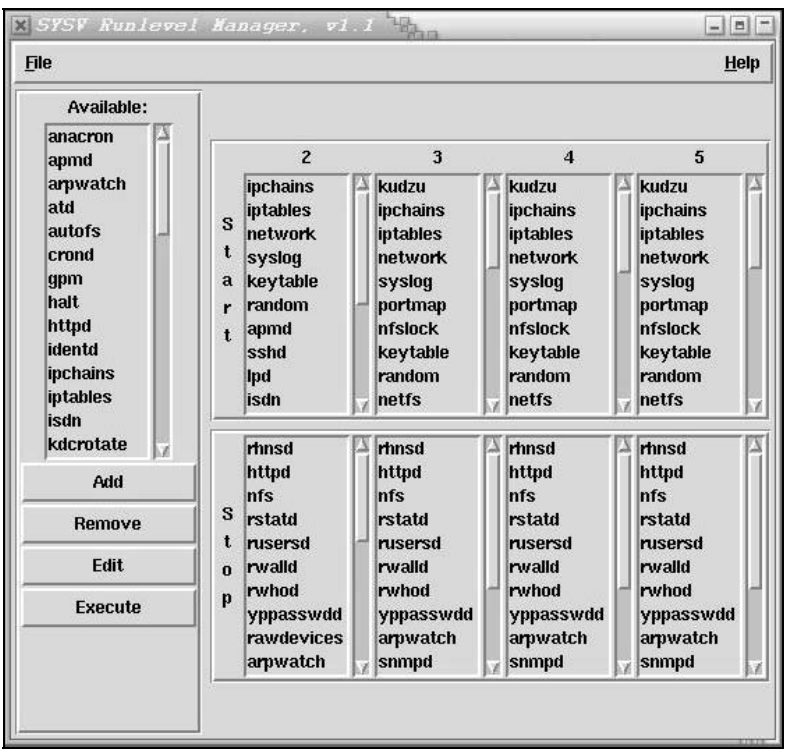


Рис. 2.3. Программа control-panel

Листинг 2.4. Файл anacron

```
#!/bin/sh
# Startup script for anacron
# chkconfig: 2345 95 05
# description: Run cron jobs that were left out due to downtime

# Source function library.
. /etc/rc.d/init.d/functions
[ -f /usr/sbin/anacron ] || exit 0
prog="anacron"
start() {
    echo -n $"Starting $prog: "
    daemon anacron
    RETVAL=$?
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/anacron
    echo
    return $RETVAL
}
stop() {
    if test "x"pidof anacron"" != x; then
      echo -n $"Stopping $prog: "
      killproc anacron
      echo
    fi
    RETVAL=$?
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/anacron
    return $RETVAL
}
case "$1" in
      start)
      start
     ;;
      stop)
      stop
     ;;
      status)
      status anacron
     ;;
      restart)
      stop
      start
     ;;
```

```
      condrestart)
      if test "x"pidof anacron"" != x; then
        stop
        start
      fi
     ;;
      *)
      echo $"Usage: $0 {start|stop|restart|condrestart|status}"
      exit 1
esac
exit 0
##################################################################
```

Обратите внимание на часть заголовка файла:

```
#!/bin/sh
# Startup script for anacron
# chkconfig: 2345 95 05
# description: Run cron jobs that were left out due to downtime
```

Помимо указания, какой командной оболочкой необходимо воспользоваться, там есть строчка

```
# chkconfig: 2345 95 05
```

из которой следует, что этот скрипт может использоваться во втором, третьем четвертом и пятом уровнях выполнения, а цифры 95 и 05 — порядковый номер для стартового (95) и стопового (05) скриптов. Обратите внимание — в сумме эти две цифры составляют 100. Таким образом, достаточно просто добиться того, чтобы порядок останова сервисов был в точности обратный стартовому. `Description` в данном файле — комментарий, который linuxconf выдает на экран для объяснения роли данного сервиса.

Если внимательно посмотреть скрипт, то сразу видно, что опций у него больше, чем стандартных `start` и `stop`. Имеются еще `restart`, `condrestart` и `status`. Старт, останов и проверка состояния демона выполняется рядом функций типа `daemon`, `killproc`, `status`. Процедуры `daemon`, `killproc`, `status` определяются в файле /etc/rc.d/init.d/functions (а тот пользуется определениями из /etc/sysconfig/init). Они предназначены для старта, останова и проверки статуса демона (сервиса).

Функция `daemon` обеспечивает старт сервиса. При этом можно учесть особенности поведения демона. Перед стартом сервиса всегда делается проверка его наличия в системе. Так как появление дампа (дамп — моментальный снимок памяти, используемой зависшей программой на момент ее краха) памяти сервиса может привести к проблемам с безопасностью, то все демоны запускаются в режиме без создания дампа памяти.

Останов сервиса выполняется процедурой `killproc`. Данная функция предполагает один аргумент в виде имени демона и, при необходимости, еще один для указания сигнала, который будет послан сервису. Сигнал `SIGKILL` часто может быть не желателен для останова сервиса, поэтому если сигнал назначен, то используется только он, в противном случае, сперва посылается `SIGTERM`, и если данный сигнал не произвел на процесс впечатления, то посылается сигнал `SIGKILL`. В последнюю очередь скрипт "подчищает" различные блокировочные файлы.

Функция `status` позволяет проверить текущее состояние сервиса. Если сервис нормально функционирует, то просто сообщается об этом факте. В противном случае осуществляется проверка на наличие флаговых файлов (/var/run/*подсистема*.pid и /var/lock/subsys/*подсистема*), которые должны блокировать повторный запуск сервиса. Таким образом, несложно самому создать скрипт для управления сервисом.

Файл rc.local

Файл /etc/rc.d/rc.local выполняется после скрипта rc. В него рекомендуется помещать дополнительные сервисы или персональные настройки. Однако обычный пользователь редко использует эту возможность.

Другие файлы, влияющие на процесс загрузки

Все файлы конфигурации, задействованные при загрузке системы, находятся в каталоге /etc:

- fstab — содержит информацию об автоматически монтируемых при старте файловых системах;
- skel — образцы файлов конфигурации, используются при создании учетных записей новых пользователей;
- bashrc — общесистемный файл для командной оболочки;
- initscript — файл, позволяющий задать специфические действия для каждой команды из файла /etc/inittab (подробную информацию следует искать в справочной системе).

Второстепенные файлы конфигурации:

- /etc/issue — сообщение, выдаваемое системой до регистрации пользователя (до приглашения "login:");
- /etc/motd — сообщение, выдаваемое системой после регистрации пользователя.

Средства тестирования сети и сетевых настроек

Для проверки правильности настройки сети и корректности функционирования необходимо использовать набор утилит, описанных в этом разделе. Данные утилиты — простые и маленькие программы, однако большинство стандартных проблем можно выявить с их помощью. Частично мы с ними уже встречались ранее, но, тем не менее, я еще раз позволю себе обратить на них ваше внимание.

Утилита ifconfig

Утилита может показать состояние сетевых интерфейсов. Пример команды `ifconfig` выполненной на компьютере, неподключенном в данный момент к сети:

```
eth0 Link encap:Ethernet HWaddr 00:07:E9:46:A7:A2
  inet addr:192.168.0.1 Bcast:192.168.0.255 Mask:255.255.255.0
  UP BROADCAST MULTICAST MTU:1500 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:100
  RX bytes:0 (0.0 b) TX bytes:126 (126.0 b)
  Interrupt:11 Base address:0xbc00 Memory:feafd000-feafd038

lo  Link encap:Local Loopback
  inet addr:127.0.0.1 Mask:255.0.0.0
  UP LOOPBACK RUNNING MTU:16436 Metric:1
  RX packets:10 errors:0 dropped:0 overruns:0 frame:0
  TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:0
  RX bytes:700 (700.0 b) TX bytes:700 (700.0 b)
```

Из вышеприведенного примера видно, что компьютер имеет один сетевой интерфейс (eth0) и один виртуальный (loop back — lo). Из выявленной информации можно получить VFC-адрес, IP-адрес, маску сети, размер пакета, информацию по принятым и переданным данным и т. д. За более подробной информацией обращайтесь к справочной странице.

Утилита hostname

Утилита отображает или устанавливает символическое имя хоста.

При запуске без параметров программа выдает имя хоста:

```
hostname
HOME
```

Утилита ping

Программа ping позволяет проверить наличие доступа к другому хосту сети. Она посылает на указанный хост запрос и ожидает отклика. Если ответ приходит, то результаты запроса выводятся на консоль, один за другим. Запросы посылаются непрерывно до тех пор, пока пользователь не остановит утилиту нажатием клавиш <Ctrl>+<C>. Если ping не может связаться с указанной машиной, она выдает сообщение о том, что машина недоступна.

С помощью ping можно проверить как IP-настройки интерфейса (ping 192.168.0.33), так и функционирование/настройки DNS (ping main.test.com)

Если в первом случае соединение происходит нормально, а во втором возникают проблемы, то у вас неправильно указан в настройках адрес DNS-сервера, либо DNS-сервер не функционирует или недоступен (например, DNS-сервер находится за пределами локальной сети, а у вас не определен маршрут по умолчанию для интерфейса).

Также эта утилита может приблизительно показать качество соединения, поскольку она отображает количество посланных/принятых/потерянных пакетов.

Утилита tracetoute

Утилита позволяет отобразить прохождение IP-пакетов от хоста до точки назначения. Она строит таблицу, в которой может отображаться имя либо адрес промежуточного хоста, задержка в прохождении пакета в миллисекундах и другая дополнительная информация.

Утилита arp

С помощью этой утилиты можно посмотреть содержимое ARP-таблицы.

Для просмотра таблицы достаточно запустить команду `arp` с ключом `-v`.

Утилита netstat

Утилита netstat позволяет получить в режиме реального времени информацию о состоянии сетевых соединений, а также статистические данные и таблицу маршрутизации. У этой программы много опций, с помощью которых можно задавать вид получаемой информации (табл. 2.4).

Таблица 2.4. *Опции программы netstat*

Опция	Описание
`-a`	Выдать информацию обо всех сокетах включая сокеты, работающие только на прием

Таблица 2.4 *(окончание)*

Опция	Описание
-i	Выдать статистическую информацию обо всех сетевых устройствах
-c	Непрерывно выдавать информацию о состоянии сети до тех пор, пока работа программы не будет прервана
-n	Выдать IP-адреса удаленной и локальной системы
-o	Выдать информацию о количестве повторно переданных байтов и состоянии таймера (on/off)
-r	Выдать хранящуюся в ядре таблицу маршрутизации
-t	Выдать информацию только о TCP-соединениях, включая работающие на прием
-u	Выдать информацию только об UDP-соединениях
-v	Выдать информацию о версии netstat
-w	Выдать информацию только о RAW-сокетах
-x	Выдать информацию о доменных сокетах типа UNIX

Запуск netstat без опций позволяет увидеть список сетевых соединений данной системы. Сначала перечисляются активные TCP-соединения, а затем активные сокеты домена типа UNIX. Гнезда (порты) этого домена заняты процессами, обеспечивающими установление соединения данной системы с другими системами.

Утилита TCPdump

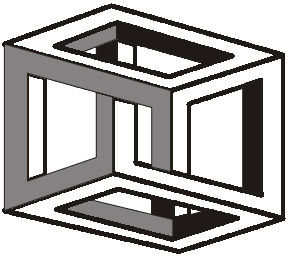
TCPdump — утилита, которая позволяет перехватывать и записывать пакеты, проходящие через сетевой интерфейс. С помощью знаний о структуре TCP/IP-пакетов и этой утилиты можно получить информацию, которая позволит определить проблемы с сетью, некорректное функционирование программ и т. д. Однако утилита может перехватывать все пакеты, полученные интерфейсом, поэтому она может быть использована в неблаговидных целях.

Литература и ссылки

- Костромин В. А. Linux для пользователя. Самоучитель. — БХВ-Петербург, 2002.
- Справочные страницы man — init, inittab, telinit, initscript.
- Ethernet-HOWTO — различные тонкости настройки сетевых адаптеров.

- Linux NET-3-HOWTO.
- **gazette.linux.ru.net/lg86/vinayak.html** — Исследование TCP/IP с помощью TCPdump и Tethereal. Автор: Vinayak Hegde. Перевод Ивана Песина.
- **msk.nestor.minsk.by/kg/2003/05/kg30503.html** — Компьютерная газета Linux в Сети. Сага третья. Подготовил X-Stranger.
- **msk.nestor.minsk.by/kg/2003/06/kg30604.html** — Компьютерная газета Linux в Сети. Сага четвертая. Подготовил X-Stranger.
- **www.citforum.ru/nets/semenov/5/dia_5.shtml** — Диагностика локальных сетей и Интернет. Семенов Ю. А. (ГНЦ ИТЭФ).
- **www.infocity.kiev.ua/inet/content/inet091.phtml** — Протоколы SLIP/CSLIP и PPP.
- **www.protocols.ru** — Энциклопедия сетевых протоколов.
- **www.osp.ru/os/2001/02/073.htm** — Облаков И. Восход солнца вручную.

ГЛАВА 3



Настройка модемного соединения

В этой главе мы обсудим организацию модемного соединения: подключение к серверу, организацию сервера, организацию Callback-доступа. Также рассмотрим организацию доступа через модемное соединение небольшой локальной сети.

Начальные установки

Как правило, во всех современных дистрибутивах Linux ядро собрано так, что оно работает как маршрутизатор пакетов между разными сетями и поддерживает механизм защиты маршрутизируемых пакетов и подсчет статистики.

Однако перед началом настройки системы необходимо убедиться, что в ядре вашей операционной системы присутствуют следующие элементы:

- Networking support (поддержка сетевых свойств);
- TCP/IP networking (поддержка TCP/IP);
- IP forwarding/gatewaying (поддержка IP-маршрутизации);
- IP multicasting (поддержка специфических свойств IP-протокола);
- IP firewalling (поддержка firewall);
- IP accounting (поддержка управления IP);
- Network device support (поддержка сетевых устройств).

Так же ядро операционной системы должно уметь работать с сетевыми картами, установленными на вашем компьютере и поддерживать протокол PPP.

Последовательный порт и модем должны быть правильно сконфигурированы.

Настройка модема и последовательного порта

В большинстве случаев ядро операционной системы настраивает последовательные порты при загрузке, и вмешательства от администратора практически не требуется. Однако если у вас старое или нестандартное оборудование, придется заниматься конфигурированием.

Символические имена последовательных портов традиционно разделялись на два типа:

- cuaXX — устаревшее наименование, использовалось для исходящих звонков, практически не применяется;
- ttySXX — первоначально обозначал текстовые терминалы и входящие звонки, теперь используется для именования последовательных портов вне зависимости от направленности потока информации.

В стандартной модели персонального компьютера последовательных портов может быть 4 штуки: COM1—COM4 в терминах DOS и ttyS0—ttyS3 в терминах Linux. За портами ttyS0 и ttyS2 закреплено аппаратное прерывание IRQ3, а за ttyS1 и ttyS3 — аппаратное прерывание IRQ4. Кроме этого, за каждым портом закреплен диапазон адресов портов ввода/вывода, для организации приема/передачи и управления обменом данными.

Помимо стандартных последовательных портов можно установить в систему специальные платы, на которых может присутствовать до 32 последовательных портов. Конфигурирование таких плат не всегда тривиальная задача, но, как правило, этот процесс хорошо описан в документации, поставляемой в комплекте с устройством.

Что касается обычных последовательных портов, то для их конфигурации используется программа setserial. С помощью этой программы можно последовательному порту назначить скорость ввода/вывода информации, количество бит в посылке, количество стоповых битов, наличие битов четности, назначить диапазон адресов портов и изменить IRQ. Однако setserial нигде не сохраняет настройки портов, поэтому, если вам необходимо использовать нестандартные настройки последовательного порта, необходимо прописать вызов setserial в конфигурационных скриптах, выполняемых при старте системы, например, rc.local.

Для правильного конфигурирования модема необходимо воспользоваться документацией, идущей в комплекте, и какой-нибудь терминальной программой. В большинстве случаев для локализированных версий модемов (например, IDC, GVC, Zyxel, USRobotics) достаточно установить заводские установки и подправить один-два параметра для учета "национальных осо-

бенностей" (время ожидания ответа станции, тип набора: импульсный или тоновый, возможно, максимальная скорость установки соединения).

Связь с провайдером

Для подключения локальной сети к Интернету при помощи модема обычно используют два варианта. Первый из них предназначен для тех, кто платит за трафик, а второй используется теми, кто оплачивает проведенное в Интернете время.

В первом случае выход в Интернет осуществляется при помощи стандартного для Linux набора программ — pppd, chat и, возможно, еще нескольких дополнительных скриптов. Происходит это следующим образом — вначале маршрутизатор дозванивается до провайдера и устанавливает с ним связь по протоколу PPP. После установления соединения полученным каналом может пользоваться любой компьютер в вашей локальной сети. Канал удерживается до тех пор, пока не выключится ваш маршрутизатор или администратор явным образом не разорвет соединение.

Второй вариант — модификация первого, в англоязычной литературе он носит название dial on demand (звонок по требованию). Для его организации дополнительно используется программа diald, с помощью которой можно организовать работу таким образом, что если в течение заранее обусловленного времени не происходит обмена данными между локальной сетью и Интернетом, то diald разрывает соединение. При первой же попытке пользователя подключиться к Интернету diald снова дозванивается и устанавливает связь.

Схема организации подключения локальной сети

Далее приведены требования, которым должно удовлетворять подключение локальной сети к Интернету.

- Возможность доступа в Интернет — модем, телефонный номер и подключение к провайдеру.
- Набор программ для организации связи — pppd, chat и diald.
- Средство для управления брандмауэром — утилита ipchains или iptables.
- Программное обеспечение для организации Proxy-сервера.
- Программное обеспечение для учета и просмотра статистики.

Теперь, когда цели и средства известны, можно приступать к настройке программ.

Организация связи по коммутируемому соединению

Старейший вариант соединения с провайдером, и, к сожалению, наиболее распространенный в нашей стране. По сравнению с организацией связи по выделенному каналу представляет собой схему более сложную, поэтому рассмотрим ее первой.

Настройка программ

Будем считать, что на компьютере, который должен выходить в Интернет, правильно настроены сетевые параметры, и вы убедились в работоспособности локальной сети.

Настройка связи с провайдером

Настроим подсистему дозвона и соединения с провайдером. Для удобства разобьем работу на два этапа.

1. Настройка PPP-соединения.
2. Установка и конфигурирование демона дозвона по требованию (diald).

Для организации связи между провайдером и клиентом необходимо получить данные, представленные в табл. 3.1.

Таблица 3.1. *Необходимые данные для настройки модемного соединения*

Необходимые данные	Значения в примере
Имя пользователя (login)	myname
Пароль пользователя (password)	test
IP-адрес пользователя (если есть)	192.168.110.100
IP-адрес сервера DNS	192.168.10.1

Процесс установления связи между вами и провайдером состоит из следующих этапов:

- соединения с компьютером провайдера с помощью модема;
- регистрации пользователя в удаленной системе;
- установки PPP-соединения.

Для решения этих задач в Linux используется небольшой набор скриптов, каждый из которых выполняет какую-то небольшую функцию. А поскольку это набор скриптов — никто не мешает на их базе определить именно те

действия, которые необходимы вам при установлении или обрыве PPP-соединения.

Размещение скриптов зависит от настройки и предпочтений вашего дистрибутива. В современных версиях дистрибутива Red Hat используется два места — каталог /etc/ppp и /etc/sysconfig/network-scripts. Наименования скриптов так же могут быть произвольными и очень часто зависят от предпочтений сборщика дистрибутива или системного администратора.

Для нашего случая будем считать, что у нас есть следующие файлы:

- /etc/ppp/chap-secrets — этот файл используется для аутентификации пользователя провайдером по протоколу chap. Обычно содержит имя и пароль пользователя для входа к провайдеру. В нашем случае это будет выглядеть так:

  ```
  myname * test
  ```

- /etc/ppp/pap-secrets — этот файл используется для аутентификации пользователя провайдером по протоколу pap. Обычно содержит имя и пароль пользователя для входа к провайдеру. В нашем случае это будет выглядеть следующим образом:

  ```
  myname * test
  ```

- /etc/ppp/ip-up — данный скрипт используется для соединения с провайдером. Зачастую этот файл содержит только следующую строку:

  ```
  /usr/sbin/pppd
  ```

  Здесь можно настроить установление модемом соединения с провайдером либо вызвать необходимый скрипт или программу;

- /etc/ppp/ip-down — этот файл используется для разрыва соединения с провайдером;

- /etc/ppp/options — это самый сложный и ответственный файл. Он определяет параметры нашего модема, скорость передачи по последовательному интерфейсу данных, настройки программы pppd и некоторые другие параметры. Обычно файл /etc/ppp/options оставляют неизменным, а для конфигурирования параметров соединения создают копию файла с именем /etc/ppp/options.ttySX, где ttySX — имя последовательного порта, к которому подключен наш модем. Пусть для определенности модем будет подключен к ttyS0 (COM1).

  ```
  # Устройство
  /dev/ttyS0
  # Скорость
  115200
  mru 1500
  # наш интерфейс : удаленный интерфейс
  192.168.110.100:192.168.110.101
  ```

```
# маска подсети
netmask 255.255.255.0
bsdcomp 0
chap-interval 15
debug
crtscts
defaultroute
```

Первые две строки определяют последовательный порт, к которому подключен наш модем, и скорость, на которой будет происходить обмен между модемом и последовательным портом. Далее — обратите внимание на строку со следующим содержимым:

```
192.168.110.100:192.168.110.101
```

Эта строка определяет IP-адреса нашего последовательного интерфейса и провайдера. Такую строку необходимо добавить, если провайдер выдал нам постоянный IP-адрес. Как правило, в современном мире с коммутируемыми соединениями такого не происходит. Для статического IP-адреса также необходимо задать маску подсети.

Поскольку наш компьютер является маршрутизатором для локальной сети, необходимо настроить маршрутизацию. Для этого воспользуйтесь программой route и документацией по ней. В случае если у вас одно подключение к провайдеру (а мы предположили, что точка подключения к провайдеру у нас одна), можно в конец файла вписать команду `defaultroute`, что позволит вам добавить маршрут в системную таблицу маршрутизации, используя удаленную сторону как шлюз.

Команды pppd

Далее мы рассмотрим основные команды программы pppd (табл. 3.2).

Таблица 3.2. *Основные команды программы pppd*

Команда	Описание
`asyncmap 0`	Async-карта символов — 32-bit hex; каждый бит — символ, который надо представить в виде Escape-последовательности, чтобы pppd мог его принять
`auth`	Требует от удаленной стороны назвать себя перед тем, как начнется обмен пакетами
`bsdcomp 0`	Определяет использование сжатия передаваемого трафика. На обычном модемном соединении не используется, позволяет в некоторых случаях почти в два раза увеличить количество передаваемых данных за единицу времени

Таблица 3.2 *(продолжение)*

Команда	Описание
chap-interval <интервал>	Определяет, что pppd будет заново вызывать удаленную сторону каждые <интервал> секунд
chap-restart <интервал>	Устанавливает интервал рестарта CHAP (пауза возобновления передач challenges) в <интервал> секунд
chap-max-challenge <значение>	Устанавливает максимальное число передач CHAP challenge
connect <программа>	Определяет программу для установки соединения
crtscts	Предписывает использовать аппаратное управление потоком данных для управления потоком данных на последовательном порте
debug	Предписывает увеличить уровень отладки. Если эта опция есть, pppd будет записывать в журнал все прибывшие и отправленные пакеты в понятной для человека форме. Пакеты регистрируются в LOG-файлах через syslog. Эта информация может быть перенаправлена в файл соответствующей установкой /etc/syslog.conf
disconnect <программа>	Предписывает запустить данную программу после того, как pppd завершил связь
domain <имя_домена>	Добавляет имя домена к имени машины
ipcp-max-configure <значение>	Устанавливает максимальное число передач IPCP configure-request
ipcp-max-terminate <значение>	Устанавливает максимальное число передач IPCP terminate-request
ipcp-max-failure <значение>	Устанавливает максимальное число IPCP configure-NAK, возвращенных перед началом отправки вместо configure-Rejects
ipcp-restart <интервал>	Устанавливает интервал перезапуска IPCP в <интервал> секунд
local	Предписывает не использовать линии управления модемом
lock	Предписывает, что pppd должен использовать lock в стиле UUCP для последовательного устройства
login	Предписывает использовать базу данных паролей для идентификации удаленной стороны
modem	Предписывает использовать линии управления модемом

Таблица 3.2 *(продолжение)*

Команда	Описание
`mru` *<число>*	Устанавливает значение MRU (Maximum Receive Unit, максимально принимаемый пакет) в *<число>*. При договоренности pppd запросит удаленную сторону отправлять пакеты не более чем по *<число>* байт. Минимальное значение MRU — 128. Значение MRU по умолчанию — 1500. Для медленных соединений рекомендуется 296 (40 байтов для заголовка TCP/IP плюс 256 байтов данных)
`mtu` *<число>*	Устанавливает значение MTU (Maximum Transmit Unit-максимально передаваемый пакет) в *<число>*. 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 *число* байт через сетевой интерфейс PPP
`name` *<имя_машины>*	Устанавливает имя машины (для аутентификации)
`noauth`	Не требует удаленную сторону назвать себя перед тем, как начнется обмен пакетами
`noipdefalut`	Запрещает поведение по умолчанию (когда не указан локальный IP-адрес), которое определяет локальный IP-адрес по имени хоста. С этой опцией удаленная сторона должна обеспечить локальный IP-адрес в течение IPCP-переговоров (если она не определена явно в командной строке или в файле options)
`pap-restart` *<интервал>*	Устанавливает интервал возобновления передачи PAP в *<интервал>* секунд
`pap-max-authreq` *<значение>*	Устанавливает максимальное число передач PAP authenticate-request (запросов на аутентификацию по протоколу PAP)
`passive`	Разрешить опцию "passive" в LCP. С этой опцией pppd будет пытаться инициировать соединение, а если ответ от другой стороны не принят, то pppd будет пассивно ожидать правильный LCP-пакет от другой стороны вместо выхода
`silent`	С этой опцией pppd не будет передавать LCP-пакеты для инициации соединения, пока не придет правильный LCP-пакет от другой стороны
`user` *<имя>*	Устанавливает имя пользователя для аутентификации этой машины на другой стороне, используя PAP. Нельзя использовать вместе с `name`
`xonxoff`	Предписывает использовать программное управление потоком данных для управления потоком данных на последовательном порте

Таблица 3.2 *(окончание)*

Команда	Описание
`+chap`	Двусторонняя CHAP-аутентификация
`+pap`	Двусторонняя PAP-аутентификация
`-all`	Не разрешает договариваться о любых опциях LCP и IPCP
`-am`	Запрещает договариваться о `asyncmap`
`-chap`	Предписывает отказаться от CHAP-аутентификации
`-d`	Устанавливает уровень отладки. Если эта опция есть, pppd будет записывать в журнал все прибывшие и отправленные пакеты в понятной для человека форме. Пакеты регистрируются в LOG-файлах через syslog. Эта информация может быть перенаправлена в файл соответствующей установкой /etc/syslog.conf
`-detach`	Предписывает не переходить в фоновый режим
`-ip`	Предписывает не договариваться об IP-адресе
`-mru`	Запрещает договариваться о MRU
`-pap`	Предписывает отказаться от PAP-аутентификации
`-pc`	Запрещает сжатие полей протокола

Как видите, параметров много и для полного понимания вопроса необходимо изучить документацию ppp.

Настройка diald

Обычно программа diald входит в стандартный дистрибутив, и установка ее с помощью менеджера пакетов rpm занимает совсем немного времени. После установки нужно привести стандартную конфигурацию программы diald в соответствие с нашими реалиями.

Расскажем немного о принципе работы программы diald, чтобы лучше понять то, что мы будем делать дальше. Программа создает соединение на псевдотерминале и устанавливает маршрутизацию на получившийся интерфейс. После этого она начинает отслеживать пакеты, проходящие по виртуальному каналу. Если кто-то пытается выйти в Интернет, diald перехватывает данные, анализирует их и на основе правил, определяемых администратором, присваивает им определенные тайм-ауты. Далее пакеты отправляются по назначению, а тайм-ауты заносятся в так называемый *набор соединения*. Как только в наборе появляется первый тайм-аут, diald начинает дозваниваться до провайдера и пытается установить соединение. Организовав сеанс

связи, демон переустанавливает маршрутизацию на реальный канал. Таким образом, связь с внешним миром оказывается установленной.

На протяжении всего времени соединения продолжает обновляться набор соединения. Истекшие тайм-ауты удаляются, новые поступают. И так продолжается до тех пор, пока по какой-либо причине трафик не прекратится. Тайм-аутов в наборе становится все меньше и меньше, и когда последний из них оканчивается, diald разрывает связь.

Теперь перейдем непосредственно к конфигурированию. Этот процесс состоит из трех частей:

- создание скрипта соединения — файл /etc/diald/connect;
- настройка основной конфигурации — файл /etc/diald.conf;
- настройка правил тайм-аутов — файл /etc/diald/standard.filter.

Создание скрипта соединения: /etc/diald/connect

Для организации сеанса связи необходимо выполнить несколько действий: дозвониться по телефону до поставщика услуг, пройти процедуру авторизации и запустить PPP-соединение. Поскольку у разных провайдеров этот процесс может коренным образом отличаться, то не имеет смысла встраивать эту процедуру в программу. Вместо этого используется внешний скрипт, описывающий процедуру соединения. Для чего достаточно подправить тот скрипт, который входит в стандартную поставку diald.

В листинге 3.1 приведен вариант файла /etc/diald/connect.

Листинг 3.1. Файл /etc/diald/connect

```
#!/bin/sh

NIT="ATZ"    # Строка инициализации модема
PHONE="2223322"    # Телефон провайдера
ACCOUNT="myname"    # логин
PASSWORD="test"    # пароль
# Определяем функцию для посылки
# сообщений в системный журнал
# и в FIFO-канал diald
function message ()
 {
   [ $FIFO ] && echo "message $*" >$FIFO
   logger -p local2.info -t connect "$*"
 }
# Начинаем процедуру связи
# Инициализируем модем
message "*** Initializing Modem ***"
```

```
chat "" $INIT OK ""
if [ $? != 0 ]
   then
      message "!!! Failed to initialize modem !!!"
      exit 1
fi
# Пытаемся дозвониться
message "*** Dialing system  ***"
chat \
      ABORT "NO CARRIER" \
      ABORT BUSY \
      ABORT "NO DIALTONE" \
      ABORT ERROR \
      "" ATDT$PHONE \
      CONNECT ""
case $? in
      0) message "*** Connected ***";;
      1) message "!!! Chat Error !!!"; exit 1;;
      2) message "!!! Chat Script Error !!!"; exit 1;;
      3) message "!!! Chat Timeout !!!"; exit 1;;
      4) message "!!! No Carrier !!!"; exit 1;;
      5) message "!!! Busy !!!"; exit 1;;
      6) message "!!! No DialTone !!!"; exit 1;;
      7) message "!!! Modem Error !!!"; exit 1;;
      *) esac
# Проходим авторизацию
message "*** Send login and password ***"
chat \
      login: $ACCOUNT \
      password: $PASSWORD        TIMEOUT 5 ""
if [ $? != 0 ] then
      message "!!! Failed to send !!!"
      exit 1
fi
# Все прошло удачно!
message "*** Protocol started *** "
```

Приведенный скрипт — просто сценарий на языке командной оболочки, который вам необходимо немного адаптировать для ваших параметров.

Настройка основной конфигурации: /etc/diald.conf

/etc/diald.conf — основной конфигурационный файл программы diald, в котором задаются параметры устанавливаемого соединения и определяется

поведение программы. Набор команд конфигурации у diald достаточно обширен, поэтому в приведенном примере будут использованы только необходимые, а подробную информацию по конфигурационным командам можно посмотреть в документации к программе diald.

Приведем содержимое файла diald.conf (листинг 3.2).

Листинг 3.2. Файл diald.conf

```
# Протокол для связи с провайдером
mode ppp
# Вести журнал сеансов связи diald.log
accounting-log /var/log/diald.log
# Для управления демоном из внешних программ
# организовать канал FIFO — diald.ctl.
fifo /etc/diald/diald.ctl
# Для дозвона использовать файл /etc/diald/connect
connect /etc/diald/connect
# Далее несколько команд, описывающих применяемый модем.
# Поскольку мы уже определили параметры в /etc/ppp/options,
# то приведенные далее команды необходимо закомментировать во избежание
# конфликтов в файле /etc/ppp/options
#device /dev/modem
#speed 115200
#modem
#lock
#crtscts
# Назначаем локальный и удаленный адреса нашего
# соединения. Если при связи с провайдером IP-адрес
# для вас выделяется динамически, то здесь можно
# поставить любые свободные адреса из диапазона,
# оговоренного при настройке нашей TCP/IP-сети.
# При запуске PPP diald сам выставит корректные значения
local 192.168.110.100
remote 192.168.110.101
# Провайдер дает нам динамический IP
dynamic
# Установить маршрут по умолчанию
# на виртуальное соединение
defaultroute
# Максимальное количество неудачных попыток дозвона
dial-fail-limit 10
# Задержка между попытками дозвона
redial-timeout 5
```

```
# Время ожидания завершения скрипта connect
connect-timeout 120
# Файл с правилами для тайм-аутов
include /etc/diald/standard.filter
```

Настройка правил тайм-аутов: /etc/diald/standard.filter

Следующее, что вы должны сделать, — произвести настройку правил тайм-аутов. Это самый сложный момент настройки diald, т. к. требует знания внутренней структуры IP-пакетов. Однако разработчики diald позаботились о пользователях, и стандартный файл standard.filter имеет вполне приемлемые для большинства случаев настройки. Оставив в нем все как есть, мы получим набор правил, рассчитанный на трехминутную паузу между окончанием активности в Интернете и разрывом связи с провайдером.

Комплексное тестирование

После проделанных манипуляций настало время проверить — правильно ли настроены наши программы. Для этого на компьютере желательно временно отключить все настройки брандмауэра (если вы, конечно, установили его). Затем необходимо запустить программу diald и попытаться выйти в "большой мир". Можно использовать браузер lynx (и зайти, например, на сайт **http://www.bhv.ru**), можно — программу ping.

Если все было настроено корректно, то после ввода предыдущей команды модем должен начать дозваниваться до провайдера. Через некоторое время связь будет установлена. Однако практически всегда lynx выдает сообщение о том, что не может соединиться с удаленным сервером! В данном случае — это нормальное явление. Дело в том, что при PPP-соединении с динамическими IP-адресами в силу определенных особенностей первый пакет обычно бывает утерян и не доходит до адресата. В результате мы ждем ответа от сервера, а он об этом и не подозревает. Достаточно повторить введенную ранее команду, чтобы все заработало.

Далее нам необходимо убедиться, что модем аккуратно разорвет соединение по прошествии трех минут. Дождавшись конца загрузки Web-страницы, засечем время. Примерно через три минуты diald должен разорвать соединения.

Если у вас все прошло именно таким образом, значит, система работает как надо. В противном случае проанализируйте последние строки системного журнала (/var/log/messages).

Указанными действиями мы проверили корректную работу только с нашего компьютера-маршрутизатора. Однако нам надо сделать то же самое и с любого компьютера в локальной сети, поэтому попробуем повторить описанную процедуру на любом компьютере. Реакция diald должна быть аналогич-

ной. Если что-то пошло не так, проверьте корректность настройки протокола TCP/IP на этой машине, в частности — настройки сетевого шлюза, которые должны указывать на наш компьютер-маршрутизатор.

Настройка сервера входящих звонков (DialIn)

В этом разделе мы настроим наш сервер для приема входящих звонков

Мы уже умеем настраивать систему таким образом, чтобы она выступала в роли шлюза для вашей локальной сети. Но нам часто необходимо получить доступ к локальной сети организации, например из дома, в том числе также из дома выйти в Интернет через корпоративную локальную сеть. Для этого нужно установить программу, которая умеет "поднимать трубку" модема по входящему звонку и совершать некоторые дополнительные действия. Одной из таких программ является mgetty, умеющая помимо всего прочего посылать и принимать факсы, а также с помощью голосового модема принимать и отправлять голосовую почту (voice mail).

Настройка mgetty

Обычно mgetty, как и ppp, входит в стандартную поставку дистрибутива. Единственное, что необходимо проверить, был ли пакет mgetty скомпилирован с опцией `-DAUTO_PPP`, и если нет, то пакет следует перекомпилировать с этой опцией (в дистрибутиве Red Hat mgetty скомпилирован с нужной нам опцией).

После установки mgetty нам следует отредактировать конфигурационные файлы.

В файле /etc/mgetty+sendfax/login.config мы должны написать следующее:

```
/AutoPPP/ — a_ppp /usr/sbin/pppd auth refuse-chap require-pap login
- — /bin/login @
```

Эта строка говорит mgetty:

- после установления входного соединения необходимо вызвать программу pppd;
- для пользователя требуется авторизация;
- аутентификацию по протоколу CHAP отклонять и требовать авторизации по протоколу PAP.

После установления соединения mgetty анализирует данные, приходящие с модема, и в случае, когда приходит запрос на авторизацию по протоколу PAP, программа сразу же запускает pppd, который и проводит аутентификацию по протоколу PAP.

Далее, нам необходимо отредактировать файл /etc/mgetty+sendfax/mgetty.config приблизительно следующим образом:

```
port ttyS1
speed 115200
data-only y
debug 3
init-chat "" ATZ OK
     answer-chat "" ATA CONNECT \c \r
```

Как видите, модем подключен ко второму последовательному порту, скорость обмена 115 200, строка инициализации `ATZ`.

Далее нужно добавить mgetty в файл inittab. Для этого достаточно дописать всего лишь одну строку:

```
S4:2345:respawn:/sbin/mgetty /dev/ttyS1
```

Перегрузив операционную систему, можно приступать к испытаниям — попробуйте позвонить на телефонный номер, где установлен ваш модем — если все настроено нормально — модем должен "поднять трубку".

Настройка pppd

С настройкой pppd вы уже ознакомились ранее. Поэтому, чтобы не повторяться, просто приведем соответствующие конфигурационные файлы с небольшими комментариями.

Файл options.ttyS1 должен содержать следующие данные:

```
# Устройство
lock
login
auth
modem
crtscts
-chap
+pap
# наш интерфейс : удаленный интерфейс
192.168.10.100:192.168.10.101
# маска подсети
netmask 255.255.255.0
# адрес сервера DNS для клиента Windows
ms-dns 192.168.10.100
```

Файл /etc/ppp/pap-secrets должен содержать следующие данные:

```
user1  сервер.домен  ""      *
user2  сервер.домен  ""  *
```

где:

- `user1` — имя пользователя, причем он должен существовать в вашей системе, где установлен модем;
- `user2` — сервер, на котором будет проводиться аутентификация, в нашем случае вместо *`сервер.домен`* необходимо поставить имя вашего компьютера, где расположен модем;
- "" — отсутствие пароля указывает на то, что пароли необходимо брать из файла /etc/shadow;
- `*` — абонент может производить аутентификацию с любого IP-адреса.

Вот и все — вы стали микропровайдером, причем сильно облегчили жизнь пользователям Windows, поскольку IP-адрес и адрес DNS-сервера вы выдаете автоматически, кроме того, отпадает потребность в использовании скрипта для соединения.

Настройка Callback-сервера

Вы настроили свой Dial-in-сервер, попользовались им какое-то время и захотели облегчить кошелек фирмы и сохранить свой в неприкосновенности. Например, в вашем городе повременная оплата и часами работать в Интернете из дома дорого, но руководство вашей организации не возражает против того, чтобы вы работали за ее счет. Дело за малым — организовать ваш Dial-in-сервер таким образом, чтобы не вы ему звонили, а он вам. В западной литературе такой сервер называется Callback-сервером (сервер обратного звонка). Функционирует он следующим образом.

Сначала клиент дозванивается через модем к Callback-серверу. Модем на сервере настроен на прием входящих звонков. После установки соединения сервер предлагает клиенту пройти аутентификацию. Клиент подключается к нему как особый Callback-пользователь. После этого модем на сервере обрывает связь и сам звонит клиенту по номеру, который закреплен за компьютером клиента (либо определяется с помощью АОНа). Модем на клиентском компьютере готов принять обратный звонок, и после установления соединения происходит повторная авторизация. По окончании аутентификации устанавливается PPP-соединение. Далее клиент работает обычным образом.

Конфигурация Callback-сервера

После того как настройка Dial-in-сервера завершена, необходимо настроить Callback. Для этого нужно выполнить следующие действия:

1. Создать нового пользователя back.
2. Создать пустой файл с именем callback.conf в /etc/mgetty/.

3. В файл /etc/mgetty/login.config добавить следующую строку:

```
back — — /usr/sbin/callback -S 1234567
```

После ключа `-S` указывается номер, по которому сервер должен сделать обратный звонок клиенту.

Конфигурация клиентов

Поскольку сервер мы уже сконфигурировали, необходимо сконфигурировать клиента и проверить, каким же образом работает Callback. Начнем с операционной системы Linux.

Конфигурирование Linux-клиента

Для конфигурирования клиента Linux необходимо выполнить следующее:

1. Создать файл /etc/ppp/options, в котором должны быть такие строки:

```
lock
defaultroute
noipdefault
modem
115200
crtscts
debug
passive
```

2. Создать файл ppp-callback в /etc/ppp/peers/, в котором должны быть следующие строки:

```
ttyS1 33600 crtscts
connect '/usr/sbin/chat -v -f /etc/ppp/chat-callback'
noauth
```

3. Создать файл /etc/ppp/chat-callback, в котором должны быть такие строки:

```
ABORT BUSY
ABORT VOICE
ABORT "NO DIALTONE"
ABORT "NO ANSWER"
"" ATZ
OK ATDP7654321            # Телефонный номер сервера
CONNECT \d\d
ogin: \q\dback
TIMEOUT 90
RING AT&C0S0=1
ogin: \q\dvasya
assword: \q\dpasswordfortest
```

В файл chat-callback необходимо вписать телефон Callback-сервера, имя и пароль пользователя.

4. Создать файл /usr/bin/pppcall, в котором должны быть такие строки:

```
#!/bin/bash
/usr/sbin/pppd -detach call ppp-callback &
```

И сделать его исполняемым.

Теперь для того чтобы позвонить на ваш сервер, достаточно запустить скрипт pppcall.

Конфигурирование клиента MS Windows

Для Windows конфигурация производится по-другому. Выберите команду меню **Пуск | Программы | Стандартные | Удаленный доступ к сети | Новое соединение**. Укажите данные, необходимые для дозвона к серверу. Помимо этого, в настройках модема на вкладке **Подключения** нажмите кнопку **Дополнительно** и в строке инициализации модема укажите следующее:

```
&c0s0=1
```

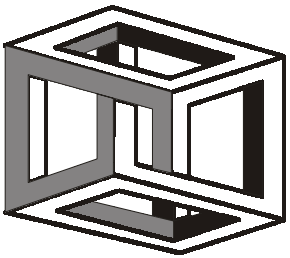
Теперь пробуем дозвониться до нашего сервера. После дозвона в открывшемся окне терминала вы увидите приглашения для аутентификации.

Зарегистрируйтесь в системе как `back`. После этого модем со стороны сервера оборвет связь, подождет несколько секунд и перезвонит вам. После установки Callback-соединения вам предложат пройти повторно авторизацию. Введите ваш нормальный логин и пароль и нажмите кнопку **Продолжить** в окне терминала.

Литература и ссылки

- **cs.uni-bonn.de/ppp/part1.html, netware.nwsoft.ru** — John Wobus. Протокол PPP. Перевод Виталия Горохова.
- **www.idir.net/~gromitkc/winmodem.html** — драйверы и настройки для Win-модемов на базе чипов различных производителей. — Lucent, Connexant (Rockwell), Pctel.
- **www.linmodems.org** — драйверы и настройки для Win-модемов на базе чипов различных производителей. — Lucent, Connexant (Rockwell), Pctel.
- **www.o2.net/~gromitkc/winmodem.html** — драйверы и настройки для Win-модемов на базе чипов различных производителей. — Lucent, Connexant (Rockwell), Pctel.
- **www.olitec.com/pci56kv2.html** — драйверы для Win-модемов на базе чипа Connexant (Rockwell).

- **www.heby.de/ltmodem/** — драйверы и настройки для Win-модемов на базе чипа Lucent.
- **www.sfu.ca/~cth/ltmodem**/ — драйверы и настройки для Win-модемов на базе чипа Lucent.
- **linux.uatel.net.ua/ppp-dialin.phtml** — Настройка PPP Dial-in-сервера (PAP-аутентификация).
- **www.softerra.ru/freeos/12279/** — Денис Колисниченко. Пошаговая настройка Dial-in-сервера.
- **www.linuxgazette.com** — Sunil Thomas Thonikuzhiyil. Настройка Callback-сервера на базе Linux. Перевод Александра Куприна.
- **www.bdcol.ee/linux/callback.shtml** — Linux-callback.
- **www.leo.org/~doering/mgetty/** — документация по Mgetty+Sendfax.
- **http://koi.citforum.tula.ru/operating_systems/articles/ppp.shtml** — Водолазкий В. Установка PPP-соединения в Linux.
- Документация pppd.
- **linux.yaroslavl.ru/Howto/Howto-mini/call-back-mini-HOWTO.html** — Callback MINI-HOWTO (русский перевод).
- PPP-HOWTO.
- **www.linux.org.ru/books/gateway/** — Костарев А. Ф. ОС Linux как мост между локальной сетью и Internet.
- **lin-omts.airport.sakhalin.ru/departs/ccito/guide1.htm** — Как установить, настроить и запустить Web-узел UNIX не тратя лишних денег, сил и здоровья.
- **www.geocities.com/SiliconValley/Pines/7895/PPP.DOC** — Водолазкий В. Установка PPP-соединения в Linux.



ЧАСТЬ II

Сетевые службы

Здесь рассматриваются вопросы конфигурирования большинства основных сетевых служб. Описывается конфигурирование DHCP, DNS, почтового сервера, службы LDAP, FTP, NNTP, Apache, Proxy-сервера и NTP-сервера. Именно данная часть позволит вам создать полнофункциональный сервер, способный выполнить около 90% задач типичного сервера небольшой организации.

Вторую часть я старался сделать доступной для понимания начинающему пользователю, однако материал требует размышлений и экспериментов от читателя, т. к. здесь содержится большой объем информации, но я надеюсь, что большинство читателей данной книги прошли хорошую школу советских ВУЗов и обилие информации в достаточно сжатом изложении их не отпугнет.

Глава 4. DHCP

Глава 5. DNS

Глава 6. Почта

Глава 7. Сетевая информационная система NIS (NIS+) и ее конфигурирование. LDAP

Глава 8. FTP

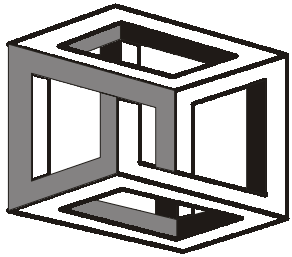
Глава 9. NNTP. Сервер новостей INN

Глава 10. Web-сервер Apache

Глава 11. Proxy-сервер

Глава 12. Синхронизация времени через сеть, настройка временной зоны

ГЛАВА 4



DHCP

Как вы знаете, без IP-адреса компьютер не может быть включен в сеть с протоколом TCP/IP. В малой сети назначить IP-адреса каждому компьютеру и прописать их в соответствующих местах не представляет особого труда. Все намного хуже, если под вашим крылом сеть из 40—50 компьютеров или даже больше. А если в вашей сети у пользователей "очумелые ручки", и установлены компьютеры с операционной системой Windows — наверняка не пройдет и недели, как вы получите вызов к "пациенту", у которого очень грамотный пользователь снес сетевые настройки, неправильно выставил DNS или начудил с IP-адресом компьютера.

Похожие проблемы возникают при подключении нового или "гостевого" компьютера, особенно если IP-адреса ранее выдавались бессистемно.

Для решения проблем с назначением IP-адресов и предназначен протокол DHCP.

DHCP-протокол

DHCP — протокол динамического конфигурирования хостов (Dynamic Host Configuration Protocol), клиент-серверный протокол, предназначен для управления сетевыми параметрами хостов. Описание протокола содержится в RFC 2131, RFC 2132, которые сменили устаревшие RFC 1531 и RFC 1541.

Архитектура и формат сообщений

Как уже ранее упоминалось, DHCP — классический клиент-серверный протокол. Клиентами выступают компьютеры сети, пытающиеся получить IP-адрес, адрес сетевого шлюза, имя хоста и другие параметры, о которых вы узнаете чуть позже. Сервер DHCP выдает в ответ на запрос клиентов назначаемые им сетевые параметры (IP-адрес, адрес шлюза), контролирует

использование IP-адресов, поддерживает пул свободных адресов и ведет собственную базу клиентов.

В роли транспортного протокола для обмена DHCP-сообщениями выступает протокол UDP. При отправке сообщения с клиента на сервер используется 67-й порт DHCP-сервера, при передаче в обратном направлении — 68-й. Эти номера портов, как и схожая структура сообщений, обеспечивают обратную совместимость протоколов DHCP с BOOTP.

Структура DHCP-пакета приведена в табл. 4.1.

Таблица 4.1. *Структура DHCP-пакета*

Название поля	Величина в байтах	Описание
op	1	Тип сообщения (1 = BOOTREQUEST (запрос), 2 = BOOTREPLY (ответ))
htype	1	Тип адреса оборудования
hlen	1	Длина адреса оборудования
hops	1	Используется ретранслирующим агентом
xid	4	Идентификатор транзакции между клиентом и сервером
secs	2	Время с момента выдачи запроса или начала обновления конфигурации
flags	2	Флаги
ciaddr	4	IP-адрес клиента
yiaddr	4	IP-адрес, предлагаемый сервером хосту в качестве клиентского
siaddr	4	IP-адрес следующего сервера, участвующего в загрузке
giaddr	4	IP-адрес ретранслирующего агента
chaddr	16	MAC-адрес клиента
sname	64	Хост-имя сервера (опционально)
file	128	Имя загрузочного файла (опционально)
options	312—576	Используется для передачи параметров конфигурации

Режимы выдачи IP-адресов

Казалось бы, раз протокол предназначен для динамической выдачи адресов, то и режим один — динамический. Но нет! Чтобы не лишать администратора гибкости при назначении IP-адресов, предусмотрено три режима: статический, динамический и ручной.

Рассмотрим их отличия:

- *Статический* — DHCP-сервер конфигурируется таким образом, что хостам назначаются неизменные со временем IP-адреса;
- *Динамический* — хосты получают IP-адреса, которые могут меняться с течением времени;
- *Ручной* — DHCP-сервер уведомляет клиента об адресе, присвоенном ему администратором сети вручную.

Как видите, первый и последний случаи достаточно тривиальны, и особо на них останавливаться не будем. Нас интересует второй случай — динамическое распределение адресов.

Итак, выдача IP-адреса в аренду производится по инициативе (запросу) клиента. DHCP-сервер гарантирует, что до истечения срока аренды этот IP-адрес не будет выдан в аренду другому клиенту. Сервер обычно настраивается таким образом, что при повторных обращениях клиента в течение определенного срока (зависит от администратора, обычно неделя-две) он старается выдать клиенту IP-адрес, использовавшийся им ранее. Клиент может (при соответствующей настройке) запросить продление сроков аренды IP-адреса либо досрочно от него отказаться.

Давайте рассмотрим процесс получения IP-адреса клиентом.

1. Клиент посылает широковещательный запрос, в котором может указываться устраивающий клиента IP-адрес и срок его аренды. Если в физическом сегменте сети клиента DHCP-сервер отсутствует, сообщение будет передано в другие сегменты сети ретранслирующими агентами протокола BOOTP
2. DHCP-сервер посылает в ответ пакет, содержащий доступный IP-адрес (поле `yiaddr`), и возможно, параметры конфигурации клиента. На этой стадии сервер не обязан резервировать указанный в поле `yiaddr` адрес, однако обязан проверить посредством ICMP то, что этот IP-адрес не используется кем-либо в сети.
3. Клиент не обязан реагировать на первое поступившее предложение. Возможен вариант, что клиент получил отклики от нескольких DHCP-серверов, выбрал понравившийся ему адрес и отправил в сеть широковещательное сообщение, в котором содержатся идентификатор выбранного DHCP-сервера и желательные значения запрашиваемых параметров конфигурации. Именно поэтому сервер не резервирует сразу IP-адрес, переданный в первом ответе сервера.
4. Сервер посылает пакет-подтверждение, в котором содержатся значения параметров конфигурации, и производит резервирование IP-адреса за клиентом. Если к моменту поступления ответа от клиента предложенный ранее адрес уже закреплен за другим клиентом, сервер сообщает клиенту о невозможности получения именно этого IP-адреса.

5. Клиент, получив сетевые параметры от DHCP-сервера, должен средствами протокола ARP убедиться в уникальности IP-адреса и зафиксировать суммарный срок его аренды. В том случае, если IP-адрес уже используется другим хостом, клиент обязан отправить серверу уведомляющее сообщение и начать всю процедуру снова не ранее чем через 10 секунд.

Параметры конфигурации (поле *options*)

В качестве параметров конфигурации DHCP-сервер может выдать клиенту, помимо IP-адреса и имени хоста, достаточно большой объем данных. Далее мы перечислим основные данные, которые может получить клиент от DHCP-сервера.

- Маска подсети
- MTU
- TTL
- Адреса COOKIE-серверов
- Адреса DNS-серверов
- Адреса FINGER-серверов
- Адреса IRC-серверов
- Адреса LOG-серверов
- Адреса LPR-серверов
- Адреса WINS-серверов
- Адреса NIS-серверов
- Адреса NNTP-серверов
- Адреса NTP-серверов
- Адреса POP-серверов
- Адреса SMTP-серверов
- Адреса TFTP-сервера
- Адреса WWW-серверов

А также большое количество второстепенных параметров, полное описание которых можно найти в документации на DHCP-сервер.

Недостатки DHCP

К недостаткам протокола прежде всего следует отнести крайне низкий уровень информационной безопасности, что обусловлено непосредственным использованием протокола UDP. Также не существует защиты от появления в сети несанкционированных DHCP-серверов, способных рассылать клиентам ошибочную или потенциально опасную информацию: некорректные или уже задействованные IP-адреса, неверные сведения о маршрутизации и т. д.

Помимо этого существует проблема согласования информационной адресной базы в службах DHCP и DNS.

И наконец, централизация процедуры назначения адресов снижает надежность системы: при отказе DHCP-сервера все его клиенты оказываются не в состоянии получить IP-адрес и другую информацию о конфигурации.

DHCP-сервер

Традиционно, начнем описание программного обеспечения с серверной части, как с наиболее трудоемкой в настройке и более ответственной. Программное обеспечение DHCP-сервера входит в практически любой современный дистрибутив. Если же вы не обнаружили его на дисках дистрибутива, можно загрузить его с сайта разработчика DHCP — Internet Software Consortium — **http://www.isc.org**.

Установка пакета не вызывает никаких сложностей. После нее нужно убедиться, что демон dhcpd будет автоматически стартовать при загрузке операционной системы.

За конфигурацию dhcpd отвечает два файла:

- /etc/dhcpd.conf;
- /var/lib/dhcp/dhcpd.leases.

Файл dhcpd.conf

В этом файле содержатся все настройки DHCP-сервера. Сначала опишем эти настройки, а после рассмотрим парочку типичных конфигурационных файлов.

Итак, файл dhcpd.conf содержит конфигурационную информацию для демона dhcpd. Он представляет собой текстовый файл ASCII.

Комментарием является строка, начинающаяся с символа #. Конфигурационные переменные состоят из двух частей: параметров и значений, заканчивающихся точкой с запятой.

Глобальные параметры, действие которых распространяется на все группы, размещаются вначале данных, до описания групп.

Замечание

Большинство параметров можно безболезненно размещать как в глобальной области, так и в группах.

Вначале файла можно определить параметры, действие которых будет распространяться на все группы данных:

- `ddns-update-style none;` — разрешает либо запрещает использование динамического обновления DNS;
- `option domain-name "test.org";` — этот параметр задает имя домена, в котором функционирует DHSP-сервер. В дальнейшем можно не указывать в переменной `host` полное имя хоста;
- `option domain-name-servers` *`имена DNS-серверов`*`;` — параметр определяет список DNS-серверов, используемых сервером DHCP при разрешении символических имен;

- `option netbios-name-servers` *`список IP-адресов`*`;` — если клиент использует протокол NetBIOS, определяет список WINS-серверов;
- `option netbios-node-type` *`цифровое значение`*`;` — определяет порядок использования параметра `netbios-name-servers`:
  - `1` — использование широковещательных запросов вместо WINS-сервера;
  - `2` — использовать только WINS-сервер;
  - `3` — сначала использовать широковещательные запросы, затем WINS-сервер;
  - `4` — сначала использовать WINS-сервер, затем широковещательные запросы;
- `option nis-domain "test.org";` — если присутствует поддержка NIS, можно задать домен подсети;
- `max-lease-time` *`секунды`*`;` — параметр определяет максимальное время аренды IP-адреса клиентом в секундах. Если за это время клиент не запросил о подтверждении аренды адреса, клиент считается отсутствующим в сети, и его IP-адрес считается свободным для аренды;
- `default-lease-time` *`секунды`*`;` — параметр определяет время по умолчанию аренды IP-адреса клиентом в секундах. Если за это время клиент не запросил о подтверждении аренды адреса, клиент считается отсутствующим в сети. Обычно данные переменные устанавливаются одинаково;
- `min-lease-time` *`секунды`*`;` — параметр определяет минимальное время аренды IP-адреса клиентом в секундах. Если за это время клиент не запросил о подтверждении аренды адреса, клиент считается отсутствующим в сети. Обычно данные переменные устанавливаются одинаково.

Следующие три параметра позволяют задать способ взаимодействия с клиентами, которые не определены в списке хостов DHCP-сервера (отсутствуют MAC-адреса):

- `allow unknown-clients;` — разрешает выдачу IP-адреса для неизвестного клиента;
- `deny unknown-clients;` — отклоняет выдачу IP-адреса для неизвестного клиента;
- `ignore unknown-clients;` — игнорирует запросы неизвестного клиента на получение IP-адреса.

Следующие параметры позволяют определить стратегию сервера по взаимодействию с клиентами, посылающими запрос по bootp-протоколу:

- `allow bootp;` — разрешить получение IP-адреса по протоколу bootp;
- `deny bootp;` — отклонять запросы по протоколу bootp;
- `ignore bootp;` — игнорировать запросы по протоколу bootp.

Логически связанные параметры могут находиться между фигурными скобками `{}`, перед ними ставят конфигурационную переменную, к которой применяются эти параметры (листинг 4.1).

В качестве таких переменных могут выступать:

- `subnet`;
- `group`;
- `host`.

Листинг 4.1. Пример использования `{}`

```
subnet 204.254.239.64 netmask 255.255.255.224 {
           параметры подсети ...
           range 204.254.239.74 204.254.239.94;
 }
group {
           параметры группы ...
           host vasya.test.org {
           параметры хоста ...
            }
           host petya.test.org {
               параметры хоста ...
            }
 }
```

где:

- `subnet` *<IP-адрес>* `netmask` *<маска сети>* — этот параметр определяет адрес подсети и маску, для которой сервер DHCP будет выдавать динамические IP-адреса. В конфигурационном файле может быть несколько записей `subnet`:
  - `range` *<IP-адрес начала диапазона>* *<IP-адрес конца диапазона>* — параметр задает диапазон IP-адресов, из которых сервер для данной подсети может выдавать адреса. Параметр `range` необязательный, при его отсутствии диапазон динамически выдаваемых IP-адресов определяется исходя из подсети и ее маски;
  - `option domain-name "test.org";` — этот параметр задает имя домена, в котором функционирует DHSP-сервер. В дальнейшем можно не указывать в переменной `host` полное имя хоста;
  - `option nis-domain "test.org";` — если присутствует поддержка NIS, можно задать домен подсети;
  - `option routers` *<IP-адрес>*`;` — параметр содержит список IP-адресов маршрутизаторов;

- `option subnet-mask <маска подсети>;` — параметр позволяет задать маску подсети;
- `option domain-name-servers <имена DNS-серверов>;` — параметр определяет список DNS-серверов, используемых сервером DHCP при разрешении символических имен;
- `range dynamic-bootp <IP-адрес начала диапазона> <IP-адрес конца диапазона>;` — для клиентов, которые производят загрузку по протоколу bootp *(см. главу 21)* — этот параметр задает диапазон адресов, откуда берутся IP-адреса;
- `option broadcast-address <IP-адрес>;` — параметр определяет адрес для посылки широковещательных сообщений;

❒ `group` — параметр, с помощью которого для некоторого набора хостов можно задавать некоторые общие параметры. Эти параметры уже упоминались в описании глобальных параметров и в описании подсети;

❒ `host <имя хоста>` — этот параметр позволяет задать для хоста *имя хоста* и некоторые параметры, специфичные именно для него:

- `hardware ethernet xx:xx:xx:xx:xx:xx;` — параметр определяет тип сетевого интерфейса и его MAC-адрес, который записывается в виде *xx*:*xx*:*xx*:*xx*:*xx*:*xx* (для Ethernet-карты), где *xx* — восьмибитовое число в шестадцатеричном представлении;
- `fixed-address <IP-адрес>;` — параметр позволяет зафиксировать IP-адрес за определенным хостом;
- `filename "filename";` — параметр задает имя файла, который загружает хост после получения IP-адреса. Используется для загрузки бездисковых клиентов.

Помимо этого существует еще десятка два конфигурационных параметров, однако они достаточно специфичны, и в большинстве случаев не используются. Для ознакомления с ними рекомендуем воспользоваться списком литературы, приведенным в конце главы.

Файл dhcpd.leases

Файл dhcpd.leases представляет собой базу данных, в которой хранятся записи о клиентах и арендованных ими IP-адресах. Запись представляет собой несколько строк следующего вида:

```
lease 192.168.10.27 {
        starts 5 2003/06/20 09:14:54;
        ends 5 2003/06/27 09:14:54;
        hardware ethernet 00:60:67:75:40:37;
        uid 01:00:60:67:75:40:37;
        client-hostname "Oskar";
}
```

где:

- `lease 192.168.10.27` — показывает, какой IP-адрес взят в аренду;
- `starts 5 2003/06/20 09:14:54;` — начало срока аренды (в данном случае 20 июня 2003 года в 9 часов 14 минут и 54 секунды);
- `ends 5 2003/06/27 09:14:54;` — предполагаемый конец аренды (если клиент не запросит продления аренды). Легко заметить — время аренды равно 7 суток;
- `hardware ethernet 00:60:67:75:40:37;` — данный параметр показывает, что сетевой интерфейс, которому назначен IP-адрес, это Ethernet-карта с MAC-адресом 00:60:67:75:40:37;
- `uid 01:00:60:67:75:40:37;` — необязательный параметр. Идентификатор клиента основывается на протоколе ARP и для Ethernet-карты представляет собой MAC-адрес, впереди которого добавлена 1;
- `client-hostname "Oskar";` — имя хоста клиента.

Помимо этого есть еще десяток параметров, с которыми вы можете ознакомиться в справке к программе dhcpd.

Пример файла dhcpd.conf

Теперь, когда мы имеем представление о структуре файла dhcpd.conf, рассмотрим конфигурационный файл, в котором описывается простая локальная сеть со следующими характеристиками:

- адрес сети 192.168.1.0;
- маска сети 255.255.255.0;
- домен test.org;
- один DHSP-сервер с адресом 192.168.0.2;
- один DNS-сервер с адресом 192.168.0.3;
- один шлюз с адресом 192.168.0.1;
- 11 клиентов, получающих адреса, причем один из них всегда постоянен.

Листинг 4.2. Содержимое файла /usr/local/etc/dhcpd.conf

```
#global options
ddns-update-style none;
option domain-name "test.org";
option domain-name-servers 192.168.10.3;

# 7 X 24 hours – lease time
default-lease-time 604800;
```

```
max-lease-time 604800;
# my subnet
subnet 192.168.1.0 netmask 255.255.255.0 {
    range 192.168.10.5 192.168.10.20;
    option routers 192.168.1.1;
}
host vasya {
hardware ethernet 00:70:58:bc:10:03;
fixed-address 192.168.1.17;
}
```

Что же мы видим в листинге 4.2? Первая строка — комментарий, показывает, что дальше идет описание глобальных параметров. В следующей строке мы запрещаем использовать динамический DNS. Затем мы определяем имя домена и адрес DNS-сервера. Далее определяем время аренды IP-адреса клиентами в семь суток. После этого определяется наша подсеть, в описании которой мы задаем диапазон динамических IP-адресов и адрес маршрутизатора. Наконец, последняя секция — определяем хост со статическим IP-адресом с именем `vasya`.

DHCP-клиент

Установка DHCP-клиента ничем не отличается от установки DHCP-сервера. Если пакет отсутствует в вашем дистрибутиве, его можно получить на сайте **http://www.isc.org**.

После установки клиента нам необходимо произвести его конфигурацию. Для этого нужно разобраться со следующими файлами:

- etc/dhclient.conf;
- /var/lib/dhcp/dhclient.leases.

Файл dhclient.conf

Файл dhclient.conf отвечает за конфигурацию DHCP-клиента. Может содержать следующие параметры:

- `timeout <секунды>;` — время, которое клиент ожидает ответа от сервера в секундах, по умолчанию 60 секунд;
- `retry <секунды>;` — параметр определяет, сколько секунд ожидает клиент перед повторением запроса к серверу, если предыдущий запрос окончился неудачей;
- `reboot <секунды>;` — этот параметр определяет, сколько времени при перезагрузке клиент ожидает выделения того же самого IP-адреса. Если за это время DHCP-сервер не предоставляет старый IP-адрес, клиент

это время DHCP-сервер не предоставляет старый IP-адрес, клиент делает попытку получить новый IP-адрес;

- `initial-interval <секунды>;` — параметр определяет, сколько приблизительно времени пройдет между первой попыткой послать сообщение серверу и второй. При каждой попытке этот параметр увеличивается на некоторую величину;
- `select-timeout <секунды>;` — этот параметр используется в том случае, если в сети находятся несколько DHCP-серверов. Он задает время ожидания ответов от DHCP-серверов, по истечении которого клиент определяет, с каким сервером он будет сотрудничать;
- `reject <IP-адрес>;` — этот параметр указывает, с каким DHCP-сервером клиент не должен сотрудничать.

Следующие три параметра отвечают за временные настройки аренды IP-адреса. Обычно эти параметры выдают DHCP-серверы, но их можно явно указать в конфигурационном файле:

- `renew <дата>;` — определяет момент времени, когда необходимо попытаться у сервера, выдавшего IP-адрес, продлить время аренды IP-адреса;
- `rebind <дата>;` — определяет момент времени, когда клиент должен произвести попытку продлить аренду IP-адреса у любого доступного DHCP-сервера;
- `expire <дата>;` — определяет время истечения аренды IP-адреса.

В файле dhclient.conf, подобно dhcpd.conf, можно определять групповые параметры.

Например, `interface "Имя интерфейса"`. В этом групповом параметре можно задать специфичные для данного интерфейса настройки.

Давайте рассмотрим описание параметров сетевой карты клиента (листинг 4.3).

Листинг 4.3. Секция описания настроек сетевой карты DHCP-клиента

```
interface "eth0" {
   send host-name "andare.fugue.com";
   send dhcp-client-identifier 1:0:a0:24:ab:fb:9c;
   send dhcp-lease-time 3600;
   request subnet-mask, broadcast-address, routers,
   domain-name, domain-name-servers;
   }
```

В первой строке идет определение, к какому интерфейсу относятся параметры. Затем идет группа строк, начинающихся с `send`. В этих параметрах

задаются те переменные, которые клиент отсылает на сервер для своей идентификации. В данном случае это имя хоста, идентификатор хоста (MAC-адрес сетевой карты, вначале которого добавлена 1) и время аренды IP-адреса в секундах. Затем идет строка, начинающаяся с `request`. Она определяет переменные, которые запрашивает клиент от сервера. В нашем примере это маска подсети, широковещательный адрес, адрес маршрутизаторов, имя домена, адреса DNS-серверов.

Существует еще несколько малоиспользуемых параметров, описание которых вы можете посмотреть в литературе.

Но самое интересное, что для функционирования DHCP-клиента в простейшей локальной сети нет необходимости заполнять файл dhclient.conf! Достаточно просто факта его существования.

Файл dhclient.leases

Файл dhclient.leases, подобно файлу dhcpd.leases, представляет собой базу данных, в которой хранятся параметры DHCP-клиента, полученные от DHCP-сервера (листинг 4.4).

Листинг 4.4. Пример записи из файла dhclient.leases

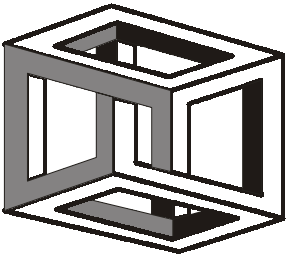
```
lease {
  interface "eth0";
  fixed-address 192.168.1.15;
  option subnet-mask 255.255.240.0;
  option routers 192.168.1.1;
  option domain-name-servers 192.168.1.3;
  option broadcast-address 255.255.255.255;
  option dhcp-server-identifier 192.168.1.2
  option host-name "vasya";
  option domain-name "test.org";
  renew 3 2003/4/2 00:22:38;
  rebind 6 2003/4/5 02:50:06;
  expire 6 2003/4/5 23:50:06;
  }
```

Из этой записи видно, что параметры относятся к сетевому интерфейсу `eth0`. Этой сетевой карте выдан IP-адрес, равный `192.168.1.15`, маска подсети `255.255.240.0`, адрес маршрутизатора, DHCP-сервера и DNS-сервера соответственно `192.168.1.1`, `192.168.1.2` и `192.168.1.3`. Имя хоста — `vasya`, домен — `test.org`. Последние три записи определяют временные параметры аренды IP-адреса.

Литература и ссылки

- Журнал "Сети" № 10/99 г. Иванов П. DHCP: искусство управления IP-адресами.
- **www.asmodeus.com.ua/library/net/dhcp_linux.htm** — Калошин В. Настраиваем DHCP.
- **ezine.daemonnews.org/200207/dhcp.html** — Pham Linh. HOWTO — Setting Up ISC-DHCP 3.x Under FreeBSD.
- **www.dhcp.org** — сервер, полностью посвященный протоколу DHCP.
- **mvd.h1.ru/tr/** — DHCP mini-HOWTO, русский перевод.
- **www.isc.org** — Internet Software Consortium (разработчик DHCP).
- **www.nominum.com/resources/faqs/dhcp-faq.html** — Nominum's DHCP FAQ.
- **www.onlamp.com/pub/a/bsd/2003/04/17/** — Lavigne Dru. Introducing DHCP.
- **www.onlamp.com/pub/a/bsd/2003/05/01/FreeBSD_Basics.html** — Lavigne Dru. Configuring a DHCP Server.
- **www.onlamp.com/lpt/a/3689** — Lavigne Dru. DHCP on a Multi-Segment Network.
- man dhcpd.conf.
- man dhcpd.leases.
- man dhcp-options.
- man dhclient.leases.

ГЛАВА 5



DNS

DNS — это доменная система имен (Domain Name System). DNS преобразует символическое имя хоста в IP-адрес и наоборот — из IP-адреса в символическое имя. Для чего это нужно? Человеку легче запомнить осмысленное имя — типа www.petechkin.ru, чем 213.162.145.242, а для компьютера проще передать 4 байта адреса, чем 50—60 байт имени.

На заре эры глобальных сетей, все пары "имя — IP-адрес" хранились в файле /etc/hosts. Со временем, когда компьютеров в сети стало тысячи и десятки тысяч, этот механизм стал крайне неэффективен, и на смену пришли DNS-серверы.

DNS-сервер представляет собой базу данных, в которой хранится соответствие символического имени хоста IP-адресу. В сети существуют десятки тысяч серверов DNS, которые обмениваются информацией с другими серверами DNS.

DNS — это иерархическая система. Вершина записывается как . (точка) и произносится как root (корень). В корне существует некоторое количество доменов верхнего уровня (Top Level Domains, TLDs), наиболее известными из которых являются ORG, COM, EDU, GOV, MIL, NET, RU, UA и т. п.

При поиске машины запрос обрабатывается рекурсивно, начиная с корня. Если нужно найти адрес машины moshkin.bins.ru, то ваш сервер имен сперва проверяет свою базу. Если в базе не оказалось нужной нам записи, ваш сервер должен найти сервер имен, который обслуживает ru. Он запрашивает корневой сервер (.), который выдает список серверов ru. Из полученного списка выясняется, какие серверы имен обслуживают зону ru. Далее запрашивается сервер (выбирается по определенному алгоритму или берется первый в полученном списке), чтобы узнать, какие серверы обслуживают зону bins.ru. Затем берется сервер из полученного списка и узнается IP-адрес хоста moshkin.bins.ru. А чтобы в следующий раз не повторять этот поиск, полученную пару "имя — IP-адрес" ваш сервер DNS сохраняет в своей базе данных.

При обратной задаче — по IP-адресу узнать имя хоста — опять используется DNS-сервер. Для этих целей существует псевдодомен in-addr.arpa и в нем точно так же прописываются адреса, только порядок следования цифр обратный. Например, для адреса 213.162.145.242 запрос получится как к 242.145.162.213.in-addr.arpa, а схема поиска ответа остается такая же.

По своим функциональным обязанностям различают два вида DNS-серверов:

- *кэширующий сервер DNS* используется для локального хранения запрошенных пользователем пар "имя — IP-адрес", что при интенсивном общении со многими Web-серверами позволяет экономить время на DNS-запросах. Кэширующий сервер не отвечает на внешние DNS-запросы;
- *обычный сервер DNS* — это полнофункциональный сервер, позволяющий получать, передавать и синхронизировать DNS-данные с другими DNS-серверами.

Настройка сетевых параметров

Поскольку DNS-сервер очень сильно завязан на всю сетевую инфраструктуру, его работоспособность зависит от правильной конфигурации сети. В современных дистрибутивах, если вы выбрали "устанавливать DNS-сервер", то его конфигурирование производится автоматически. Однако разработчик дистрибутива рассчитывает на абстрактную среднестатистическую систему, которой, как показывает практика, не существует. Поэтому следует убедиться, что с сетевыми настройками у вас все в порядке, вследствие чего рассмотрим основные файлы, отвечающие за конфигурацию сети.

Файл host.conf

Следующая запись в файле host.conf означает, что при поиске хостов система сначала посмотрит в /etc/hosts, а потом только обратится к серверу DNS:

```
order hosts,bind
```

Запись должна быть именно такая, поскольку возможен случай, что по какой-либо причине нет доступа к серверу DNS (простейший случай — он еще не запущен), а уже есть необходимость воспользоваться сетью.

Файл /etc/hosts

В этом файле должны находиться пары "IP-адрес — имя":

```
127.0.0.1    localhost localhost.localdomain
192.168.0.1  user
192.168.0.2  user2
```

Причем обязательно должна присутствовать следующая строка:

```
127.0.0.1    localhost localhost.localdomain
```

Этот файл позволяет делать преобразование "Имя хоста — IP-адрес" без обращений к DNS-серверу. Обычно используется для небольшой локальной сети, когда нет необходимости в установке и настройке DNS-сервера.

Файл /etc/resolf.conf

В этом файле должны находиться строки, подобные приведенным далее:

```
search bins.ru
nameserver 213.166.195.22
```

В строке, которая начинается со слова `search`, указывается, какое доменное имя будет принято по умолчанию. Так, если вы напишете user, то система сразу попытается обратиться к компьютеру user.bins.ru. После `search` можно указывать несколько имен. В следующей строчке указываются адреса DNS-серверов, к которым будет обращаться ваша машина.

Настройка кэширующего сервера

Кэширующий сервер предназначен для нахождения пары "Имя хоста — IP-адрес" в своей базе или на других DNS-серверах и сохранения пары у себя в базе. Но при этом он сконфигурирован таким образом, что только принимает данные извне сети, но не отдает информацию наружу. Такого типа DNS-серверы используются для уменьшения времени ожидания ответа от DNS-сервера и рекомендуются при медленном соединении или в том случае, когда внешний DNS-сервер перегружен.

Рассмотрим конфигурационные файлы.

Файл /etc/named.conf

Это основной конфигурационный файл DNS-сервера. Для кэширующего сервера он должен содержать следующие строки:

```
options {
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};
```

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file "127.0.0";
};
```

Строка `directory` указывает `bind`, где искать файлы. Все файлы, используемые впоследствии, будут иметь путь относительно этого каталога.

Строка `zone "0.0.127.in-addr.arpa"` показывает, что `bind` также отвечает за обратную зону для подсети 127.*.*.*, является в ней мастером, и файл описания зоны — 127.0.0.

Секция `zone "."` самая важная. Она описывает, в каком файле лежат адреса корневых DNS-серверов, которые отвечают за зоны первого уровня.

Файл, названный /var/named/root.hints, должен находиться в указанном каталоге и содержать приблизительно следующую 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.

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
```

Этот файл описывает имена корневых серверов имен по всему миру. Их список периодически изменяется. Поэтому данный файл необходимо время от времени корректировать.

Для получения файла root.hints существует по меньшей мере два пути: либо забрать его по FTP с сервера internic, либо выполнить команду:

```
dig @rs.internic.net . ns >root.hints
```

Файл /etc/127.0.0

127.0.0 — это файл, который отвечает за преобразование IP-адресов в символические имена.

Файл 127.0.0 должен выглядеть следующим образом:

```
@             IN     SOA   ns.bins.ru. hostmaster.bins.ru. (
                                 1       ; Serial
                                 8H      ; Refresh
                                 2H      ; Retry
                                 1W      ; Expire
                                 1D)     ; Minimum TTL
IN                       NS      ns.bins.ru.
1                        PTR     localhost.
```

Эта запись обозначает следующее:

- `@` указывает, что описываем сами себя;
- описываемая зона поддерживается сервером с именем `ns.bins.ru`;
- отвечает за нее администратор, доступный по адресу `hostmaster@bins.ru` (первая точка заменяет `@`);
- у зоны серийный номер равен `1` (обычно для него используют дату последней правки зоны — на него опираются другие серверы, которые получают информацию с вашего сервера);
- другие серверы будут обновлять информацию о вашем сервере с периодичностью в `8` часов;
- при неудачном обновлении следующая попытка будет произведена через `2` часа;
- зона будет считаться содержащей недостоверную информацию на кэширующих серверах через `1` неделю;
- но не менее, чем через `1` день;
- строка `IN NS ns.bins.ru.` показывает, что авторитетным сервером для этой зоны является `ns.bins.ru.`, и именно ему надо рассылать обновления зоны `ns.bins.ru`;

- строка `1 PTR localhost.` показывает, что хост с адресом `1` в зоне `127.0.0.` имеет имя `localhost`.

Запуск named

После правки конфигурационных файлов можно запускать сервер. Наберите `ndc start` без опций и нажмите клавишу <Enter>.

Затем запускаем программу nslookup:

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1

>_
```

Если вы увидели нечто похожее на мониторе — система работает. Каждый раз, когда вы изменяете файл named.conf, необходимо перезапустить named, используя команду `ndc restart`.

Теперь проверим, как функционирует ваш кэширующий сервер — введем `user.bins.ru`:

```
> user7.bins.ru
Server:  localhost
Address:  127.0.0.1

Name:    user7.bins.ru
Address:  213.166.195.55
```

При этом программа nslookup попросила ваш сервер DNS посмотреть информацию о данном хосте. Если вы повторно попросите узнать адрес компьютера `user7.bins.ru`, то получите такой ответ:

```
> user7.bins.ru
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
Name:    user7.bins.ru
Address:  213.166.195.55
```

В этот раз вы получили сообщение "Non-authoritative answer". Это значит, что DNS во второй раз не делал запрос к внешним серверам имен, а произвел поиск в своем кэше и нашел нужную запись. Поскольку вы увидели это сообщение, ваш кэширующий DNS-сервер нормально функционирует. Получив положительный результат, можно завершить работу программы nslookup, дав команду `exit`.

Настройка полнофункционального DNS-сервера

Настройка полнофункционального DNS-сервера несколько сложнее, чем кэширующего, но, в основном, файлы и записи те же самые. Для чистоты эксперимента рекомендуется произвести настройку для несуществующего домена. У нас он будет называться `ivan.petrov`.

Файл /etc/named.conf

Для нашего сервера DNS данный файл должен содержать следующие строки:

```
options {
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
        type master;
        file "127.0.0";
};
zone "ivan.petrov" {
        notify no;
        type master;
        file "ivan.petrov";
};
zone "0.168.192.in-addr.arpa" {
        notify no;
        type master;
        file "192.168.0";
};
```

Как можно видеть, по сравнению с кэширующим сервером добавилась только секция `zone "ivan.petrov"` и секция `zone "0.168.192.in-addr.arpa"`.

Секция `zone "ivan.petrov"` описывает, что наш DNS-сервер предназначен для зоны `ivan.petrov`, и является в ней мастером (т. е. другие серверы лишь синхронизируют по нему свои записи по зоне `ivan.petrov`), при изменении записей в зоне не извещает другие серверы и использует для описания зоны файл ivan.petrov.

Секция `zone "0.168.192.in-addr.arpa"` описывает, что наш DNS-сервер поддерживает реверсную зону `0.168.192.in-addr.arpa`, является в ней мастером, при изменении записей в зоне не извещает другие серверы и использует для описания зоны файл 192.168.0.

Файл /etc/named/ivan.petrov

В файле зоны `ivan.petrov` поместим следующие данные:

```
@   IN   SOA   ns.ivan.petrov. hostmaster.ivan.petrov. (
               199802151      ; serial, todays date + todays serial #
               8H             ; refresh, seconds
               2H             ; retry, seconds
               1W             ; expire, seconds
               1D)           ; minimum, seconds
;
    NS      ns                     ; Интернет-адрес сервера имен
    MX      10 mail.ivan.petrov.  ; Основной почтовый сервер
    MX      20 mail2.ivan.petrov. ; Дополнительный почтовый сервер
;
localhost       A       127.0.0.1
ns              A       192.168.0.1
mail            A       192.168.0.40
```

Этот файл зоны содержит 4 записи ресурсов (Resource Records, RR):

- `SOA RR` — запись SOA (Start Of Authority, начало полномочий) находится в преамбуле каждого из файлов зон, и она должна быть первой записью в файле. Описывает зону, откуда ее берут (машина, названная `ns.ivan.petrov`), кто отвечает за содержимое зоны (`hostmaster@ivan.petrov`), какая версия файла зоны текущая (`serial: 1`), и другие вещи, которые надо сделать для кэширующих и вторичных серверов DNS;
- `NS RR` — это RR для сервера имен (Name Server, NS);
- `MX RR` — запись MX (Mail eXchanger, почтовый сервер) сообщает почтовой системе, куда посылать почту, адресованную любому адресату в домене `ivan.petrov`, в нашем случае — серверам `mail.ivan.petrov` или `mail2.ivan.petrov`. Число перед каждым именем системы — это приоритет записи MX RR. Запись ресурса с наименьшим номером (10) — это хост, куда почта должна посылаться в первую очередь. Если происходит ошибка, то почта может быть послана на машину с большим номером. И так далее. Таким образом, можно указать несколько почтовых серверов, что поможет вам в случае форс-мажорных обстоятельств не потерять ваши почтовые сообщения;

- `A RR` — A (Address, адрес) — IP-адрес:

```
localhost       A       127.0.0.1
ns              A       192.168.0.1
mail            A       192.168.0.40
```

  Эти строки описывают соответствие имен `mail` и `ns` в зоне `ivan.petrov` их IP-адресам.

Файл /etc/192.168.0

Для нормального функционирования DNS-сервера требуется обратная (реверсная) зона, которая дает возможность DNS преобразовывать IP-адреса в имена хостов. Эти имена используются серверами различного рода (FTP, IRC, WWW и т. п.). Поэтому обратная зона требуется для полного доступа к различным сервисам в Интернете.

Далее представлен файл /etc/192.168.0:

```
@       IN      SOA     ns.ivan.petrov. hostmaster. ivan.petrov. (
                        199802151; Serial, todays date + todays serial
                        8H      ; Refresh
                        2H      ; Retry
                        1W      ; Expire
                        1D)     ; Minimum TTL
                NS      ns.linux.bogus.

2               PTR     gw.ivan.petrov.
1               PTR     ns.ivan.petrov.
3               PTR     petya.ivan.petrov.
40              PTR     mail.ivan.petrov.
5               PTR     ftp.ivan.petrov.
```

Этот файл в принципе мало чем отличается от файла описания прямой зоны. Появились только следующие строки:

```
2               PTR     gw.ivan.petrov.
1               PTR     ns.ivan.petrov.
3               PTR     petya.ivan.petrov.
40              PTR     mail.ivan.petrov.
5               PTR     ftp.ivan.petrov.
```

Эти строки описывают то, что машина с адресом `2` в зоне `192.68.0.` имеет имя `gw.ivan.petrov`, а компьютер с адресом `40` — `mail.ivan.petrov`.

Вот собственно и все. Перезапускаем DNS-сервер и проверяем правильность функционирования нашей системы.

Некоторые тонкости

Как вы видите, глубоко в тонкости функционирования DNS-сервера мы не погружались. Во-первых, этого вполне достаточно для настройки небольшого DNS-сервера, а во-вторых, решать проблемы следует по мере их возникновения, и для этих целей незаменима документация, поставляющаяся вместе с DNS-сервером. Тем не менее несколько полезных вещей необходимо знать.

Записи ресурсов (RR) службы DNS

Давайте рассмотрим несколько расширенный файл описания зоны:

```
gw              A       192.168.0.2
                HINFO   "i486" "RH 6.2"
                TXT     "The router"

ns              A       192.168.0.1
                MX      10 mail
                HINFO   "Pentium3" "RH 9"
www             CNAME   ns
User            A       192.168.0.3
                MX      10 mail
                HINFO   "p3"  "Windows2000"
                TXT     "Developer computer home tel 223344"
```

Помимо знакомых вам строчек появились строки, содержащие `HINFO`, `CNAME` и `TXT`.

- `HINFO` — информация о компьютере (Host INFOrmation) состоит из двух частей: первая часть — это информация об оборудовании машины, а вторая описывает программное обеспечение и операционную систему данной машины. Помимо этой информации не рекомендуется вносить ничего другого. Пример:

  ```
  HINFO   "Pentium3" "RH 9"
  ```

  Из данной строки видно, что наш DNS-сервер собран на базе процессора Pentium III и на нем установлена операционная система Linux Red Hat 9;

- `CNAME` — каноническое имя (Canonical NAME) — это способ присвоить каждой машине несколько имен. При использовании `CNAME` необходимо следовать правилу, что записи `MX`, `CNAME` или `SOA` *никогда* не должны ссылаться на имя, указанное как запись `CNAME`;

- TXT — произвольная текстовая информация. Обычно используется в качестве расширенного комментария для описания хоста. Пример:

  ```
  TXT     "Developer computer home tel 223344"
  ```

  Из содержимого строчки понятно, что это компьютер разработчика, а его домашний телефон — 223344.

Так же существует еще один тип записи — RP (Responsible Party, группа ответственных). В принципе эта же информация может храниться и в записях TXT, однако применение записи RP ускоряет поиск данных об ответственных лицах. Список основных записей ресурсов службы DNS приведен в табл. 5.1.

Таблица 5.1. *Основные записи ресурсов (RR) службы DNS*

Обозначение записи	Содержание записи	Номер RFC или автор проекта
A	IP-адрес хоста	RFC1035
AAAA	Адрес IPv6	Проект, автор Thomson
CNAME	Каноническое имя домена	RFC1035
GPOS	Географическое положение	RFC1712
HINFO	Информация о хосте (процессор и ОС)	RFC1035
ISDN	Адрес ISDN	RFC1183
KEY	Ключ шифрования	(проект, автор Eastlake)
LOC	Расположение	(проект, автор Vixie)
MX	Имя хоста или домена для переадресации почты	RFC1035
NSAP	SAP-адрес (адрес A в формате NSAP)	RFC1706
NSAP-PTR	Аналог записи PTR для адреса NSAP	RFC1706
NULL	Пустая запись ресурса	RFC1035
NXT	Следующий домен	Проект, автор Eastlake
PTR	Указатель на имя домена	RFC1035
RP	Ответственные лица	RFC1183
SIG	Цифровая подпись	(проект, автор Eastlake)
SRV	Выбор сервера	(проект, автор Vixie)
TXT	Произвольный текст	RFC1035
WKS	Описание подключенных сервисов	RFC1035
X25	Адрес X.25	RFC1183

Реверсная зона

Не забывайте об обратной (реверсной) зоне! Очень неприятно, когда по этой причине вы не сможете воспользоваться FTP-сервером или получите сообщения о нарушениях системы защиты. Не поленитесь — потратьте час на описание реверсной зоны.

Два сервера DNS

Существует множество причин, по которым нежелательно раскрывать всю информацию о вашей сети через службу DNS. Поэтому рекомендуется создать два сервера DNS: один для внутренних пользователей, другой для внешних. Для этого необходимо обеспечить два различных набора IP-адресов: для внутренних клиентов и для внешнего мира.

Иерархические поддомены

Если в вашей организации используется более одной подсети, то вам придется задать несколько доменов `in-addr.arpa`. Создание поддоменов, подчиненных первичному домену, целесообразно также при наличии в вашей организации нескольких отделов или подразделений. Это облегчит мониторинг сети, а также упростит организацию доступа в сеть и установку защитных фильтров. Конечно, если ваша сеть состоит всего из нескольких машин, смысла в создании иерархии доменов просто нет.

Вторичные DNS-серверы

Если у вас большая сеть или если вы занимаетесь хостингом сайтов, вы обязательно должны помимо первичного сервера DNS иметь еще и вторичный сервер DNS. Это позволит уменьшить время отклика на запрос, а также повысить отказоустойчивость сети.

Используйте серверы кэширования

Если вы занимаетесь обслуживанием сети, рекомендуется установить кэширующие DNS-серверы, если не на каждый компьютер, то в каждой подсети. Быстродействие, которое обеспечивает такой подход, становится заметным уже в сетях средней сложности.

Инструменты

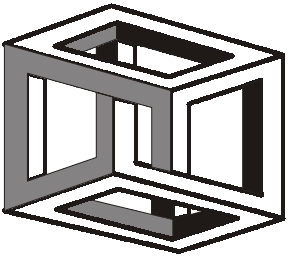
Для тех, кто не хочет подробно изучать настройку DNS с помощью конфигурационных файлов, существуют доступные инструменты, позволяющие вносить изменения, особо не задумываясь. Произведите поиск и вы навер-

няка найдете десяток-другой программ для удаленного администрирования DNS-сервера, в том числе и имеющих графическую "дружественную" оболочку. В частности, исходный код на языке HTML для создания инструментария по управлению службой DNS можно найти в Интернете по адресу **webdns.lcs.mit.edu/cgi-bin/webdns**/. Существует так же универсальная программа для администрирования множеством сервисов через Интернет — webmin.

Литература и ссылки

- DNS-HOWTO.
- **linux.webclub.ru/bind/pers_dns.html** — Водолазкий В. Мой личный сервер DNS.
- **www.biblioteka.agava.ru/nastroyka_dns.htm** — Калошин В. Настройка DNS.
- **www.4com.ru/support/DNSAdvanSetup.html** — Холл Э. Тонкая настройка DNS.
- **www.webmin.com/webmin/** — сайт программы webmin.

ГЛАВА 6



Почта

Функционирование электронной почты очень похоже на свой бумажный прототип. Человек пишет письмо, подписывается, указывает на конверте адрес отправителя и получателя, запечатывает конверт и бросает его в почтовый ящик. Специальная служба, условно назовем ее курьерской, периодически объезжает почтовые ящики, собирает письма и отвозит их на почтамт. Там их сортируют и отправляют на почтамты в города назначения. На этих почтамтах письма сортируются по районам и адресатам, и почтальоны доставляют их по почтовым ящикам адресатов.

По такому же принципу работает и электронная почта. Есть программа — почтовый клиент, в которой происходит подготовка почты к отправке и получение почты. Есть программа — транспортный агент, которая отвечает за доставку электронной почты от компьютера к компьютеру, и есть программы, выполняющие роль почтамтов — они получают почту, сортируют ее по адресатам и раскладывают по почтовым ящикам.

Для транспортировки почты используется MTA (Mail Transport Agent, почтовый транспортный агент). Старейшим и наиболее распространенным транспортным агентом является программа sendmail. Помимо sendmail получили популярность программы Qmail, postfix, exim.

Основой почтовой службы является система адресов. В Интернете принята система адресов, базирующаяся на доменном адресе компьютера, подключенного к сети. Например, для пользователя test хоста с адресом tech.mega.ru почтовый адрес будет выглядеть так: **test@tech.mega.ru**.

Почтовый адрес состоит из двух частей: идентификатора пользователя, который записывается перед знаком "коммерческого at" — "@", и доменного адреса компьютера, который записывается после знака "@". Существует еще один вариант задания почтового адреса — адрес UUCP, который записывается в виде **mega.ru!tech!test**. Правда протокол UUCP сейчас почти не используется.

Для работы электронной почты разработан специальный протокол — SMTP (Simple Mail Transfer Protocol, простой почтовый протокол). Он является протоколом прикладного уровня и использует транспортный протокол TCP.

Протокол SMTP

Протокол SMTP был разработан для обмена почтовыми сообщениями в сети Интернет. Он не зависит от транспортной среды и может использоваться для доставки почты в сетях с протоколами, отличными от TCP/IP.

Взаимодействие в рамках SMTP строится по принципу двусторонней связи, которая устанавливается между отправителем и получателем почтового сообщения. При этом отправитель инициирует соединение и посылает запросы на обслуживание, а получатель на эти запросы отвечает.

Как и множество других протоколов, команды и ответы протокола SMTP передаются в ASCII-кодах и представляют собой небольшой набор английских слов. Приведем пример отправки почтового сообщения по протоколу SMTP:

```
Отправитель: MAIL FROM: <test@tech.mega.ru>
Получатель: 250 Ok
Отправитель: RCPT TO: <tvan@mail.ru>
Получатель: 250 Ok
Отправитель: DATA
Получатель: 354 Start mail input; end with <CRLF>.<CRLF>
Текст почтового сообщения
Отправитель:
Получатель: 250
```

Протокол помимо отправки почты поддерживает переадресацию, прямую посылку сообщения на терминал, обработку ошибок и некоторые другие возможности.

Протокол POP3

Протокол обмена почтовой информацией POP3 (Post Office Protocol) предназначен для получения почты из почтовых ящиков пользователей на их рабочие места при помощи программ-клиентов. Таким образом, по протоколу SMTP пользователи отправляют корреспонденцию, а по протоколу POP3 получают корреспонденцию из своих почтовых ящиков на почтовом сервере в локальные файлы. Этот протокол так же основан на установлении двусторонней связи, команды и ответы протокола передаются в ASCII-кодах и представляют собой небольшой набор английских слов.

Протокол IMAP

Еще одним почтовым протоколом является протокол IMAP (Interactive Mail Access Protocol, почтовый протокол интерактивного доступа), который по своим возможностям похож на POP3, но разрабатывался как более надежная альтернатива последнему и обладает более широкими возможностями по управлению процессом обмена сообщениями с сервером.

Главным отличием от POP3 является возможность поиска нужного сообщения и разбор заголовков сообщения непосредственно на почтовом сервере.

Формат почтового сообщения

Формат почтового сообщения определен в документе RFC822. Почтовое сообщение состоит из трех частей: конверта, заголовка и тела сообщения. Конверт используется программами доставки почтового сообщения, а заголовок и тело сообщения предназначены для его получателя. Заголовок находится перед телом сообщения, отделен от него пустой строкой и состоит из определенных стандартом полей. Поля включают имя и содержание. Имя поля отделяется от содержания символом ":". Для доставки сообщения можно воспользоваться только частью полей заголовка — `Date`, `From`, `Cc` или `To`, например:

```
Date:  26 Aug 76 1429 EDT
From:  1@mail.ru
To:    Sm2@chat.ru
```

Поле `Date` определяет дату отправки сообщения, поле `From` — отправителя, а поля `Cc` и `To` — получателей. Однако, если следовать установленным правилам, необходимо определять все поля заголовка, описанные в стандарте:

```
Date:          27 Aug 76 0932
From:          Motya <1@mail.ru>
Subject:       Re: Ответ на письмо
Sender:        K@Other-host
Reply-To:      Sam.Irving@R.org.ru
To:            Geo <J@chat.ru>
Cc:
Comment:       Sam is away on bisiness
In-Reply-To:   <some.string@DBM.Group>, George`s message
Message-ID:    <4331.629.XYzi-What@Other-Host
```

Поле `Subject` определяет тему сообщения, `Reply-To` — пользователя, которому отвечают, `Comment` — комментарий, `In-Reply-To` — показывает, что сообщение относится к типу "В ответ на Ваше сообщение, отвечающее на со-

общение, отвечающее ...", `Message-ID` — уникальный идентификатор письма, используемый почтовыми программами.

Формат сообщения постоянно дополняется и совершенствуется. В частности, в RFC1327 введены дополнительные поля для совместимости с почтой X.400.

Спецификация MIME

Стандарт MIME (Multipurpose Internet Mail Extension), описанный в стандарте RFC1341, предназначен для описания тела почтового сообщения Интернета. Необходимость в этом стандарте возникла в силу того, что по стандарту RFC822 в тело почтового сообщения не могут быть включены некоторые специальные символы и восьмибитовые символы.

Стандарт RFC822 подробно описывает в заголовке почтового сообщения текстовое тело письма и механизм его рассылки, а MIME сориентирован на описание в заголовке письма структуры тела почтового сообщения и возможности составления письма из информационных единиц различных типов.

В стандарте зарезервировано несколько способов представления разнородной информации. Для этого используются специальные поля заголовка почтового сообщения:

- поле версии MIME (используется для идентификации сообщения, подготовленного в новом стандарте);
- поле описания типа информации в теле сообщения (позволяет обеспечить правильную интерпретацию данных);
- поле типа кодировки информации в теле сообщения (указывает на тип процедуры декодирования);
- два дополнительных поля (зарезервированы для более детального описания тела сообщения).

Стандарт MIME разработан как расширяемая спецификация, в которой подразумевается, что число типов данных будет расти по мере развития форм представления данных.

Рассмотрим некоторые из полей MIME.

Поле *MIME-Version*

Поле версии указывается в заголовке почтового сообщения и позволяет определить, что сообщение подготовлено в стандарте MIME. Формат поля:

```
MIME-Version: 1.0
```

Поле версии указывается в общем заголовке почтового сообщения и относится ко всему сообщению целиком.

Поле *Content-Type*

Поле типа используется для описания типа данных, которые содержатся в теле почтового сообщения. Это поле сообщает программе чтения почты, какие преобразования необходимы для того, чтобы сообщение правильно проинтерпретировать. Эта же информация используется и программой рассылки при кодировании/декодировании почты. Стандарт MIME определяет семь типов данных, которые можно передавать в теле письма. Для важнейших из них приведем краткие описания.

- *Текст* (`text`). Этот тип указывает на то, что в теле сообщения содержится текст. Основным подтипом типа `text` является `plain` — плоский текст (подразумевается неразмеченный текст). Для определения размеченного текста используют подтип `richtext`, а для определения гипертекста — подтип `html`.
- *Смешанный тип* (`multipart`). Этот тип определяет смешанный документ, который может состоять из фрагментов данных разного типа. Данный тип имеет ряд подтипов.
- *Сообщение* (`message`). Этот тип предназначен для работы с обычными почтовыми сообщениями, которые напрямую не могут быть переданы по почте. Существует несколько подтипов:
  - `partial` — подтип предназначен для передачи одного большого сообщения по частям для последующей автоматической сборки у получателя;
  - `External-Body` — подтип позволяет ссылаться на внешние информационные источники;
  - `rfc822` — стандартный подтип типа `message`. Определяет сообщения стандарта RFC822.
- *Графический образ* (`image`).
- *Аудиоинформация* (`audio`).
- *Видеоинформация* (`video`).
- *Приложение* (`application`).

Поле *Content-Transfer-Encoding*

Тип кодирования сообщения. Поскольку передача сообщений происходит в неоднородной среде, неизбежны перекодирования почтового сообщения. Для того чтобы при получении данные были правильно распакованы, и используется данное поле.

Программное обеспечение

Как и многое другое, взаимодействие между участниками обмена почтовым сообщением основано на технологии "клиент-сервер". Можно выделить три независимых этапа:

- взаимодействие по протоколу SMTP между почтовым клиентом и почтовым транспортным агентом;
- взаимодействие между транспортными агентами в процессе доставки почты;
- получение сообщения из почтового ящика пользователя почтовым клиентом по протоколу POP3 или IMAP.

Спецификация S/MIME

S/MIME (Secure/Multipurpose Internet Mail Extensions) — безопасная реализация MIME, описана в RFC2630, RFC2632, RFC2633, RFC2634). S/MIME представляет собой реализацию криптографической системы с асимметричным ключом. Система может быть использована как для внедрения цифровой подписи в почтовые сообщения, так и для шифрования оных. Поддерживается также комбинация двух перечисленных ранее методов. Система с асимметричным ключом отличается от работающей с симметричным ключом тем, что отправителю не придется сообщать пароль, который использовался для пересылки секретных сведений. Все, что необходимо иметь — публичную часть S/MIME-сертификата вашего адресата, т. е. ту его часть, которая секретом не является. Корреспондент должен иметь и секретную часть своего S/MIME-сертификата, иначе он не сможет дешифровать полученный текст.

В случае применения электронной подписи подписанное сообщение передается в незашифрованном виде, однако получатель письма имеет возможность убедиться в подлинности отправителя и в целостности (не искаженности) письма. При этом получателю письма для проверки не требуется никакой дополнительной информации, т. к. публичная часть S/MIME-сертификата отправителя подписанного письма передается вместе с этим письмом.

Для использования S/MIME (а также PGP или GPG) клиентское программное обеспечение должно уметь работать с данными протоколами. Большинство современных программ поддерживают такую возможность.

PGP, GPG

Подобно S/MIME, PGP (Pretty Good Privacy) и ее аналог из мира GNU GPG (GnuPG, GNU Privacy Guard) используются для надежного шифрования почтовых сообщений и организации электронной подписи. Однако по-

мимо электронной почты PGP используется (по крайней мере достаточно широко в мире Windows) для шифрования файлов и организации шифрованных псевдодисков.

Программа sendmail

Основным средством рассылки почты является программа sendmail, хотя она одна из старейших и сложных в конфигурации. Sendmail позволяет организовать почтовую службу локальной сети и обмениваться почтой с другими серверами почтовых служб через специальные шлюзы. Sendmail может быть сконфигурирована для работы с различными почтовыми протоколами. Обычно это протоколы UUCP (UNIX-UNIX-CoPy) и SMTP (Simple Mail Transfer Protocol).

Sendmail может интерпретировать два типа почтовых адресов:

- почтовые адреса SMTP;
- почтовые адреса UUCP.

Sendmail можно настроить для поддержки:

- списка адресов-синонимов;
- списка адресов рассылки пользователя;
- автоматической рассылки почты через шлюзы;
- очередей сообщений для повторной рассылки почты в случае отказов при рассылке;
- работы в качестве SMTP-сервера;
- доступа к адресам машин через сервер доменных имен BIND;
- доступа к внешним серверам имен

  и многого другого.

Принцип работы

Sendmail идеологически копирует обычную почтовую службу — почта отправляется с заданной периодичностью, перед этим сообщения собираются в очереди и только затем отсылаются.

Как уже упоминалось ранее, каждое сообщение состоит из трех частей: конверта, заголовка и тела сообщения:

- *конверт* состоит из адреса отправителя, адреса получателя и специфической информации, которая используется программами подготовки, рассылки и получения почты. Конверт остается невидимым для отправителя и получателя почтового сообщения;
- *заголовок* состоит из стандартных текстовых строк, которые содержат адреса, информацию о рассылке и данные. Данные из заголовка могут использоваться для оформления конверта сообщения;

- *тело* сообщения следует после первой пустой строки вслед за заголовком сообщения. Все, что следует после этой строки, называется телом сообщения и передается по почте без изменений.

После постановки почтовых сообщений в очередь начинается ее рассылка. При этом выполняются следующие действия:

- адреса отправителя и получателя преобразуются в формат сети — получателя почты;
- если необходимо, то в заголовок сообщения добавляются отсутствующие данные;
- почта передается одной из программ рассылки почты.

Настройка программы

Настройка программы sendmail происходит при помощи конфигурационного файла /etc/sendmail.cf. Этот файл состоит из нескольких частей, которые описывают:

- компьютер (local information) — имя компьютера и т. п;
- макроопределения sendmail, отвечающие за работу в локальной сети;
- группы имен, которые используются программой для рассылки почты;
- номер версии файла конфигурации;
- опции команды `sendmail`, которые определяют режимы работы программы;
- доверенных пользователей;
- формат заголовка почтового сообщения — в данной секции определяются поля, отображаемые в заголовке, и их формат;
- правила преобразования адресов;
- программы рассылки;
- общий набор правил преобразования адресов;
- машинно-зависимую часть общего набора правил преобразования адресов.

Обычно после инсталляции sendmail изменения, которые вносятся в файл конфигурации, касаются только имени хоста, домена и шлюзов. В современных дистрибутивах (таких как Red Hat) иногда не приходится делать даже этого.

Подробно о конфигурировании sendmail здесь рассказано не будет — разобраться в конфигурационном файле, который имеет около 100 Кбайт текста, весьма не просто. Для детального ознакомления с конфигурацией sendmail рекомендуется почитать книгу "UNIX — руководство системного администратора", а также документацию, идущую в комплекте с sendmail.

Для примера приведем небольшую секцию локальной конфигурации sendmail:

```
##################
#   local info   #
##################
Cwlocalhost
CP.
# UUCP relay host
DYucbvax.Berkeley.EDU
CPUUCP
#  BITNET relay host
#DBmailhost.Berkeley.EDU
DBrelay.kiae.su
CPBITNET
# "Smart" relay host (may be null)
DSrelay.kiae.su
# who I send unqualified names to (null means deliver locally)
DR
# who gets all local email traffic ($R has precedence for unqualified names)
DH
# who I masquerade as (null for no masquerading)
DM
# class L: names that should be delivered locally, even if we have a relay
# class E: names that should be exposed as from this host,
  even if we masquerade
#CLroot
CEroot
# operators that cannot be in local usernames (i.e., network indicators)
CO @ % !
# a class with just dot (for identifying canonical names)
C..
# dequoting map Kdequote dequote
```

Тестирование отправки почты sendmail

Для проверки правильности функционирования программы sendmail можно запустить ее с ключом `-v` (режим verbose). При этом режиме процесс обмена между транспортными почтовыми агентами выводится на консоль или записывается в файл. Таким образом, можно исключить большую часть ошибок в настройке sendmail.

Тестирование обслуживания по протоколу SMTP

Для проверки сервиса SMTP используют программу telnet, подключаемую к 25 порту:

```
telnet ivan.petrov 25
```

Если на компьютере установлен SMTP-сервер, в ответ получим строку приглашения протокола SMTP, после чего можно вводить команды SMTP:

```
MAIL FROM: user
250 user... Sender ok
RCPT TO: user
250 user... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
This is a test message!!!
...
250 JAA24856 Message accepted for delivery
quit
221 ivan.petrov closing connection
Connection closed by foreign host.
You have new mail.
#
```

В приведенном примере мы отправили сами себе сообщение. Команда `MAIL FROM:` указывает адрес отправителя почтового сообщения. Затем вводится команда `RCPT TO:` для указания адреса получателя почтового сообщения. Команда `DATA` разрешает ввод почтового сообщения. Конец режима редактирования обозначается символом "." в первой позиции строки. Более подробную информацию следует искать в документации по sendmail, а также в табл. 6.1, где приведены команды протокола SMTP, и в табл. 6.2, содержащей коды возврата протокола SMTP.

Команды и коды возврата протокола SMTP

Для тестирования работы SMTP-сервера необходимо знать команды протокола SMTP (табл. 6.1) и его коды возврата (табл. 6.2), а также воспользоваться программой telnet.

Таблица 6.1. *Команды протокола SMTP*

Команда	Описание
`HELO <SP> <domain> <CRLF>`	Открыть сессию взаимодействия по протоколу SMTP. `<domain>` — доменное имя машины
`MAIL <SP> FROM:<reverse-path> <CRLF>`	Сообщить адрес отправителя `<reverse-path>`. Обязательная команда, которую надо выдать перед отправкой сообщения
`RCPT <SP> TO:<forward-path> <CRLF>`	Сообщить адрес получателя `<forward-path>`. Обязательная команда, которую выдают после `MAIL FROM`, но перед `DATA`

Таблица 6.1 *(окончание)*

Команда	Описание
DATA <CRLF>	Начать передачу тела почтового сообщения. Тело сообщения должно завершаться точкой "." в первой позиции строки
RSET <CRLF>	Конец операции
SEND <SP> FROM:<reverse-path> <CRLF>	Послать сообщение на терминал пользователя, который определяется командой RCPT
SOML <SP> FROM:<reverse-path> <CRLF>	SEND OR MAIL. Послать в почтовый ящик или на терминал пользователя
SAML <SP> FROM:<reverse-path> <CRLF>	SEND AND MAIL. Послать в почтовый ящик и на терминал пользователя
VRFY <SP> <string> <CRLF>	Получить информацию о пользователе, имя которого указывается в качестве аргумента команды <string>
EXPN <SP> <string> <CRLF>	Получить информацию о пользователях, зарегистрированных в качестве получателей корреспонденции
HELP [<SP> <string>] <CRLF>	Краткая справка по командам протокола
NOOP <CRLF>	Нет операции
QUIT <CRLF>	Завершить сессию
TURN <CRLF>	Поменять местами сервер и клиент

Таблица 6.2. *Коды возврата протокола SMTP*

Код возврата	Текстовое пояснение сервера	Описание
211	System status, or system help reply	Статус системы или помощь
214	Help message. [Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user]	Краткая справка
220	<domain> Service ready	SMTP-сервис готов к работе

Таблица 6.2 *(продолжение)*

Код возврата	Текстовое пояснение сервера	Описание
221	<domain> Service closing transmission channel	Сервис закрыл канал передачи данных
250	Requested mail action okay, completed	Соединение установлено
251	User not local; will forward to <forward-path>	Пользователь не местный. Выполнить перенаправление запроса
354	Start mail input; end with <CRLF>.<CRLF>	Начать ввод почтового сообщения
421	<domain> Service not available, closing transmission channel [This may be a reply to any command if the service knows it must shut down]	Сервис отсутствует. Канал передачи данных закрыт
450	Requested mail action not taken: mailbox unavailable [E.g., mailbox busy]	Нет возможности записать данные в почтовый ящик
451	Requested action aborted: local error in processing	Ошибка при обработке запроса
452	Requested action not taken: insufficient system storage	Запрос не выполнен — недостаточно памяти
500	Syntax error, command unrecognized [This may include errors such as command line too long]	Синтаксическая ошибка — нет такой команды
501	Syntax error in parameters or arguments	Синтаксическая ошибка в аргументах команды
502	Command not implemented	Данная команда не может быть выполнена
503	Bad sequence of commands	Неправильная последовательность команд
504	Command parameter not implemented	Параметр команды не может быть использован в данном контексте
550	Requested action not taken: mailbox unavailable [E.g., mailbox not found, no access]	Не найден соответствующий почтовый ящик

Таблица 6.2 *(окончание)*

Код возврата	Текстовое пояснение сервера	Описание
551	User not local; please try <forward-path>	Пользователь не найден; можно попробовать отправить почту по другому адресу
552	Requested mail action aborted: exceeded storage allocation	Превышены квоты на использование ресурсов памяти
553	Requested action not taken: mailbox name not allowed [E.g., mailbox syntax incorrect]	Имя почтового ящика неправильное
554	Transaction failed	Аварийное завершение

Тестирование обслуживания по протоколу POP3

Аналогично тестированию обслуживания по протоколу SMTP с помощью программы telnet можно проверить функционирование и POP3-протокола. Для этого необходимо подключиться к нашему серверу по порту 110.

```
telnet ivan.petrov 110
  user user
  +OK Password required for user.
  pass 12345623432
  +OK user has 3 messages (33276 octets).
  list
  +OK 3 messages (33276 octets)
  1 11276
  2 11000
  3 11000
  .
  dele 3
  +OK Message 3 has been deleted.
  quit
  +OK
  Connection closed by foreign host.
```

Очень похоже на протокол SMTP. Подключились к порту 110. Производим "опознание" пользователя с помощью команд `user` и `pass`. Затем командой `list` узнаем количество сообщений в почтовом ящике и их размер. Командой `dele` отмечаем сообщение к удалению, которое произойдет по окончании сеанса. Команда `quit` завершает сеанс работы с сервером. Все просто.

Команды протокола POP3

Для тестирования работы POP3-сервера необходимо знать его команды (табл. 6.3) и воспользоваться программой telnet.

Успешное выполнение команды оканчивается выводом сообщения `+OK`, а неуспешное `-ERR`.

Таблица 6.3. *Команды протокола POP3*

Команда	Назначение	Возможные возвращаемые значения (кроме `+OK` или `-ERR`)
`USER` *<имя пользователя>*	Посылка имени пользователя серверу	
`PASS` *<пароль>*	Посылка пароля серверу	
`QUIT`	Окончание сеанса работы	
`STAT`	Получить состояние почтового ящика	`+OK` *<кол-во сообщений> <общий размер всех сообщений>*
`UST [`*<номер сообщения>*`]`	Получить параметры всех сообщений в ящике пользователя. Если задан номер сообщения, то будут получены только его параметры	`+OK` *<параметры сообщений>* Возвращаемые параметры сообщений зависят от того, был ли задан номер сообщения. Если да, то сразу после `+OK` следует сообщение сервера. Затем строка за строкой передаются параметры всех сообщений в формате *<номер сообщения> <размер сообщения>*
`RETR` *<номер сообщения>*	Получить сообщение с сервера	`+OK` *<тест запрошенного сообщения>* — если команда прошла удачно `-ERR` *<комментарий сервера>* — если запрошенное сообщение отсутствует на сервере
`DELE` *<номер сообщения>*	Пометить сообщение на сервере как удаленное. Реально оно будет удалено после команды `quit`	`+OK` *<комментарий сервера>* — если сообщение было помечено на удаление `-ERR` *<комментарий сервера>* — если сообщение не существует или уже отмечено как удаленное

Таблица 6.3 *(окончание)*

Команда	Назначение	Возможные возвращаемые значения (кроме +OK или -ERR)
NOOP	Пустая операция	+OK
RSET	Отменить удаление сообщений, помеченных как удаленные	
TOP *<номер сообщения>* *<кол-во строк>*	Считать заголовок сообщения и первые строки в количестве, заданном параметром *<кол-во строк>*	+OK Далее строка за строкой передается заголовок сообщения. За ним следует пустая строка и, если имеется второй параметр, передаются начальные строки сообщения
UIDL [*<номер сообщения>*]	Получить уникальные идентификаторы всех сообщений в ящике пользователя. Если задан номер сообщения, то будет получен только его идентификатор	+OK *<параметры сообщений>* Возвращаемые параметры сообщений зависят от того, был ли задан номер сообщения. Если да, то сразу после +OK идут номер запрошенного сообщения и его идентификатор. Если команда вызвана без параметра, то после статуса +OK следует сообщение сервера. Затем строка за строкой передаются параметры всех сообщений в формате *<номер сообщения>* *<идентификатор>*
APOP *<имя пользователя>* *<дайджест>*	Осуществляет подключение к почтовому серверу по закодированной алгоритмом MD5 строке, защищая транзакцию от разглашения пароля пользователя	+OK *<комментарий сервера>* — если имя пользователя или дайджест соответствуют имеющемуся почтовому ящику пользователя

Программа Postfix

В последнее время Postfix становится популярным MTA-агентом. Sendmail проигрывает Postfix по ряду параметров и к тому же неоправданно сложен в настройке. Для работы Postfix создает группы postfix и postdrop. В группе

postfix добавляется пользователь postfix. Почтовая система работает с правами пользователя postfix, а группа postdrop владеет очередью сообщений.

Конфигурационные файлы

Файлы настройки располагаются в каталоге /etc/postfix. Основные параметры определяются в файле main.cf.

В первоначальном виде этот файл содержит конфигурацию, позволяющую серверу работать в пределах машины, а также развернутые комментарии с примерами.

Далее рассмотрим кратко конфигурирование Postfix. Большинство опций останутся неизменными, а рассмотрены будут те опции, которые необходимо изменить.

- `myhostname=tech.test.ru`. Имя хоста. Лучше всего устанавливать в соответствии с результатами выполнения `hostname`.
- `mydomain=test.ru`. Имя домена. Если переменная не определена, то Postfix использует подкорректированное значение `myhostname`.
- `inet_interfaces=192.168.0.2, 195.80.10.26`. Адреса интерфейсов, на которых нужно ждать SMTP-соединений. Можно использовать слово `all` для установки прослушивания всех интерфейсов.
- `mydestination=$myhostname, $mydomain`. Список доменов, которые обслуживает программа.
- `mynetworks=192.168.0.0/24, 127.0.0.0/8`. Задает список доверенных сетей. Клиенты с адресами, принадлежащими этим сетям, смогут рассылать через нас почту. Если этот параметр не определен, то доверенной сетью считается IP-подсеть, к которой принадлежит машина с Postfix.
- `alias_database=dbm:/etc/postfix/aliases`. Путь к файлу почтовых псевдонимов.

Вот вкратце и все. Само собой, параметров намного больше, что позволяет очень тонко настроить программу. Однако для небольшой сети достаточно произвести перечисленные ранее операции.

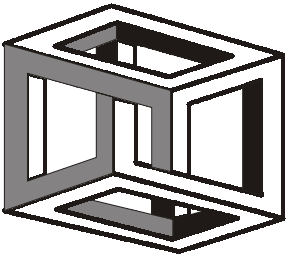
Для проверки корректности конфигурационного файла достаточно выполнить команду `postfix check`.

После редактирования конфигурации при работающем Postfix ее нужно активизировать командой `postfix reload`.

Литература и ссылки

- Руководство администратора ALT Linux.
- Allman E. Sendmail installation and operation guide.

- Linux Mail-Queue mini-HOWTO.
- Sendmail address rewriting mini-HOWTO.
- Sendmail+UUCP HOWTO.
- **host.ru/documentation/v-www/0020.html** — Использование PGP.
- **onix.opennet.ru/mail/mail.html** — Бешков А. Почтовая система среднего и малого офиса.
- **www.arh.ru/~zwon/gph/book1.html** — Описание GNU Privacy Guard.
- **www.citforum.ru/internet/servers/** — Храмцов П. Организация и администрирование почтовых и файловых серверов Internet. Центр Информационных Технологий.
- **www.compulenta.ru/dk/offline/2001/64/13214/print.html** — Зиммерманн Ф. Основы криптографии.
- **www.opennet.ru/docs/RUS/mail/index.html** — Мамаев М. Электронная почта.



ГЛАВА 7

Сетевая информационная система NIS (NIS+) и ее конфигурирование. LDAP

В этой главе описаны две службы NIS (NIS+) и LDAP, предназначенные для оптимизации настроек "сетевых" пользователей (т. е. тех пользователей, которые могут часто менять свое местоположение и использовать разные компьютеры), а также для облегчения жизни сетевых администраторов. Как обычно, используется технология "клиент-сервер", поэтому будет рассмотрена настройка как клиентской части, так и серверной.

NIS

NIS (Network Information Service, служба сетевой информации) — служба, предоставляющая информацию для компьютеров, подключенных к сети. Основная информация, которую предоставляет NIS, — это:

- имена для входа в систему/пароли/домашние каталоги (/etc/passwd);
- информация о группах (/etc/group).

Первоначально NIS была разработана фирмой Sun Microsystems, Inc. и носила название Yellow Pages. Однако из-за того, что Yellow Pages является торговой маркой, принадлежащей British Telecom, пришлось переименовать этот протокол.

Как работает NIS

Идеальная структура серверной части NIS — наличие нескольких серверов, среди которых один является главным (мастер-сервером), а все остальные являются подчиненными серверами (для определенного "домена" NIS). Вырожденный случай — один NIS-сервер.

Подчиненные серверы хранят только копии баз данных NIS и получают эти копии от мастер-сервера в тот момент, когда в базы данных на мастер-

сервере вносятся изменения. Такая структура подразумевает, что в том случае, когда NIS-сервер станет недоступен или будет перегружен, клиент NIS, подключенный к этому серверу, будет искать дублирующий NIS-сервер (рабочий или менее загруженный).

При изменении баз данных мастер-сервера подчиненные серверы будут извещены об этом посредством программы yppush, и автоматически получат необходимые изменения. Клиенты NIS не нуждаются в синхронизации, поскольку не производят кэширования данных.

Программа-сервер ypserv

Если вы используете ваш сервер как мастер, определите, какие файлы нужны вам для доступа через NIS, а затем добавьте или удалите соответствующие записи в правиле `all` в /var/yp/Makefile.

Теперь отредактируйте /var/yp/securenets и /etc/ypserv.conf. Убедитесь, что portmapper (portmap(8)) запущен и запустите сервер ypserv. Команда

```
rpcinfo -u localhost ypserv
```

должна выдать примерно следующее:

```
program 100004 version 1 ready and waiting
program 100004 version 2 ready and waiting
```

Строка `version 1` может быть опущена в зависимости от версии ypserv и настроек, которые вы используете.

Далее необходимо сгенерировать базу данных NIS (YP). На мастер-сервере запустите команду

```
ypinit -m
```

На подчиненном сервере убедитесь, что `ypwhich -m` работает. Это означает, что ваш подчиненный сервер должен быть настроен как клиент NIS перед запуском команды

```
ypinit -s masterhost
```

для установки этого узла как подчиненного сервера NIS.

Для обновления базы данных NIS запустите `make` в каталоге /var/yp на вашем мастер-сервере. Это приведет к ее обновлению и загрузке на подчиненные серверы, если ее исходный файл имеет более свежую дату.

NIS+

NIS+ (Network Information Service Plus, расширенная служба сетевой информации). Этот протокол предназначен для замены NIS и обратно совместим с ним. Однако, как и NIS, NIS+ является коммерческим и бесплатной

реализации для Linux не имеет. Под Linux реализован только клиент NIS+. Поэтому NIS+ в среде Linux не получил большого распространения и вместо него используется LDAP.

Как работает NIS+

NIS+ — это новая версия службы имен сетевой информации от Sun. Самое большое отличие между NIS и NIS+ это то, что NIS+ имеет поддержку шифрования данных и авторизацию через безопасный RPC.

Модель имен в NIS+ основывается на структуре дерева. Каждый узел в дереве соответствует одному объекту NIS+ из шести типов: каталог, запись, группа, ссылка, таблица и личное.

Каталог NIS+, формирующий главное пространство имен NIS+, называется корневым каталогом. Имеется два специальных каталога NIS+: org_dir и groups_dir. Каталог org_dir содержит все административные таблицы, такие как passwd (пароли), hosts (узлы) и mail_aliases (почтовые псевдонимы). Каталог groups_dir содержит объекты групп NIS+, которые используются для управления доступом. Совокупность org_dir, groups_dir и их родительского каталога называется доменом NIS+.

LDAP

LDAP (Lightweight Directory Access Protocol, облегченный протокол службы каталогов) основан на клиент-серверной модели.

Один или более серверов LDAP содержат данные, составляющие дерево каталога LDAP, или базу данных LDAP. Клиент LDAP подключается к LDAP-серверу и задает ему вопросы. Сервер выдает ответ или указатель места, где клиент может получить более подробную информацию (обычно другой LDAP-сервер). Не имеет значения, к какому LDAP-серверу подключается клиент, он видит один и тот же каталог.

Протокол LDAPv3 описывается в RFC2251-2256,2829-2831.

Далее рассматривается реализация LDAP, известная как OpenLDAP. Основные особенности OpenLDAP:

- поддержка LDAPv2 и LDAPv3;
- поддержка взаимодействия с существующими клиентами;
- поддержка IPv4 и IPv6;
- Strong Authentication (SASL) (RFC2829);
- Start TLS (RFC2830);
- Language Tags (RFC2596);
- служба расположения, основанная на DNS (RFC2247);

- усовершенствованный автономный сервер;
- Named References/ManageDsaIT;
- усовершенствованная подсистема контроля доступа;
- пул нитей (threads pool);
- поддержка приоритетов нитей;
- поддержка множества слушателей;
- LDIFv1 (RFC2849);
- усовершенствованное определение платформы/подсистемы.

Установка LDAP-сервера

Для установки LDAP-сервера помимо самого пакета OpenLDAP необходимо установить еще несколько пакетов.

- Библиотеки OpenSSL TLS. Обычно входят в состав системы или являются опциональным программным компонентом. Службы аутентификации Kerberos.
- Kerberos. Клиенты и серверы OpenLDAP поддерживают службы аутентификации на основе Kerberos. В частности, OpenLDAP поддерживает механизм аутентификации SASL/GSSAPI, используя либо пакет Heimdal, либо пакет MIT Kerberos V.
- Sleepycat Software BerkeleyDB или с Free Software Foundation's GNU Database Manager (GDBM). Если на время конфигурирования ни один из этих пакетов не доступен, вы не сможете построить slapd с поддержкой главного механизма работы с базой данных.

При наличии вышеперечисленных пакетов установка OpenLDAP не вызовет никаких трудностей.

Имя исполняемого файла сервера OpenLDAP — slapd.

Настройка LDAP-сервера

Для конфигурирования сервера необходимо в файле slapd.conf определить соответствующие переменные. Файл может располагаться либо в каталоге /etc, либо в /usr/local/etc/openldap. Далее приводятся наиболее общеупотребимые директивы файла slapd.conf. Полный список директив можно найти в соответствующем файле справки.

Формат конфигурационного файла

Файл slapd.conf состоит из трех типов конфигурационной информации: глобальной, специфичной для механизма базы данных и специфичной для базы данных. Глобальная информация указывается первой, за ней следует

информация, связанная с механикой базы данных, за которой располагается информация, связанная с отдельным экземпляром базы данных.

Глобальные директивы могут переопределяться в описании механизмов баз данных и/или директивах базы данных. Директивы механизма базы данных могут переопределяться директивами базы данных.

Пустые строки и строки с комментариями (начинаются с символа #) игнорируются. Если строка начинается с пробельного символа, она рассматривается как продолжение предыдущей строки. Общий формат файла slapd.conf приведен далее:

```
# глобальные конфигурационные директивы
<глобальные конфигурационные директивы>

# определение механизма базы данных
backend <typeA>
<специфичные для механизма базы данных директивы>

# определение базы данных и конфигурационные директивы
database <typeA>
<директивы, специфичные для базы данных>

# определение второй базы данных и конфигурационные директивы
database <typeB>
<директивы, специфичные для базы данных>

# определение третьей базы данных и конфигурационные директивы
database <typeA>
<директивы, специфичные для базы данных>

# последующие механизмы баз данных, определения баз данных и
# конфигурационные директивы
...
```

Конфигурационные директивы могут иметь аргументы, в этом случае они разделяются пробелами. Если аргумент содержит пробелы, он должен заключаться в кавычки. Если аргумент содержит двойную кавычку или обратный слэш (\), символ должен предваряться символом обратного слэша.

Глобальные директивы

Директивы, описанные в этой секции, применяются ко всем механизмам баз данных и базам данных, если они не переопределяются в определениях механизмов баз данных или баз данных. Аргументы, которые необходимо заменить соответствующим текстом, приведены в скобках <>.

- `access to <что> [ by <кому> <уровень доступа> <control> ]+` — директива предоставляет доступ (указанный в `<уровень доступа>`) к набору записей

и/или атрибутов (указанных в *<что>*) одному или более запрашивающих (указанных в *<кому>*).

- `attributetype` *<RFC2252 описание типа атрибута>* — эта директива определяет тип атрибута.
- `defaultaccess { none | compare | search | read | write }` — эта директива указывает доступ по умолчанию для всех запрашивающих, если не указана директива `access`. Любой назначенный уровень доступа включает в себя все нижележащие уровни доступа (например, доступ `read` предполагает также `search` и `compare`, но не `write`).
- `idletimeout` *<целое число>* — указывает период ожидания в секундах пред принудительным закрытием соединения с клиентом, находящегося в состоянии ожидания.
- `include` *<имя файла>* — эта директива указывает slapd перед чтением следующей строчки конфигурационного файла читать дополнительную конфигурационную информацию. Включаемый файл должен быть обычным файлом в формате конфигурационного файла slapd. Используется для включения файлов, содержащих спецификации схем.
- `loglevel` *<целое число>* — эта директива указывает уровень, на котором должны регистрироваться в syslog отладочные сообщения и статистика работы. Чтобы данная функция работала (за исключением двух уровней статистики, которые всегда включены), вам следует сконфигурировать OpenLDAP с опцией `enable-debug` (по умолчанию). Уровни доступа аддидивны. Для отображения номеров соответствующих определенному типу отладочной информации вызовите slapd с ключом `-?`.

  Для *<целое число>* возможны значения:

  - `-1` — включает всю отладочную информацию;
  - `0` — без отладочной информации;
  - `1` — трассировать вызовы функций;
  - `2` — отладка обработки пакетов;
  - `4` — тщательная отладочная трассировка;
  - `8` — управление соединением;
  - `16` — печать принятых и отправленных пакетов;
  - `32` — обработка фильтра поиска;
  - `64` — обработка конфигурационного файла;
  - `128` — обработка списка контроля доступа;
  - `256` — регистрировать статистику соединения/обработки/результатов;

- 512 — регистрировать статистику отправленных элементов;
- 1024 — печать коммуникаций с shell-механизмом баз данных;
- 2048 — печать отладки анализа элемента.

- objectclass <RFC2252 описание класса объектов> — эта директива определяет класс объектов.
- referral <URL> — эта директива определяет возвращаемую ссылку в случае, если slapd не может найти в локальной базе данных информацию для обработки запроса.
- sizelimit <целое число> — эта директива указывает максимальное количество элементов, возвращаемых одной операцией поиска.
- timelimit <целое число> — директива указывает максимальное число секунд, которые slapd затрачивает, отвечая на запрос. Если запрос не завершился за это время, будет возвращен результат, указывающий на истечение времени.

Общие опции механизмов баз данных

Директива в этой секции применима только к механизму базы данных, в котором она определена. Она поддерживается каждым типом механизма базы данных. Эта директива применима ко всем экземплярам баз данных одного типа и может переопределяться директивами баз данных.

- backend <тип> — эта директива отмечает начало определения механизма базы данных. <тип> должен быть одним из ldbm, shell, passwd или другим поддерживаемым типом механизмов базы данных.

Общие директивы базы данных

Директивы этой секции применимы только к базе данных, в которой они определены. Они поддерживаются всеми типами баз данных.

- database <тип> — директива отмечает начало нового экземпляра базы данных. <тип> должен быть одним из ldbm, shell, passwd или другим поддерживаемым типом базы данных.
- readonly { on | off } — эта директива переводит базу данных в режим "только чтение". Любые попытки модифицировать базу данных в режиме "только чтение" вызовут ошибку "unwilling to perform".
- replica host=<имя хоста>[:<порт>] [bindmethod={ simple | kerberos | sasl }] ["binddn=<отличительное имя>"] [mech=<механизм>] [authcid=<identity>] [authzid=<identity>] [credentials=<пароль>] [srvtab=<имя файла>] — эта директива указывает место для репликации базы данных. Параметр host= указывает хост и опционально порт подчиненного сервера LDAP. В качестве <имя хоста> может применяться либо имя домена, либо IP-адрес. Если <порт> не указан, то используется стандартный для

LDAP номер порта. Параметр `binddn=` указывает имя для привязки обновлений в подчиненном сервере. Атрибут `bindmethod` может иметь значение `simple`, `kerberos` или `sasl`, в зависимости от типа используемой для подключения к подчиненному slapd аутентификации: простая парольная аутентификация, Kerberos-аутентификация или SASL-аутентификация. Простая аутентификация требует указания параметров `binddn` и `credentials`. Kerberos-аутентификация требует указания параметров `binddn` и `srvtab`. Аутентификация SASL требует указания используемого механизма в параметре `mech`. В зависимости от механизма, может устанавливаться особенность аутентификации и/или удостоверение, параметрами `authcid` и `credentials` соответственно. Параметр `authcid` может использоваться для указания особенности аутентификации.

- `replogfile` *<имя файла>* — эта директива указывает имя регистрационного файла репликации, в котором сервер регистрирует изменения. Регистрационный файл репликации обычно записывается slapd и читается slurpd.
- `rootdn` *<отличительное имя>* — указывает отличительное имя, которое не подлежит контролю доступа или административным ограничениям при операциях в этой базе данных. Не требуется, чтобы отличительное имя ссылалось на элемент каталога. Отличительное имя может ссылаться на SASL.

  Пример с элементом:

  ```
  rootdn "cn=Manager, dc=example, dc=com"
  ```

  Пример с SASL:

  ```
  rootdn "uid=root@EXAMPLE.COM"
  ```

- `rootpw` *<пароль>* — эта директива указывает пароль приведенного выше отличительного имени, который будет всегда работать, не зависимо от того, существует ли элемент с данным отличительным именем и есть ли у него пароль.
- `suffix` *<суффикс отличительного имени>* — указывает суффикс отличительного имени при запросах к этому механизму баз данных. Можно указывать несколько строк. Для каждой базы данных требуется хотя бы одна строка.
- `updatedn` *<отличительное имя>* — эта директива применима только к подчиненному slapd. Она указывает отличительное имя, которому разрешено внесение изменений в реплику. Это может быть отличительное имя, к которому привязывается slurpd при внесении изменений в реплику, или отличительное имя, ассоциированное с SASL.
- `updateref` *<URL>* — эта директива применима только к подчиненному серверу. Она указывает URL, возвращаемый клиентам, которые получа-

ют запросы обновления на реплику. Если она указана несколько раз, то предоставляется каждый URL.

Директивы, специфичные для LDBM-механизма базы данных

Директивы этой категории применимы только к механизму базы данных LDBM. Они должны следовать за строкой `database ldbm` и перед любой другой database-строкой.

- `cachesize` *<целое число>* — эта директива указывает размер поддерживаемого экземпляром механизма базы данных LDBM кэша элементов в памяти.
- `dbcachesize` *<целое число>* — эта директива указывает в байтах размер кэша в памяти, связанного с каждым открытым индексным файлом. Если она не поддерживается методом нижележащей базы данных, то она игнорируется без комментариев. Увеличение этого числа приводит к увеличению использования памяти, но приводит к резкому повышению производительности, особенно при модификации или перестроении индексов.
- `dbnolocking` — если присутствует эта опция, то она отменяет блокировку базы данных. Включение такой опции может привести к увеличению производительности ценой безопасности данных.
- `dbnosync` — эта опция приводит к отложенной синхронизации изменений в памяти с содержимым базы данных на диске. Ее включение может увеличить производительность ценой безопасности данных.
- `directory` *<каталог>* — эта директива указывает каталог, в котором находятся файлы, содержащие базу данных LDBM и связанные с ними индексные файлы.
- `index {`*<список атрибутов>*` | default} [pres, eq, approx, sub, none]` — эта директива указывает индексы для хранения данных атрибутов. Если указан только *<список атрибутов>*, то поддерживаются только индексы по умолчанию.
- `mode` *<целое число>* — эта директива указывает режим защиты вновь создаваемых индексных файлов базы данных.

Прочие механизмы баз данных

Slapd поддерживает и другие типы механизмов баз данных (кроме LDBM):

- `ldbm` — Berkeley или GNU DBM-совместимый механизм;
- `passwd` — обеспечивает доступ к /etc/passwd в режиме "только для чтения";
- `shell` — shell-механизм (доступ к внешней программе)
- `sql` — механизм базы данных на основе SQL.

Ключи командной строки

Slapd поддерживает несколько ключей командной строки.

- `-f <имя файла>` — этот ключ указывает альтернативный файл конфигурации slapd.
- `-h <URL>` — этот ключ определяет конфигурацию слушателя. По умолчанию ldap:/// — что подразумевает использование протокола LDAP поверх TCP на всех интерфейсах, определенных при конфигурировании. Вы можете указать пару "хост-порт" или другие схемы протокола. Хост можно указывать в форме IPv4-чисел, разделенных точками, или в форме имени хоста. Значение номера порта должно быть числом.
- `-n <имя службы>` — этот ключ определяет имя службы, используемой для регистрации и других целей. Служба по умолчанию — slapd.
- `-l <локальный пользователь syslog>` — этот ключ определяет локального пользователя для функции `syslog`.
- `-u <пользователь> -g <группа>` — этот ключ определяет соответственно пользователя и группу, от чьего имени происходит запуск программы. Пользователь может быть либо именем, либо uid. Группа может быть либо именем группы, либо gid.
- `-r <каталог>` — этот ключ определяет каталог запуска. После открытия слушателей, но до чтения каких-либо конфигурационных файлов или инициализации механизмов баз данных, slapd выполнит для этого каталога chroot.
- `-d <уровень> | ?` — этот ключ устанавливает уровень отладки. Если используется символ `?`, выводятся различные уровни отладки, и slapd завершается, вне зависимости от любых других приложенных ключей.

  Текущие уровни отладки:

  - `-1` — включает всю отладочную информацию;
  - `0` — без отладки;
  - `1` — трассировать вызовы функций;
  - `2` — отладка обработки пакетов;
  - `4` — тщательная отладочная трассировка;
  - `8` — управление соединением;
  - `16` — печать принятых и отправленных пакетов;
  - `32` — обработка фильтров поиска;
  - `64` — обработка конфигурационного файла;
  - `128` — обработка списка контроля доступа;

- `256` — регистрировать статистику соединения/обработки/результатов;
- `512` — регистрировать статистику отправленных элементов;
- `1024` — печать коммуникаций с shell-механизмом базы данных;
- `2048` — печать отладки анализа элемента.

Вы можете активировать несколько уровней, указав по одному отладочному уровню в каждом ключе, или вычислить число самостоятельно.

База данных LDAP

Как уже упоминалось ранее, LDAP имеет свою базу данных. Далее мы рассмотрим основные моменты структуры баз данных.

Механизмы баз данных LDAP, объекты и атрибуты

Slapd может работать с тремя различными механизмами баз данных.

- LDBM — высокоскоростная дисковая база данных.
- SHELL — интерфейс базы данных к обычным UNIX-командам и shell-скриптам.
- PASSWD — простая база данных на основе файла паролей.

LDAP по умолчанию использует LDBM-базу данных, что подразумевает повышенную производительность.

База данных LDBM назначает уникальный четырехбайтовый идентификатор каждому элементу базы данных. Она использует этот идентификатор для ссылок на записи из индексов. База данных состоит из одного главного файла индексов, который устанавливает соответствие уникального индекса элемента (EID) текстовому представлению самого элемента. Также поддерживаются другие индексные файлы.

Для импортирования или экспортирования информации каталога между серверами каталогов, основанных на LDAP, или для описания набора вносимых в каталог изменений обычно используется формат LDIF (LDAP Data Interchange Format, LDAP-формат обмена данных). Файл LDIF хранит информацию об объектно-ориентированной иерархии элементов. Пакет LDAP, который вы собираетесь использовать, поставляется с утилитой конвертирования LDIF-файлов в LDBM-формат.

Типичный LDIF-файл выглядит так:

```
dn: o=Home, c=UA
o: Home
objectclass: organization
dn: cn=Vasya Pupkin, o=Home, c=UA
cn: Vasya Pupkin
sn: Pupkin
```

```
mail: vasya@yahoo.com
objectclass: person
```

Вы можете заметить, что каждый элемент уникально идентифицируется отличительным именем DN, которое состоит из имени элемента плюс путь имен, ведущих к вершине иерархии каталога.

В LDAP класс объектов определяет набор атрибутов, которые используются для определения элемента. Стандарт LDAP определяет такие основные виды классов объектов:

- группа каталогов, включая неупорядоченный список индивидуальных объектов или групп объектов;
- местоположение, такое как имя страны и описание;
- организации в каталоге;
- люди в каталоге.

Элемент может принадлежать более чем одному классу. Например, элемент человека определен классом объектов `person`, но также могут быть определены атрибуты в классах объектов `inetOrgPerson`, `groupOfNames` и `organization`. Структура классов объектов сервера (его схема) определяет общий список требуемых и разрешенных атрибутов отдельного элемента.

Данные каталога представлены в виде пар "атрибут-значение". Любая определенная часть информации ассоциируется с этим описательным атрибутом.

Например, атрибут `cn` (commonName) используется для размещения имени человека.

Каждый человек, введенный в каталог, определяется набором атрибутов в классе объектов `person`.

Требуемые атрибуты включают атрибуты, которые могут представляться в элементах, используя класс объектов. Все элементы требуют наличия атрибута `objectClass`. В нем перечислены классы объектов, к которым принадлежит элемент.

Разрешенные атрибуты включают атрибуты, которые могут быть представлены в элементах класса объектов. Например, в классе объектов `person` обязательны атрибуты `cn` и `sn`. Атрибуты `description`, `telephoneNumber`, `seeAlso` и `userpassword` разрешены, но не обязательны.

Каждый атрибут имеет соответствующее синтаксическое определение. Синтаксическое определение описывает тип предоставляемой атрибутом информации:

- `bin` (binary) — двоичный;
- `ces` (case exact string) — строка с соответствием регистра (регистр символов должен совпадать при сравнении);

- `cis` (case ignore string) — строка с игнорированием регистра (регистр символов игнорируется при сравнении);
- `tel` — строка с номером телефона (подобен `cis`, но пробелы и тире '-' игнорируются при сравнении);
- `dn` (distinguished name) — отличительное имя.

Создание и поддержание базы данных

Есть два способа создания базы данных. Во-первых, вы можете создать базу данных в реальном времени, используя LDAP. Таким образом, вы просто запускаете slapd и добавляете элементы, используя любой LDAP-клиент на ваш выбор. Этот способ хорош для относительно небольших баз данных.

Второй метод создания базы данных — сделать это автономно, используя средства генерации индексов, — является наилучшим методом для случая, когда вам нужно создать тысячи элементов.

Создание базы данных в реальном времени

Программный пакет OpenLDAP поставляется с утилитой ldapadd, которая используется для добавления записей при запущенном LDAP-сервере. Если вы выбрали создание базы данных в реальном времени, вы можете использовать ldapadd для добавления элементов. После первичного добавления элементов вы все еще можете использовать ldapadd для добавления элементов. Перед запуском sldap вы должны проверить, что в вашем файле sldap.conf установлены определенные конфигурационные опции.

```
suffix <отличительное имя>
```

Настоящая опция указывает, какие элементы помещены в этой базе данных. Вы должны ее установить в значение отличительного имени корня поддерева, которое вы собираетесь создать. Например:

```
suffix "o=Home, c=UA"
```

Нужно проверить, указали ли вы каталог, где должны быть созданы индексные файлы:

```
directory <каталог>
```

Например:

```
directory /usr/local/home
```

В заключение вы должны удостовериться, что определение базы данных содержит определение необходимых вам индексов:

```
index {<attrlist> | default} [pres,eq,approx,sub,none]
```

Например, для индексации атрибутов `cn`, `sn`, `uid` и `objectclass` можно использовать следующие строки конфигурации индексов.

```
index cn,sn,uid
index objectclass pres,eq
index default none
```

После конфигурирования сервера запустите slapd, подключитесь к нему с помощью LDAP-клиента и добавляйте элементы.

Автономное создание базы данных

Второй метод создания базы данных — использование утилиты генерирования индексов. Этот метод предпочтителен при создании большого объема индексов. Утилиты считывают конфигурационный файл slapd, содержащий текстовое представление добавляемых элементов, и входной LDIF-файл. Они создают непосредственно индексный файл LDBM. Есть несколько важных конфигурационных опций, которые необходимо добавить при определении базы данных в конфигурационном файле:

```
suffix <отличительное имя>
```

Как описано в предыдущем разделе, эта опция указывает, какие элементы содержатся в базе данных. Вы должны установить ее в значение отличительного имени корня создаваемого вами поддерева. Например:

```
suffix "o=Home, c=UA"
```

Нужно проверить, указали ли вы каталог, где должны быть созданы индексные файлы:

```
directory <каталог>
```

Например:

```
directory /usr/local/home
```

Далее увеличим размер внутреннего кэша, используемого каждым открытым файлом. Для наилучшей производительности при создании индекса весь индекс должен поместиться в память. Этот размер устанавливается следующей опцией:

```
dbcachesize <целое число>
```

Например, посредством следующей опции будет создан кэш на 50 Мбайт:

```
dbcachesize 50000000
```

Далее нужно указать, какие индексы вы хотите создать. Это делается одной или более опциями.

```
index {<attrlist> | default} [pres,eq,approx,sub,none]
```

Например:

```
index cn,sn,uid pres,eq,approx
index default none
```

Будут созданы индексы `presence`, `equality` и `approximate` для атрибутов `cn`, `sn` и `uid`, а для других атрибутов не будет создано никаких индексов.

Как только вы настроили все, как хотели, вы создаете первичную базу данных и ассоциированные индексы программой slapadd:

```
slapadd -l <входной файл> -f <конфигурационный файл slapd>
  [-d <уровень отладки>] [-n <целое число>|-b <суффикс>]
```

Аргументы имеют следующие значения:

- `-l <входной файл>` — определяет входной LDIF-файл, содержащий добавляемые элементы в текстовой форме.
- `-f <конфигурационный файл slapd>` — указывает конфигурационный файл slapd, в котором определено: где создавать индексы, какие индексы создавать и т. д.
- `-d <уровень отладки>` — включает отладку, в соответствии с `<уровень отладки>`. Уровни отладки такие же, как и для slapd.
- `-n <номер базы данных>` — необязательный аргумент. Указывает, какую базу данных модифицировать. Первая по списку база данных в конфигурационном файле считается `1`, вторая — `2` и т. д. Не должен использоваться вместе с ключом `-b`.
- `-b <суффикс>` — необязательный аргумент, указывающий, какую базу данных модифицировать. Представленный суффикс сопоставляется с суффиксом базы данных и при совпадении определяется номер модифицируемой базы данных. Не должен использоваться вместе с ключом `-n`.

Утилиты

В этом разделе кратко описаны утилиты для добавления, удаления и модификации записей в LDAP.

Slapindex

Иногда возникает необходимость перегенерировать индексы (например, после модификации slapd.conf). Это можно сделать используя программу slapindex.

Slapcat

Программа slapcat используется для дампа базы данных в LDIF-файл. Она может быть полезна, когда вы хотите создать читаемую резервную копию вашей базы данных или хотите отредактировать вашу базу данных автономно.

Ldapsearch

Ldapsearch — это интерфейс shell к библиотечному вызову ldap_search. Используйте утилиту для поиска элементов в вашем LDAP-механизме базы данных.

Ldapsearch открывает соединение с сервером LDAP, присоединяется и выполняет поиск с фильтром, заданным пользователем. Фильтр должен соответствовать строковому представлению фильтров LDAP, как определено в RFC1558. Если ldapsearch находит один или более элементов, то извлекаются атрибуты, элементы и их значения печатаются в стандартный вывод.

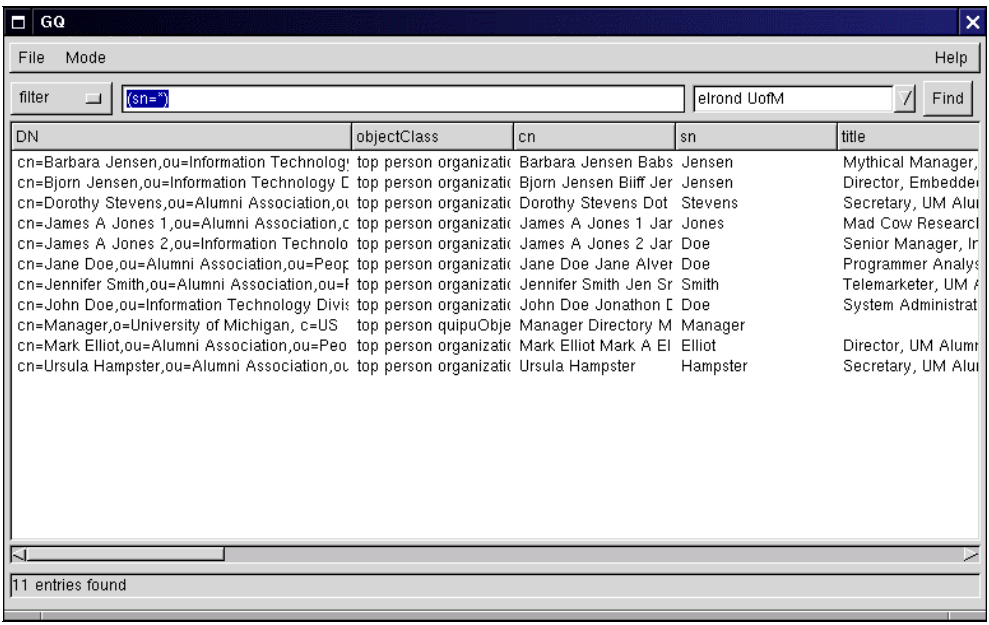


Рис. 7.1. Окно поиска

Ldapdelete

Ldapdelete — это shell-интерфейс к библиотечному вызову ldap_delete. Утилита используется для удаления элементов из вашего LDAP-механизма базы данных.

Ldapdelete открывает соединение к LDAP-серверу, подключается и удаляет один или более элементов. Если приложен один или более аргумент `dn`, элементы с этим отличительными именами будут удалены. Каждое отличительное имя должно быть строковым представлением отличительного имени в соответствии с RFC1779. Если вызов был сделан без аргументов, список отличительных имен читается со стандартного.

Ldapmodify

Ldapmodify — это shell-интерфейс к библиотечным вызовам ldap_modify и ldap_add. Утилита используется для изменения содержимого элементов вашего LDAP-механизма баз данных.

Ldapadd

Ldapadd — это просто жесткая ссылка на утилиту ldapmodify. При вызове ldapadd автоматически включается ключ `-a` (добавить элемент) утилиты ldapmodify.

Kldap

Kldap — графический LDAP-клиент для KDE. Отображает информационное дерево каталога.

GQ

GQ — графический LDAP-клиент для GNOME с простым интерфейсом.

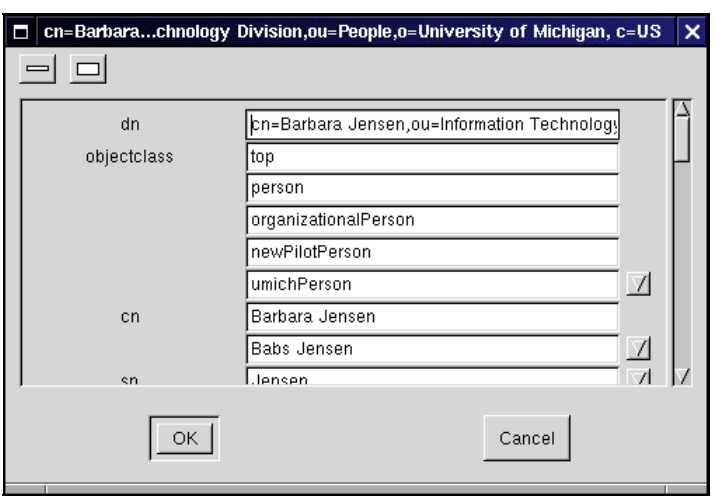


Рис. 7.2. Редактирование элемента

Взаимодействие программ с LDAP

Наиболее универсальный способ взаимодействия программ с LDAP — проведение аутентификации через модули PAM. Модуль PAM называется pam_ldap и входит в большинство современных дистрибутивов. Модуль pam_ldap использует конфигурационный файл ldap.conf.

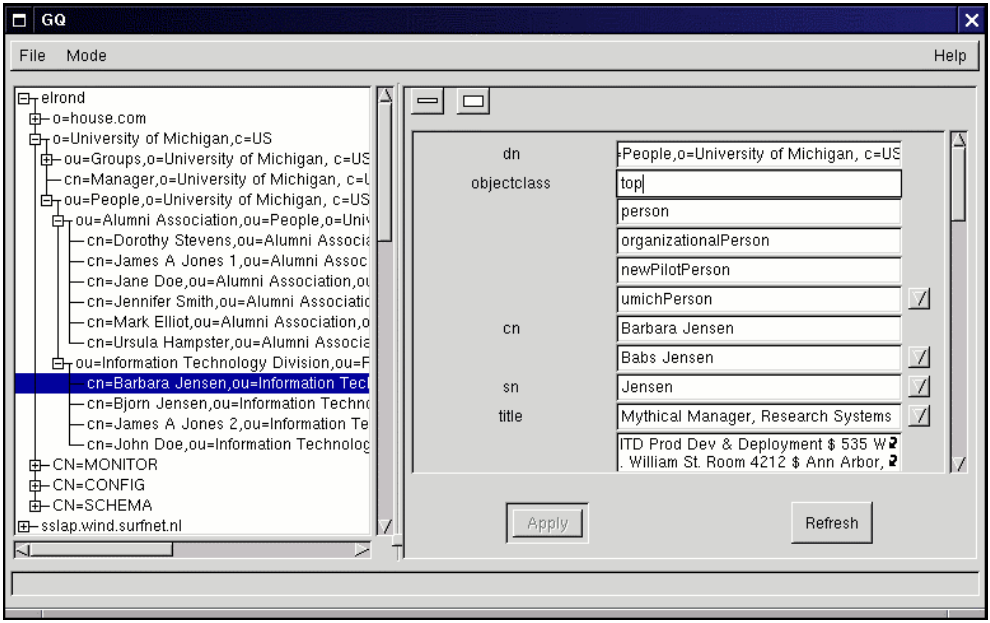


Рис. 7.3. Просмотр дерева каталогов

В простейшем случае при настройке клиента /etc/ldap.conf содержит 3 строки:

```
BASE dc=home,dc=ua
HOST 192.168.0.1
pam_password clear
```

- `BASE` — база для поиска в дереве LDAP;
- `HOST` — IP-адрес или имя хоста, на котором работает LDAP-сервер;
- `pam_password` — определяет тип шифрования паролей.

Основной конфигурационный файл PAM находится в каталоге /etc и называется pam.conf. Он состоит из правил, описывающих методы проведения аутентификации для различных сервисов. В этом файле необходимо прописать записи для соответствующих программ, которые хотят при аутентификации использовать pam_ldap.

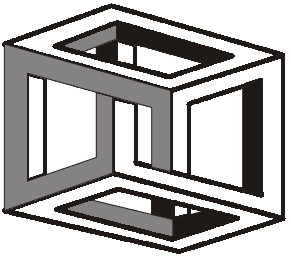
Замечание

Некоторые программы напрямую взаимодействуют с LDAP, без помощи PAM, например SQUID.

Литература и ссылки

- Немет Э., Снайдер Г., Сибасс С., Хейн Т. Р. UNIX: руководство системного администратора. Для профессионалов: Пер. с англ. — СПб.: Питер; К.: Издательская группа BHV, 2002.
- LDAP Linux HOWTO.
- The Linux NIS(YP)/NYS/NIS+ HOWTO.
- Man-страница lapd.conf.
- RFC1558: A String Representation of LDAP Search Filters.
- RFC1777: Lightweight Directory Access Protocol.
- RFC1778: The String Representation of Standard Attribute Syntaxes.
- RFC1779: A String Representation of Distinguished Names.
- RFC1781: Using the OSI Directory to Achieve User Friendly Naming.
- RFC1798: Connectionless LDAP.
- RFC1823: The LDAP Application Programming Interface.
- RFC1959: An LDAP URL Format.
- RFC1960: A String Representation of LDAP Search Filters.
- RFC2251: Lightweight Directory Access Protocol (v3).
- RFC2307: LDAP as a Network Information Service.
- **http://www.keldysh.ru/metacomputing/ism99.html** — Валиев М. К., Китаев Е. Л., Слепенков М. И. Использование службы директорий LDAP для представления метаинформации в глобальных вычислительных системах. ИПМ им. М. В. Келдыша. РАН.
- **http://www.openldap.org** — официальный сайт OpenLDAP.
- **http://www.opennet.ru/docs/RUS/ldap/index.html** — Захаров М. Руководство по настройке аутентификации пользователей посредством LDAP.

ГЛАВА 8



FTP

FTP — протокол передачи файлов, возникший очень давно (в конце 70-х— начале 80-х), но до сих пор остающийся очень популярным. В этой главе мы поговорим о самом протоколе, программах, его использующих, а также о настройке сервера FTP.

Протокол FTP

Протокол FTP (File Transfer Protocol, протокол передачи файлов) предназначен для передачи файлов в сети Интернет. Он несколько устарел, частично функции передачи файлов взял на себя Web-протокол HTTP, но, несмотря на это, протокол FTP, похоже, будет использоваться еще долгое время.

Протокол FTP, в отличие от большинства других протоколов, для пересылки файла использует два TCP-соединения. Одно соединение для передачи файла, а второе — для управления процессом передачи. Порт 20 предназначен для пересылки данных, а порт 21 для управляющего соединения. FTP-протокол может использовать как TCP-соединение, так и UDP-соединение.

Представление данных

Протокол передачи файлов допускает различные способы представления файлов и управления передачей. Далее приведены критерии, от выбора которых зависит корректность передачи файлов по протоколу FTP.

Тип файла

Протокол должен знать, каков тип передаваемого файла. От этого зависит корректное представление его на компьютере (и в операционной системе) получателя.

- *ASCII-файлы*. Текстовый файл передается как NVT ASCII. При этом требуется, чтобы программа-отправитель конвертировала текстовый файл в NVT ASCII, а программа-получатель производила обратное преобразование. Конец каждой строки передается в виде NVT ASCII-символа возврата каретки (CR), после которого следует перевод строки (LF). Если отправитель текстового файла установит тип файла как бинарный, программы не будут преобразовывать передаваемый файл. Это вызывает проблемы несовместимости между текстовыми файлами DOS/Windows и текстовыми файлами UNIX. В DOS/Windows принято конец текстовой строки обозначать парой символов возврат каретки/перевод строки (CR/LF), а в UNIX — перевод строки (LF).
- *EBCDIC-файлы*. Альтернативный способ передачи текстовых файлов.
- *Бинарные файлы*. Данные между FTP-сервером и клиентом передаются как непрерывный поток битов.
- *Локальный тип файлов*. Способ передачи бинарных файлов между компьютерами, которые имеют различный размер байта. Количество битов в байте определяется отправителем. Обычно не используется.

Управление форматом

Применяется только для передачи ASCII- и EBCDIC-файлов.

- *Nonprint*. Файл не содержит информацию вертикального форматирования.
- *Telnet format control*. Файл содержит управляющие символы вертикального форматирования Telnet, которые интерпретируются принтером.
- *Fortran carriage control*. Первый символ каждой строки — это Fortran-символ управления форматированием.

Структура

Существует несколько способов передачи структуры данных.

- *Структура файла*. Пересылаемый файл воспринимается в виде непрерывного потока байтов.
- *Структура записи*. Эта структура используется только в случае текстовых файлов.
- *Структура страницы*. Каждая страница передается с номером страницы. (Не рекомендуется использовать эту структуру.)

Режим передачи

Определяет способ передачи файла по соединению данных.

- *Потоковый режим*. Передача файла осуществляется как поток байтов.
- *Блочный режим*. Передача файла осуществляется как последовательность блоков. Блок имеет управляющий заголовок и собственно пересылаемые данные.

- *Режим сжатия.* При передаче осуществляется замена неоднократно встречающихся повторяющихся байтов на байт и число его повторений.

Как видите, протокол FTP поддерживает большое количество представлений данных. Однако в реальной жизни большинством программного обеспечения наиболее часто используется приведенное далее ограниченное подмножество представлений:

- тип файла — ASCII или двоичный;
- управление форматом — только nonprint;
- структура — только структура файла;
- режим передачи — только потоковый режим.

Управляющие команды FTP

Управляющие команды и ответы передаются по управляющему соединению между клиентом и сервером в формате NVT ASCII. В конце каждой строки присутствует пара символов возврат каретки/перевод строки (CR/LF).

В полном наборе команд насчитывается более 30. В табл. 8.1 приведены наиболее часто используемые команды. Полный список команд можно посмотреть в соответствующем RFC.

Таблица 8.1. *Управляющие команды протокола FTP*

Команда	Описание
`ABOR`	Прервать последнюю команду FTP и любую передачу данных
`LIST <`*`список файлов`*`>`	Список файлов или каталогов
`PASS <`*`пароль`*`>`	Передача пароля пользователя
`PORT a, b, c, d, e, f`	IP-адрес клиента (`a.b.c.d`) и порт (`e×256 + f`)
`QUIT`	Разорвать соединение
`RETR <`*`имя файла`*`>`	Получить файл
`STOR <`*`имя файла`*`>`	Выгрузить файл
`SYST`	Сервер возвращает тип системы
`TYPE <`*`тип`*`>`	Указать тип файла: `A` — для ASCII, `I` — для бинарного
`USER <`*`имя пользователя`*`>`	Передача имени пользователя (логин)

Ответы на управляющие FTP-команды

Ответы на управляющие FTP-команды состоят из трехзначного числа в формате ASCII и необязательного текстового сообщения, которое следует за числом.

Каждая из трех цифр в коде ответа имеет собственное значение. Расшифровка первой и второй цифр кода приведена в табл. 8.2.

Таблица 8.2. *Значения первой и второй цифр в коде ответа на управляющие команды*

Ответ	Описание
1*xx*	Положительный предварительный отклик. Действие началось, однако необходимо дождаться еще одного отклика перед отправкой следующей команды
2*xx*	Положительный отклик о завершении. Может быть отправлена новая команда
3*xx*	Положительный промежуточный отклик. Команда принята, однако необходимо отправить еще одну команду
4*xx*	Временный отрицательный отклик о завершении. Требуемое действие не произошло, однако ошибка временная, поэтому команду необходимо повторить позже
5*xx*	Постоянный отрицательный отклик о завершении. Команда не была воспринята и повторять ее не стоит
*x*0*x*	Синтаксическая ошибка
*x*1*x*	Информация
*x*2*x*	Соединения. Отклики имеют отношение либо к управляющему соединению, либо к соединению данных
*x*3*x*	Аутентификация и бюджет. Отклик имеет отношение к регистрации пользователя в системе или командам, связанным с бюджетом
*x*4*x*	Не определено
*x*5*x*	Состояние файловой системы

Третья цифра дает уточняющее определение сообщению об ошибке. В табл. 8.3 приведены коды и пояснения.

Таблица 8.3. *Значения третьей цифры в коде ответа на управляющие команды*

Ответ	Описание
125	Соединение данных уже открыто; начало передачи
200	Команда выполнена
214	Сообщение о помощи
331	Имя пользователя принято, необходимо ввести пароль
425	Невозможно открыть соединение данных

Таблица 8.3 *(окончание)*

Ответ	Описание
452	Ошибка записи файла
500	Неизвестная команда
502	Нереализованный тип MODE

Обычно каждая FTP-команда генерирует однострочный ответ. Если необходим ответ, состоящий из нескольких строк, то первая строка содержит дефис вместо пробела после трехзначного кода отклика, а последняя строка содержит тот же самый трехзначный код отклика, за которым следует пробел.

Управление соединением

Использовать соединение данных можно тремя способами:

- отправка файлов от клиента к серверу;
- отправка файлов от сервера к клиенту;
- отправка списка файлов или каталогов от сервера к клиенту.

Третий способ необходимо пояснить. FTP-сервер посылает список файлов по соединению данных — при таком использовании канала данных появляется возможность избежать любых ограничений в строках, накладывающихся на размер списка каталога, и несколько упрощает обмен информацией.

Управляющее соединение остается в активизированном состоянии все время, пока установлено соединение "клиент-сервер", но соединение данных может устанавливаться и отключаться по необходимости. Рассмотрим, как выбираются номера портов для соединения данных и кто осуществляет активное открытие, а кто пассивное.

Основной режим передачи — потоковый режим. В этом режиме конец файла обозначает закрытие соединения данных. Из этого вытекает, что для передачи каждого файла требуется новое соединение данных. Обычная процедура выглядит следующим образом:

1. Создание соединения данных осуществляется клиентом.
2. Клиент выбирает динамически назначаемый номер порта на компьютере клиента для своего конца соединения данных и осуществляет пассивное открытие с этого порта.
3. Клиент посылает номер порта на сервер по управляющему соединению с использованием команды PORT.
4. Сервер принимает номер порта с управляющего соединения и осуществляет активное открытие на этот порт компьютера клиента. Сервер всегда использует порт 20 для соединения данных.

Сервер всегда осуществляет *активное открытие* соединения данных. Обычно сервер также осуществляет *активное закрытие* соединения данных, за исключением тех случаев, когда клиент отправляет файл на сервер в потоковом режиме, который требует, чтобы клиент закрыл соединение.

Если клиент не выдает команду `PORT`, сервер осуществляет активное открытие на тот же самый номер порта, который использовался клиентом для управляющего соединения.

Программное обеспечение

Для работы по протоколу FTP необходимо две программы — сервер и клиент. Клиентских программ — очень много: от простейших, работающих в командной строке, до имеющих весьма развитый графический интерфейс. Любой современный Web-браузер способен выступать в роли FTP-клиента. Поэтому на клиентских программах останавливаться не будем, а перейдем сразу к программному обеспечению сервера FTP.

Сегодня стандартом де-факто для множества дистрибутивов является использование в качестве программного обеспечения пакета wu-ftp (Washington University at Saint Louis FTP daemon).

Пакет wu-ftp

Программный пакет wu-ftp написан в Вашингтонском университете. Обычно поставляется вместе с дистрибутивом, поэтому установка его не представляет сложности.

Команды

Как уже упоминалось ранее, FTP-серверы имеют свои наборы команд, иногда несколько отличающиеся друг от друга. В табл. 8.4 приведен список команд сервера wu-ftp.

Таблица 8.4. *Стандартные команды FTP-сервера wu-ftp*

Команда	Описание
`ABOR`	Прервать предыдущую команду
`APPE`	Добавить к файлу
`CDUP/XCUP`	Подняться на каталог вверх
`CWD /XCWD`	Поменять текущий каталог
`DELE`	Удалить файл
`HELP`	Получить справочную информацию

Таблица 8.4 *(окончание)*

Команда	Описание
LIST	Получить список файлов и каталогов в текущем каталоге
MKD /XMKD	Создать каталог
MDTM	Показать время последнего изменения файла
MODE	Задать режим пересылки файла
NLST	Получить список файлов
PASS	Передать пароль пользователя
PASV	Вход в "пассивный" режим передачи
PORT	Задает порт для последующей передачи данных
QUIT	Окончание сеанса
REST	Продолжить прерванную передачу данных
RETR	Получить файл
RMD/XRMD	Удалить каталог
RNFR	Исходное имя переименовываемого файла
RNTO	Новое имя переименовываемого файла
SIZE	Получить размер файла
STAT	Показать состояние сервера
STOR	Сохранить файл
STOU	Сохранить файл с уникальным именем
STRU	Задает структуру передачи
SYST	Вывести тип операционной системы, на которой работает сервер
TYPE	Задать тип передачи
USER	Передать имя пользователя

Помимо перечисленных ранее команд, сервер wu-ftp имеет несколько специфичных. Они приведены в табл. 8.5.

Таблица 8.5. *Нестандартные команды FTP-сервера wu-ftp*

Команда	Описание
SITE EXEC	Запустить программу на выполнение
SITE GROUP	Сменить группу

Таблица 8.5 *(окончание)*

Команда	Описание
SITE GPASS	Передать пароль группы
SITE IDLE	Задать время неактивности пользователя, по истечении которого соединение разрывается
SITE MINFO	Показать список файлов более новых, чем указанная дата. Команда выдает более расширенную информацию, чем NEWER
SITE NEWER	Показать список файлов более новых, чем указанная дата
SITE UMASK	Задать umask для файлов, сохраняемых пользователем на сервере

Конфигурирование сервера

Конфигурирование сервера wu-ftp проводится в два этапа. Первый — компилирование сервера со специфическими для вашего случая свойствами. Этот вариант мы описывать не будем, поскольку для создания простого сервера достаточно пакета rpm, входящего в дистрибутив. Второй этап — редактирование конфигурационных файлов сервера.

Как вы уже знаете, конфигурационные файлы находятся в каталоге /etc. Сервер wu-ftp использует следующие конфигурационные файлы:

- ftpaccess;
- ftpservers;
- ftpusers;
- ftphosts;
- ftpgroups;
- ftpconversion.

Рассмотрим подробно каждый конфигурационный файл.

Файл ftpaccess

Этот конфигурационный файл, используется для определения прав доступа к серверу. Здесь определяется, какие и сколько пользователей могут получить доступ к серверу, а также важные элементы настройки безопасности сервера.

Рассмотрим подробно конфигурационные параметры, используемые в этом файле.

Управление правами доступа:

- autogroup *<имя_группы>* *<класс>* ... — в том случае, если анонимный пользователь является членом указанного класса, то сервер использует заданную группу, что позволяет анонимным пользователям из разных классов получать доступ к различным наборам каталогов;

- class <класс> typelist <шаблон_адресов> ... — позволяет закрепить клиента за указанным классом, исходя из IP-адреса и типа клиента, где:
  - typelist — список из ключевых слов, обычно anonymous, guest и real (зарегистрированные на локальном хосте — /etc/passwd), через запятую;
  - <шаблон_адресов> — шаблон имени или адреса хоста клиента или адрес:маска или имя файла (имя файла должно начинаться с /, а файл — содержать шаблоны адресов);
- deny <шаблон_адресов> <файл_с_текстом_сообщения> — запретить доступ клиентов с указанного адреса с выдачей текста сообщения;
- guestgroup <имя_группы> ... — если реальный пользователь является членом указанной группы, то с ним поступают так же, как с анонимным. Вместо имени группы можно использовать номер группы, перед которым надо поставить знак процента, или интервал номеров, или звездочку для всех групп;
- guestuser <имя_пользователя> ... — аналогично guestgroup, но используется имя реального пользователя;
- realgroup <имя_группы> ... — инвертирует действие guestgroup и guestuser;
- realuser <имя_пользователя> ... — инвертирует действие guestgroup и guestuser;
- defumask umask [<класс>] — задание umask, применяемой при создании файлов;
- keepalive { yes | no } — установить TCP SO_KEEPALIVE;
- timeout accept <секунд> — сколько ожидать входного соединения для передачи данных (PASV);
- timeout connect <секунд> — сколько ожидать установления выходного соединения для передачи данных (PORT);
- timeout data <секунд> — максимальный период неактивности пользователя при передаче данных;
- timeout idle <секунд> — сколько ожидать следующей команды;
- timeout maxidle <секунд> — поскольку клиент имеет возможность установить idle самостоятельно, параметр maxidle позволяет установить верхний предел для клиента;
- timeout RFC931 <секунд> — максимальное время ожидания ответа для протокола ident;
- file-limit [raw] { in | out | total } <число> [<класс>] — ограничивает число передаваемых файлов;

- `byte-limit [ raw ] { in | out | total } <число> [ <класс> ]` — ограничивает число передаваемых байтов;
- `limit-time { * | anonymous | guest } <минут>` — ограничение времени сессии. Реальные пользователи не ограничиваются никогда;
- `guestserver [ <имя_серверного_хоста> ]` — гостевой и анонимный доступ предоставляется только к указанному хосту. Имеет смысл, если сервер обслуживает несколько виртуальных доменов;
- `limit <класс> <число> <временной_интервал> <имя_файла_с_сообщением>` — ограничение на число одновременно работающих клиентов из данного класса. Проверка производится только в момент входа. Если к сеансу применимо несколько команд `limit`, то используется первая;
- `noretrieve [ absolute | relative ] { class=<класс> } <имя_файла> ...` — запретить клиенту читать указанные файлы. Если имя начинается с /, то только этот файл, иначе любой файл с соответствующим именем. Если указан каталог, то любой файл из этого каталога;
- `allowretrieve [ absolute | relative ] { class=<класс> } <имя_файла> ...` — отменить действие директивы `noretrieve`;
- `loginfails <число>` — после указанного числа неудачных попыток зайти на сервер, сделать запись в журнале и разорвать соединение;
- `private { yes | no }` — нестандартные команды `SITE GROUP` и `SITE GPASS` позволяют пользователю поменять текущую группу.

Выдача сообщений клиенту:

- `greeting { full | brief | terse | text <строка> }` — определяет, какой текст будет выдаваться в строке приветствия:
  - `full` — имя хоста и версия сервера;
  - `brief` — имя хоста;
  - `terse` — ничего, кроме факта готовности к обслуживанию;
  - `text` — произвольная строка текста.
- `banner <имя_файла>` — определяет текст сообщения, выдаваемого клиенту до ввода имени/пароля;
- `hostname <имя_хоста>` — определяет имя хоста по умолчанию (имя локального хоста);
- `email <адрес>` — адрес администратора;
- `message <имя_файла> { LOGIN | CWD=<имя_каталога> { <класс> } }` — содержимое файла выдается клиенту при входе или смене каталога.
- `readme <имя_файла> { LOGIN | CWD=<имя_каталога> { <класс> } }` — при входе или смене каталога сервер информирует клиента о наличии указанного файла и дате создания/последней модификации.

Журнализация:

- `log commands` *<список_типов>* — выводить в журнал все команды клиента, где `список_типов` — список через запятую слов `real`, `guest` и `anonymous`;
- `log transfers` *<список_типов>* *<список_направлений>* — выводить в журнал пересылки файлов, где *<список_типов>* — список через запятую слов `real`, `guest` и `anonymous`; *<список_направлений>* — список через запятую слов `incoming` и `outbound`;
- `log security` *<список_типов>* — выводить в журнал нарушения правил безопасности, где *<список_типов>* — список через запятую слов `real`, `guest` и `anonymous`;
- `log syslog` — перенаправлять сообщения о пересылках в syslog вместо файла xferlog;
- `log syslog+xferlog` — направлять сообщения о пересылках в syslog и файл xferlog.

Виртуальные серверы:

- `daemonaddress` *<IP-адрес>* — использовать для соединения только указанный адрес;
- `virtual` *<IP-адрес>* `{ root | banner | logfile }` *<имя_файла>* — определить соответственно: корень файловой системы, файл, содержащий баннер приветствия, и журнал для указанного виртуального сервера;
- `virtual` *<IP-адрес>* `{ hostname | email }` *<строка>* — определить имя хоста (отображаемое в приветствии) и адрес администратора для указанного виртуального сервера;
- `virtual` *<IP-адрес>* `private` — закрыть анонимный доступ по указанному адресу;
- `virtual` *<IP-адрес>* `incmail` *<email-адрес>* — кого извещать в случае анонимной загрузки файлов;
- `virtual` *<IP-адрес>* `mailfrom` *<email-адрес>* — какой обратный адрес подставлять при рассылке сообщений о анонимной загрузке файлов;
- `defaultserver { deny | allow }` *<имя_пользователя>* `...` — по умолчанию доступ разрешен всем;
- `defaultserver private` — закрыть анонимный доступ;
- `defaultserver incmail` *<email-адрес>* — кого извещать в случае анонимной загрузки файлов;
- `defaultserver mailfrom` *<email-адрес>* — какой обратный адрес подставлять при рассылке сообщений об анонимной загрузке файлов.

Права доступа:

- { chmod | delete | overwrite | rename | umask } { yes | no } *<список_типов>* — разрешить/запретить пользователям выполнять соответствующее действие. По умолчанию — все разрешено. *<Список_типов>* — список через запятую слов anonymous, guest, real или class=*<имя_класса>*;
- passwd-check { none | trivial | rfc822 } ({ enforce | warn }) — уровень проверки правильности вводимых анонимными пользователями в качестве пароля адресов e-mail и реакция сервера в случае ошибки:
  - none — никакой проверки;
  - trivial — строка должна содержать @;
  - rfc822 — полная проверка согласно стандарту RFC-822;
  - warn — если обнаружена ошибка, то выдавать предупреждение;
  - enforce — если обнаружена ошибка, то не впускать пользователя;
- deny-email *<email-адрес>* — считать данный адрес неправильным;
- path-filter *<список-типов> <имя_файла_сообщения> <шаблон_допустимых_имен> <шаблон_недопустимых>* ... — когда пользователь типа из списка типов пытается загрузить файл на сервер, то сервер проверяет имя файла на соответствие регулярному выражению, указанному в шаблоне допустимых имен, и на несоответствие ни одному из регулярных выражений в шаблонах недопустимых имен;
- upload [absolute | relative] [class=*<имя-класса>*]... [–] *<корень шаблон_каталога>* { yes | no } owner group mode [dirs | nodirs] [dir_mode] — определяет каталоги, в которые разрешено/запрещено записывать файлы пользователям из указанного класса. Все создаваемые файлы будут иметь соответствующие права доступа и принадлежность;
- throughput — позволяет задать скорость передачи определенных файлов на определенные хосты;
- anonymous-root *<корень>* [*<класс>*] ... — определяет корневой каталог (chroot) для анонимных пользователей указанного класса и их домашний каталог;
- guest-root *<корень>* [*<интервал-uid>*] ... *<корень>* — определяет аргумент chroot для гостевых пользователей и их домашний каталог. Можно задавать отдельные uid или интервалы через дефис;
- deny-uid *<интервал>* ... — запрещает доступ к серверу определенным пользователям и может использоваться вместо файла ftpusers;
- deny-gid *<интервал>* ... — запрещает доступ к серверу определенным группам пользователей и может использоваться вместо файла ftpusers;
- allow-uid *<интервал>* ... — разрешает доступ к серверу определенным пользователям и может использоваться вместо файла ftpusers;

- allow-gid <интервал> ... — разрешает доступ к серверу определенным группам пользователей и может использоваться вместо файла ftpusers;
- restricted-uid <интервал> ... — разрешает реальному или гостевому пользователю доступ вовне его домашнего каталога;
- restricted-gid <интервал> ... — разрешает группе пользователей доступ вовне его домашнего каталога;
- unrestricted-uid <интервал> ... — запрещает реальному или гостевому пользователю доступ вовне его домашнего каталога;
- unrestricted-gid <интервал> ... — запрещает группе пользователей доступ вовне его домашнего каталога;
- site-exec-max-lines <число> [<класс>] ... — ограничивает число строк посылаемых командой SITE EXEC;
- dns refuse_mismatch <файл_с_сообщением> [override] — выдает сообщение, если прямой и обратный адреса клиента не совпадают. Если не указано override, то прекращать сеанс;
- dns refuse_no_reverse <файл_с_сообщением> [override] — выдает сообщение, если клиент не имеет обратного адреса. Если не указано override, то прекращать сеанс.

Разное:

- alias <строка> <имя_каталога> — позволяет переходить в указанный каталог по команде cd <строка> из любого каталога;
- cdpath <имя_каталога> — добавляет каталог к переменной cdpath, которая используется в качестве списка поиска для команды cd;
- compress { yes | no } <шаблон_классов> ... — разрешает/запрещает компрессию/декомпрессию для классов, подпадающих под шаблон;
- tar { yes | no } <шаблон_классов> ... — разрешает/запрещает использование tar для классов, подпадающих под шаблон;
- shutdown <имя_управляющего_файла> — файл содержит описание для остановки сервера;
- passive address <возвращаемый_IP-адрес> <CIDR_шаблон> — если клиент выдает команду PASS, то сервер определяет возвращаемый адрес, исходя из соответствия IP-адреса клиента CIDR-шаблону;
- pasive ports <CIDR_шаблон> min max — определяется интервал портов, из которых сервер выбирает порт для прослушивания случайным образом и передает его номер клиенту;
- pasv-allow <класс> <шаблон_адресов> — позволяет пользователям указанного класса соединяться не только с исходного адреса, но и с заданных шаблоном адресов;

- `port-allow` *<класс> <шаблон_адресов>* — позволяет пользователям данного класса указывать в команде `PORT` адрес, подходящий под шаблон;
- `lslong` *<команда>* [*<параметры>*] — какую команду и параметры использовать для генерации расширенного списка файлов в каталоге;
- `lsshort` *<команда>* [*<параметры>*] — какую команду и параметры использовать для генерации списка файлов в каталоге;
- `lsplain` *<команда>* [*<параметры>*] — какую команду и параметры использовать для генерации списка файлов в каталоге;
- `incmail` *<email-адрес>* — кого извещать в случае анонимной загрузки файлов;
- `mailserver` *<имя-хоста>* — какой почтовый сервер использовать для рассылки сообщений о анонимной загрузке файлов;
- `mailfrom` *<email-адрес>* — какой обратный адрес подставлять при рассылке сообщений о анонимной загрузке файлов.

Файл ftpservers

Этот файл определяет набор файлов конфигурации для каждого виртуального сервера. Каждая строка в данном конфигурационном файле описывает виртуальный сервер и состоит из двух полей:

- имя и IP-адрес виртуального сервера;
- имя каталога, содержащего конфигурационные файлы. Имена файлов фиксированы: ftpaccess, ftpusers, ftpgroups, ftphosts, ftpconversions. Если какой-либо конфигурационный файл отсутствует, то вместо него используется конфигурационный файл основного сервера.

Файл ftpconversions

В этом файле каждая строка описывает возможное преобразование файлов "на лету" и состоит из 8 полей, разделенных двоеточиями:

- удаляемый префикс;
- удаляемый суффикс;
- добавляемый префикс;
- добавляемый суффикс;
- используемая для преобразования внешняя программа и ее параметры;
- типы преобразуемого файла: `T_REG` — обычный файл, `T_ASCII` — текстовый, `T_DIR` — каталог или сочетание перечисленных типов;
- опции: `O_COMPRESS`, `O_UNCOMPRESS`, `O_TAR` или их сочетание;
- комментарий к строке преобразования.

Файл ftpgroups

Этот файл используется для поддержки функционирования нестандартных команд типа `SITE GROUP` и `SITE GPASS`. В файле ftpgroups находятся строки, состоящие из трех полей, разделенных двоеточием:

- задаваемое клиентом имя группы;
- зашифрованный пароль группы;
- реальное имя группы.

Файл ftphosts

Этот файл предназначен для ограничения доступа к FTP-серверу с определенных хостов. Используется всего две команды:

- `allow` *<имя_пользователя>* *<шаблон_IP-адреса>* `...` — разрешить доступ;
- `deny` *<имя_пользователя>* *<шаблон_IP-адреса>* `...` — запретить доступ.

Файл ftpusers

Этот файл предназначен для запрета доступа к FTP-серверу некоторым реальным пользователям. Используется обычно для повышения безопасности системы, чтобы исключить доступ пользователей типа `root`, `news` и т. п.

Параметры запуска программ, входящих в пакет

Помимо демона ftpd в пакет входит несколько программ, выполняющих различные действия. Они могут быть полезны для отладки FTP-сервера, получения статистической информации и управления сервером.

Программа ftpd

Эта программа — сервер ftp. При запуске можно использовать следующие ключи (приведены только основные):

- `-d` — выдавать отладочную информацию;
- `-l` — вести протокол по каждой сессии;
- `-t` *<число_секунд>* — время бездействия клиента, после которого сервер автоматически разрывает соединение (может быть изменен клиентом);
- `-T` *<число_секунд>* — время бездействия клиента, после которого сервер автоматически разрывает соединение;
- `-a` — использовать файл ftpaccess;
- `-A` — не использовать файл ftpaccess;
- `-i` — вести протокол о полученных файлах в файле xferlog;

- `-I` — запрещает использовать протокол IDENT;
- `-o` — записывать имена переданных файлов в xferlog;
- `-X` — делать записи о полученных и переданных файлах в файле syslog;
- `-u umask` — маска файла по умолчанию;
- `-w` — записывать заходы в wtmp;
- `-W` — не записывать заходы в wtmp;
- `-s` — самостоятельный запуск, не используя INETD;
- `-S` — самостоятельный запуск, не используя INETD, отсоединиться от терминала;
- `-p` *<порт>* — управляющий порт, по умолчанию берется FTP-порт из файла /etc/services, при использовании INETD не применяется;
- `-P` *<порт>* — порт данных, по умолчанию берется значение `ftp-data` из файла /etc/services;
- `-q` — использовать файлы для хранения номеров процессов;
- `-Q` — не применять файлы для хранения номеров процессов; при использовании этого параметра не будет работать ограничение на количество пользователей в классе;
- `-r rootdir` — сделать chroot (определение корневого каталога для программы) немедленно после запуска, не дожидаясь ввода имени пользователя; используется для построения защищенной системы.

Программа ftpwho

Эта утилита показывает информацию о каждом подключенном в данный момент клиенте.

Программа ftpcount

Утилита показывает текущее и максимальное количество пользователей для каждого класса пользователей.

Программа ftpshut

Утилита используется для безаварийного завершения работы FTP-сервера. Представляют интерес следующие ключи запуска:

- `-l` *<минуты>* — позволяет задать время, за сколько минут до завершения работы сервера запрещать установку новых соединений;
- `-d` *<минуты>* — задает время, за сколько минут до завершения работы сервера разрывать текущие соединения;

- *<время_завершения>* — время завершения работы сервера. Может быть задано в следующем виде:
  - `now` — немедленно завершить работу сервера;
  - `+минут` — через сколько минут завершить работу сервера;
  - `ЧЧММ` — время завершения работы сервера.

Программа ftprestart

Утилита производит запуск FTP-сервера, если он был завершен командой `stop`.

Программа ckconfig

Утилита проверяет конфигурацию FTP-сервера. Позволяет выявить случаи явных ошибок в конфигурационных файлах, однако бессильна в обнаружении логических ошибок.

Формат файла журнала xferlog

Как и положено, FTP-сервер ведет журнал событий. Файл журнала событий называется xferlog, и в нем протоколируется любой прием или передача файла. Информация о событии записывается строкой, состоящей из более чем десятка полей. Далее приведено описание полей записи.

1. Название дня недели, например `Sat`.
2. Название месяца.
3. День.
4. Часы:минуты:секунды.
5. Год.
6. Продолжительность передачи в секундах.
7. Имя удаленного хоста.
8. Размер файла в байтах.
9. Имя файла.
10. Тип передачи:
    - `a` — текстовый;
    - `b` — бинарный.
11. Действие над файлом в процессе передачи:
    - `C` — сжат;
    - `U` — разархивирован;

- `T` — обработан программой tar;
- `_` (символ подчеркивания) — не было произведено никаких действий.

12. Направление передачи:
    - `o` — с сервера;
    - `i` — на сервер.
13. Тип пользователя:
    - `a` — анонимный;
    - `g` — `guest` (гость);
    - `r` — `real` (зарегистрированный).
14. Имя реального пользователя либо идентификационная строка для анонимного или гостевого пользователя.
15. Имя сервиса.
16. Способ аутентификации:
    - `0` — отсутствует;
    - `1` — ident (RFC931).
17. Аутентифицированный идентификатор пользователя. Если аутентификация не использовалась — *.
18. Состояние передачи:
    - `c` — передача была закончена;
    - `i` — не закончена.

Безопасность

Во время конфигурации FTP-сервера очень желательно подумать о его безопасности. Зачастую неправильно сконфигурированный FTP-сервер становится тем слабым местом, через которое осуществляется прорыв безопасности вашей операционной системы.

Чрезвычайно важно, чтобы ваши анонимные и гостевые пользователи FTP не имели доступа к реальному командному процессору. Тогда, даже если они по каким-либо причинам смогут покинуть окружение FTP, то не смогут выполнить никаких посторонних задач. Для обеспечения этого требования убедитесь, что в файле /etc/passw у пользователей `guest` и `anonymous` в поле, где находится командная оболочка пользователя, находится что-то типа /dev/null.

Файл ftpusers должен содержать список нижеприведенных псевдопользователей, которым будет отказано в подключении к FTP-серверу.

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

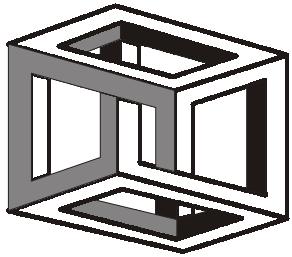
Обычно FTP-сервер разрешает загрузку файлов на сервер (upload) всем пользователям. Однако необходимо запретить пользователям загружать свои файлы в некоторые каталоги (а иногда и во все). Для этого в файле ftpaccess необходимо прописать опцию `upload` с ключом `no` и указать каталог, на который налагается запрет.

Иногда желательно запретить пользователям получение с FTP-сервера некоторых каталогов и файлов. Для этого в файле ftpaccess добавляем строку `noretrieve` с каталогом, куда необходимо запретить доступ пользователям.

Литература и ссылки

- RFC959 — RFC, описывающий FTP-протокол.
- **ftp.fni.com/pub/wu-ftpd/guest-howto** — HOWTO по настройке анонимного доступа на FTP-сервер.
- **ftp.wu-ftpd.org** — исходный текст пакета wu-ftp.
- **www.bog.pp.ru/work/ftpd.html** — описание конфигурирования сервера wu-ftp.
- **www.westnet.com/providers/multi-wu-ftpd.txt** — описание настройки виртуальных FTP-серверов.

ГЛАВА 9



NNTP. Сервер новостей INN

Одним из популярных сервисов, доступных в Интернете, является Usenet (новости, телеконференции, эхо-конференции в FIDO). Это похоже на смесь электронной почты и Web-форума — те же темы и сообщения, как в Web-форуме, только пользователь получает и отправляет сообщения, как электронную почту. В Usenet минимальной единицей информации является статья. Статья помещается в конференцию. Каждая конференция имеет свою тему. Конференций множество — несколько десятков тысяч. Конференции имеют иерархическую структуру. Имя образуется из имени родительской иерархии, к которому через точку добавляется имя конференции. К примеру — **fido7.ru.linux,** где **fido7** — корень иерархии, показывающий, что группа новостей импортирована из эхо-конференций FIDO, **ru** — русскоязычная, **linux** — конференция посвящена Linux. Для приема и передачи статей используются News-серверы (Usenet-серверы). Эти серверы производят синхронизацию (обмен статьями) между собой. Для передачи и приема статей используется протокол NNTP (Network News Transfer Protocol, сетевой протокол передачи новостей).

Протокол NNTP

Протокол NNTP описывается в документе RFC977, а стандарт обмена сообщениями в Usenet в документе RFC1036.

Протокол NNTP предназначен для рассылки, подписки, поиска и доставки новостей на основе TCP по технологии "клиент-сервер". Протокол NNTP использует стандартные сообщения, описанные в RFC850. Единицей хранения на сервере является статья.

Рассмотрим стандартный сценарий обмена информацией по протоколу NNTP. Пусть есть два или более хоста (один из них выступает в роли клиента, остальные — серверы). Процедура обмена начинается с запроса на получение списка новых групп новостей, для чего выдается команда

NEWGROUPS. Затем клиент делает запрос командой NEWNEWS о наличии новых статей из групп, представляющих интерес. Сервер высылает список статей, клиент обрабатывает список и запрашивает о получении статей, отсутствующих у клиента. И наконец, клиент может сообщить серверу, какие новые статьи он получил в последнее время.

Протокол NNTP использует протокол TCP и порт 119. На команды, посылаемые клиентом, предусмотрены текстовые и статусные отклики. Всякая сессия начинается с процедуры установления соединения между клиентом и сервером по инициативе клиента.

Данные могут посылаться только после цифрового статусного отклика. Данные представляют собой последовательности строк, каждая из которых завершается парой символов CR-LF. В конце текста всегда посылается строка, содержащая один символ (.), за которым следует CR-LF. Если исходный текст содержит точку в начале строки, то она перед посылкой дублируется. Статусный отклик представляет собой реакцию сервера на команду, полученную от клиента. Строки статусного отклика начинаются с 3-значного десятичного кода, в котором закодировано определенное состояние системы.

Трехзначный код является позиционным. Слева направо — первая цифра определяет состояние команды — успешно, в процессе выполнения, ошибка (табл. 9.1). Вторая цифра характеризует категорию команды (табл. 9.2). Третья цифра — уточняющее сообщение.

Некоторые коды являются предшественниками последующего текстового отклика. Первая цифра сообщает об успехе, ошибке или процессе исполнения команды.

Таблица 9.1. *Расшифровка первой цифры кода статусного отклика*

Код	Описание
1*xx*	Информационное сообщение
2*xx*	Команда ok
3*xx*	Команда корректна, можно продолжать обмен
4*xx*	Команда корректна, но не может быть выполнена по какой-то причине
5*xx*	Команда неприменима, неверна или произошла ошибка

Таблица 9.2. *Расшифровка второй цифры кода статусного отклика*

Код	Описание
*x*0*x*	Соединение, установка режима, прочие сообщения
*x*1*x*	Выбор группы новостей

Таблица 9.2 *(окончание)*

Код	Описание
*x*2*x*	Выбор статьи
*x*3*x*	Функции распределения
*x*4*x*	Отправка адресату
*x*8*x*	Нестандартное расширение
*x*9*x*	Отладочный вывод

Некоторые статусные отклики могут иметь параметры (числа или имена). Число и тип параметров фиксировано для каждого конкретного отклика. Параметры отделяются от кода отклика и друг от друга одиночным пробелом. Все цифровые параметры имеют десятичное представление и могут начинаться с нулей. Все строковые параметры начинаются после пробела и завершаются пробелом или символами CR-LF. Любой текст, который не является параметром отклика, должен отделяться от последнего параметра, если таковой имеется, пробелом и завершаться пробелом.

Коды категории *x*9*x* предназначены для отладочных целей. Так как большинство отладочных откликов можно рассматривать как информационные сообщения, для отладочных выдач зарезервирован диапазон кодов 190—199.

Далее приведен список кодов сообщений общего назначения, которые может послать NNTP-сервер (табл. 9.3). Эти отклики не привязаны к каким-то конкретным командам и могут быть присланы в результате сбоя или каких-то других необычных обстоятельств.

Таблица 9.3. *Коды сообщений общего назначения*

Код	Описание
100	Поясняющий текст
190—199	Отладочный вывод
200	Сервер готов, отправка разрешена
201	Сервер готов, отправка запрещена
400	Обслуживание прерывается
500	Команда не распознана
501	Синтаксическая ошибка в команде
502	Доступ ограничен или нет разрешения
503	Ошибка в программе, команда не выполнена

Основные команды протокола NNTP

В этом разделе мы рассмотрим основные команды протокола NNTP. Команды можно записывать в любом регистре, поскольку NNTP-сервер регистронезависим. Длина команды не должна превышать 512 байт.

ARTICLE

Существует две формы команды ARTICLE, каждая из которых использует различные методы спецификации извлекаемой статьи. Когда за командой ARTICLE следует идентификатор сообщения в угловых скобках ("<" и ">"), используется первая форма команды; если же в команде указывается цифровой параметр или нет параметра совсем, реализуется вторая форма.

ARTICLE <message-id>

Команда отображает заголовок, пустую строку и текст заданной статьи. Идентификатор сообщения (message-id) представляет собой идентификатор, содержащийся в заголовке статьи. Предполагается, что клиент получил идентификатор сообщения из списка, полученного командой NEWNEWS. Команда не изменяет указателя текущей статьи.

ARTICLE [nnn]

Отображает заголовок, пустую строку и текст текущей или указанной в цифровом параметре статьи. Опционный параметр *nnn* представляет собой числовой идентификатор статьи в текущей группе новостей. Он выбирается из диапазона, который был выдан при выборе группы. Если этого параметра нет, предполагается текущая статья. Эта команда устанавливает указатель текущей статьи, если номер статьи указан корректно.

В ответ сервер выдает номер текущей статьи, строка-идентификатор сообщения и текст статьи. Присылаемая строка идентификатора сообщения представляет собой заключенную в угловые скобки последовательность символов, которая извлечена из заголовка статьи.

BODY

Команда BODY идентична команде ARTICLE за исключением того, что она возвращает только основной текст статьи.

HEAD

Команда HEAD идентична команде ARTICLE за исключением того, что она возвращает только строки заголовка статьи.

STAT

Команда STAT похожа на команду ARTICLE за исключением того, что в ответ не присылается никакого текста. Команда STAT служит для установки указа-

теля статьи без пересылки какого-либо текста. Возвращаемый отклик содержит идентификатор сообщения.

GROUP ggg

Команда предназначена для выбора группы сообщений. Обязательный параметр *ggg* — имя группы новостей, которая должна быть выбрана. Отклик на успешный выбор группы возвращает номера первой и последней статьи в группе и оценку общего числа статей в группе.

После выбора группы внутренний указатель статьи устанавливается на первую в ней запись. Если выбрана несуществующая группа, остается в силе выбор предыдущей группы.

HELP

Эта команда дает краткое описание команд, которые может воспринять сервер. Отклик на команду имеет текстовую форму и завершается строкой с одиночной точкой в начале.

IHAVE <message-id>

Команда `IHAVE` информирует сервер о том, что клиент владеет статьей с идентификационным кодом *<message-id>*. Если сервер хочет скопировать статью, он пришлет отклик, предлагающий клиенту прислать ее.

Если запрошена передача статьи, клиент должен выслать полный текст статьи, включая заголовок. Сервер в этом случае пришлет отклик, уведомляющий об успехе или неудаче этой операции.

LAST

По команде `LAST` указатель на статью устанавливается на предшествующую запись в текущей группе. Если указатель уже установлен на первую статью, отправляется сообщение об ошибке, а указатель остается неизменным.

LIST

Присылает список доступных групп новостей. Каждой группе новостей соответствует строка в следующем формате:

<group> <last> <first> <p>

где *<group>* — название группы новостей, *<last>* — номер последней известной статьи в данной группе, *<first>* — номер первой статьи в группе, *<p>* — может быть либо 'y', либо 'n', указывая на наличие или отсутствие разрешения на рассылку.

Поля *<first>* и *<last>* являются числовыми. Если код поля *<last>* превосходит код поля *<first>*, в файле данной группы новостей нет ни одной статьи.

NEWGROUPS date time [GMT] [<distributions>]

Список групп новостей созданных, начиная с даты `<date>` и времени `<time>`, будет представлен в том же формате, что и в случае команды `LIST`.

Дата посылается в виде 6 цифр в формате ГГММДД, где ГГ — последние две цифры года, ММ — номер месяца (с нулем в начале, если необходимо), ДД — номер дня в заданном месяце. Дополнение для года берется из предположения о ближайшем тысячелетии, так 86 предполагает 1986, 30 — 2030, 99 — 1999, а 00 — 1900 годы.

Время должно характеризоваться 6 цифрами в формате ЧЧММСС, где ЧЧ — часы с начала суток в 24-часовом исчислении, ММ — минуты 00—59, а СС — секунды 00—59. Временная зона определяется сервером, в противном случае появляется символьная комбинация "GMT" (время указано по Гринвичу).

Заключенный в угловые скобки опциональный параметр `distributions` представляет собой список групп рассылки. Если параметр задан, рассылаемая часть новых групп новостей будет сравниваться с данным списком, и только при совпадении включается в список.

NEWNEWS newsgroups date time [GMT] [<distribution>]

Команда формирует список идентификаторов статей для заданной группы новостей, с датой после указанной. Для каждого идентификатора сообщения в списке выделяется одна строка. Список завершается строкой с одиночным символом точки, за которым следует CR-LF. Дата и время задаются в том же формате, что и для команды `NEWGROUPS`.

Для расширения зоны поиска в имени группы новостей можно использовать символ "*". Программа может подставить вместо звездочки любую комбинацию символов. Если вместо имени группы подставлен символ звездочка, поиск будет проведен по всем группам новостей.

Имя группы должно быть взято из списка доступных групп. Допускается задание нескольких групп (имена разделяются запятыми). После последнего имени группы не должно быть запятой.

NEXT

Команда устанавливает указатель текущей статьи на следующую запись в текущей группе новостей. Если в группе нет больше статей, посылается сообщение об ошибке, а указатель текущей статьи остается не измененным.

В качестве отклика на команду возвращается номер текущей статьи и идентификатор сообщения. Никаких текстовых сообщений не посылается.

POST

Команда позволяет отправить сообщение в текущую новостную группу.

QUIT

Сервер подтверждает получение команды `QUIT` и затем закрывает канал связи с клиентом. Эта команда предоставляет клиенту корректную возможность сообщить NNTP-серверу, что все операции завершены и сессия закончена.

SLAVE

Команда сообщает серверу, что он связывается не с пользователем, а с обслуживающим сервером (slave). Эта команда позволяет разделить случаи соединения сервера с отдельным пользователем и промежуточными обслуживающими серверами.

Сервер новостей INN

Пакет INN (InterNetNews) является одним из старейших пакетов программного обеспечения, предназначенного для создания сервера новостей. Использует стандартный протокол NNTP. Новости хранятся на сервере в дереве каталогов, имена которых формируются из имен телеконференций и повторяют их иерархическую структуру.

Работа пакета INN

Основной процесс — innd постоянно запущен в системе. Он ожидает и принимает поток статей по протоколу NNTP от серверов новостей, прослушивает порт 119 на наличие входящих соединений, ведет список активных групп, список статей, статьи, базу заголовков статей, пакеты статей для рассылки по серверам новостей, журналы.

При соединении клиентов для чтения новостей программа innd передает управление демону nnrpd. Этот демон просматривает файл nnrp.access для определения прав доступа к локальной базе статей.

Для управления работой innd — добавления, удаления групп, статей, серверов новостей, изменения параметров работы — используется программа ctlinnd.

Удалением старых статей с истекшим сроком хранения занимаются программы expire и expireover, которые удаляют устаревшие файлы, не останавливая innd.

Для автоматического обновления списка новостей используются управляющие сообщения.

Управляющие сообщения

Представляют собой обычные статьи в группе новостей, имеющие заголовок `Control:`. Встретив такую статью, innd обрабатывает записанную в ней команду и сохраняет статью.

Сообщения запоминаются в псевдогруппе `control`. Если создать подгруппу `control.`*имя_команды*, то все соответствующие статьи будут помещаться в эту подгруппу.

Настройка системы INN

Сервер новостей INN по возможностям и сложности настройки весьма напоминает пакет sendmail. Перечислим конфигурационные файлы, расположенные в каталоге /etc/news:

- /etc/news/actsync.cfg — используется для конфигурации автоматического изменения списка групп новостей. Обычно добавление новых групп новостей возлагается на администратора системы;
- /etc/news/actsync.ign — используется для конфигурации автоматического изменения списка групп новостей;
- /etc/news/control.ctl — в этом файле описывается как обрабатывать управляющие сообщения. Каждая его строка задает действие. Строки состоят из четырех полей, разделенных двоеточием. Первое поле задает команду, к которой применяется действие (можно указать ключевое слово `all`), последнее поле — действие. Строки просматриваются по порядку. Используется последняя подошедшая строка. Возможные действия:
  - `doit`
  - `doifarg`
  - `doit=`*отдельный_журнал*
  - `doit=mail`
  - `doit=` (без журнализации)
  - `drop`
  - `log` (запись в журнал — `errlog`)
  - `log=`*отдельный_журнал*
  - `mail`

  Для увеличения безопасности и устойчивости системы рекомендуется не использовать управляющие сообщения, а в файл control.ctl записать единственную строку `all:*:*:drop` — не делать никакой обработки вообще;
- /etc/news/cycbuff.conf — файл содержит конфигурацию метода хранения CNFS, обычно не используется;
- /etc/news/distrib.pats — файл используется программами посылки статей, в частности inews — для определения области распространения статьи. Область распространения определяется по шаблону группы новостей и приоритету. Обычно файл не используется;

- /etc/news/expire.ctl — файл определяет, через какое время статьи в базе устаревают. Использование файла зависит от метода хранения статей. В частности, метод хранения CNFS самостоятельно удаляет старые статьи. В этом же файле определяется, сколько времени хранить в "истории" информацию об удаленных или отвергнутых статьях.

  В начале файла обязательно должна находиться строка, определяющая срок хранения записи об идентификаторах статей в файле history после удаления тела статьи. Это позволяет отклонить статью, если поставщик новостей вновь предложит ее в определенный промежуток времени. Эта строка имеет следующий формат: `/remember/:`*`время`*, где *`время`* — срок хранения в днях, по истечении которого из системы удаляются идентификаторы старых статей.

  Здесь можно задать для различных групп или набора иерархий групп разные сроки хранения статей. Правила хранения статей задаются следующей строкой, состоящей из пяти полей, разделенных двоеточием:

  `<Шаблоны_имени_группы_через_запятую>``:`*`<флаг>`*`:min:default:max`

  - первое поле в строке задает группу или иерархию, удовлетворяющую шаблону;
  - второе поле содержит флаг, который определяет, к какому типу групп применять данное условие:
    - `A` — все группы;
    - `M` — только модерируемые;
    - `U` — только немодерируемые;
    - `X` — все группы. Если статья была послана в несколько групп и удовлетворяет данному шаблону, то она удалится не только из данной группы, но и из всех остальных групп, в которые была отослана;
  - третье поле задает минимальное число дней хранения. Также можно использовать ключевое слово `never`;
  - четвертое поле определяет число дней хранения по умолчанию. Также можно использовать ключевое слово `never`;
  - пятое поле определяет максимальное число дней хранения статьи в базе. Также можно использовать ключевое слово `never`;

- /etc/news/incoming.conf — в файле определяется, кто может служить для нашего сервера поставщиком новостей. Определяющая строка имеет вид: имя, двоеточие, пробел, значение. В качестве имени используются следующие слова:
  - `hostname`. В качестве значения — список полных доменных имен хостов или десятичных IP-адресов через запятую;

- `streaming`. В качестве значения — `true` или `false`; параметр определяет, разрешен ли потоковый режим;
- `max-connections` — параметр определяет максимальное число параллельных соединений;
- `password` — если сервер новостей требует авторизации, здесь прописывается пароль, обычно не используется;
- `patterns`. В качестве значения — шаблон групп, принимаемых с указанного хоста;
- `noresendid`. В качестве значения — `true` или `false`; определяет, должен ли сервер новостей посылать ответ `431 RESENDID` в потоковом режиме и `436 Retry later` в непотоковом режиме в ответ на попытку послать статью, которая уже была принята;

❒ /etc/news/inn.conf — файл содержит глобальные параметры сервера новостей и параметры, используемые при формировании заголовков статей, создаваемых на этом сервере. Все изменения, сделанные в этом файле, считываются демоном innd только после перезагрузки сервера новостей. Строки в конфигурационном файле имеют следующий формат:

<имя>: *<значение>*

Далее описываются имена параметров и их значения:

- `fromhost` — параметр используется при формировании заголовка `From:`, если его нет. Переменная окружения `FROMHOST` переопределяет это значение. По умолчанию, это полное доменное имя локальной машины;
- `moderatormailer` — имя хоста, содержащего псевдонимы для всех модерируемых групп. Рекомендуется использовать файл moderators;
- `organization` — определяет содержимое заголовка `Organization:`, если таковой отсутствует. Если определена переменная окружения `ORGANIZATION`, то она переопределяет это значение;
- `pathhost` — определяет, какое имя локального узла помещается в заголовок `Path:`. По умолчанию, это полное доменное имя локальной машины;
- `server` — определяет имя NNTP-сервера, на котором должны публиковаться созданные статьи. В том случае, если определена переменная окружения `NNTPSERVER`, то она изменяет это значение;
- `domain` — определяет имя домена, к которому принадлежит локальная машина;
- `overviewmmap` — определяет, будут ли программы expire, nnrpd и makehistory использовать mmap для доступа к файлу overview;

- `storageapi` — определяет способ хранения статей: `false` для традиционного метода хранения статей; `true` — для хэшированных имен, `cnfs` — для кольцевых буферов:
  - традиционный метод — каждая статья в отдельном файле; каждая группа — в каталоге с соответствующим именем;
  - хэшированные имена — каждая статья хранится в отдельном файле, но имена выбираются исходя из ускорения доступа к файлам;
  - CNFS — все статьи хранятся в кольцевых буферах; есть возможность группировки статей по определенным критериям;
- `maxforks` — определяет максимально возможное количество одновременно запущенных демонов innd;
- `maxartsize` — определяет максимально возможный размер статьи;
- `nicekids` — определяет приоритет процессам, порождаемым программой nnrpd;
- `nicenewnews` — определяет еще более низший приоритет программе nnrpd, обрабатывающей команду `NEWNEWS`;
- `mta` — определяет программу, используемую для посылки почтой модерируемых статей;
- `mailcmd` — определяет программу для посылки отчетов;
- `logcancelcomm` — определяет, сбрасывать ли в стандартную систему журнализации событий (syslog) сообщения о выполнении команды `cancel`;
- `wanttrash` — определяет, сохранять ли статьи для несуществующей группы в группе junk;
- `remembertrash` — определяет, запоминать ли отвергнутые статьи в файле history;
- `linecountfuzz` — определяет, исправлять ли заголовок `Lines`;
- `logartsize` — указывает серверу запоминать в журнале размер статьи;
- `logipaddr` — определяет, записывать ли в журнал событий IP-адрес вместо значения из заголовка `Path`;
- `logsitename` — определяет, сохранять ли имя хоста в журнале полученных статей;
- `overviewname` — задает имя файла для хранения истории сообщений; для каждой группы — свой; по умолчанию имя файла — .overview;
- `extendeddbz` — ускоряет работу с overview за счет увеличения DBZ-файла; требует определенного параметра `storageapi`;

- `nnrpdoverstats` — позволяет сохранять в стандартную систему журнализации событий syslog статистку истории сообщений для nnrpd;
- `storeonxref` — при использовании нестандартного метода хранения использовать `Xref:` вместо `Newsgroup:`;
- `nnrpdcheckart` — благодаря этому значению nnrpd будет не только читать overview, но и проверять реальное наличие статьи;
- `storemsgid` — разрешает хранить идентификатор сообщения (Message-ID);
- `usecontrolchan` — позволяет использовать канал для обработки управляющих статей;
- `refusecybercancel` — указывает серверу отвергать статьи, идентификатор сообщения (Message-ID) которых начинается с `cancel`;
- `activedenable`, `activedupdate`, `activedport` — указывает использовать вспомогательный процесс для буферизации доступа nnrpd к файлу active;
- `pathnews`, `pathbin`, `pathfilter`, `pathcontrol`, `pathdb`, `pathetc`, `pathrun`, `pathlog`, `pathhttp`, `pathtmp`, `pathspool`, `patharticles`, `pathoverview`, `pathoutgoing`, `pathincoming`, `patharchive`, `pathuniover` — указывают серверу пути к различным составляющим сервера новостей: исполняемым файлам, базам сообщений, журналам событий и т. п;
- `backoff` — задает ограничение на количество статей, посылаемых локальными клиентами с помощью nnrpd;
- `strippostcc` — указывает nnrpd удалять поля `To:`, `Cc:` и `Bcc:`;
- `nnrpperlauth` — указывает серверу аутентифицировать читателя nnrpd с помощью внешней программы на языке программирования скриптов Perl;
- `pathalias` — указывает, какую строку добавлять перед `pathhost`;
- `nnrpdposthost`, `nnrpdpostport` — программы nnrpd и rnews будут отправлять статьи на этот сервер;
- `wireformat` — указывает серверу хранить статьи в том же формате, что и при передаче CR LF в конце каждой строки и удвоении точки в начале строки;
- `status` — позволяет производить регулярную выдачу статистики на стандартную систему журнализации событий syslog;
- `timer` — позволяет производить регулярный вывод информации о загруженности сервера на стандартную систему журнализации событий syslog;

- `peertimeout` — определяет, сколько секунд входной канал может оставаться неактивным, прежде чем innd его закроет;
- `chaninacttime`, `chanretrytime` — определяют, сколько секунд канал может быть неактивным, прежде чем innd его закроет;
- `maxconnections` — задает число одновременных NNTP-соединений;
- `artcutoff` — задает количество дней для хранения статей (статьи, старше указанного числа дней, удаляются);
- `nntplinklog` — разрешает записывать в журнал сообщения nntplink;
- `nntpactsync` — задает, сколько статей обрабатывать между записями в журнал;
- `badiocount` — определяет, сколько ошибок ввода/вывода допускать, не закрывая канал;
- `pauseretrytime` — задает паузу между проверками канала на неактивность;
- `sourceaddress` — определяет, какой адрес будут иметь исходящие пакеты; если указано `any` — параметр будет выбран операционной системой;
- `port` — задает порт, который будет прослушиваться;
- `localmaxartsize` — определяет максимальный размер посылаемых через nnrpd статей;
- `mimeversion` — разрешает nnrpd добавлять MIME-заголовки;
- `mimecontenttype` — если добавляются MIME-заголовки, то здесь определяется значение заголовка `Content-Type`;
- `mimeencoding` — если добавляются MIME-заголовки, то здесь определяется значение заголовка `Content-Transfer-Encoding`;
- `spoolfirst` — если задано `true`, то nnrpd помещает статью от клиента в спул, даже не пытаясь обратиться к innd; если `false` — помещает ее в спул только при получении сообщения об ошибке при посылке;
- `articlemmap` — разрешает использовать mmap при доступе к статье в спуле;
- `clienttimeout` — определяет, сколько секунд клиент nnrpd может не проявлять активность до разрыва соединения;
- `innflags` — задает флаги, передаваемые innd при запуске;
- `doinnwatch` — определяет, запускать ли программу innwatch;
- `innwatchsleeptime` — задает промежуток между проверками innwatch в секундах;

- `controlfailnotice` — определяет, посылать ли администратору письма об ошибках обработки управляющих сообщений;
- `logcycles` — задает, сколько копий старых журналов нужно сохранять;
- `innwatchpauseload` — содержит среднюю загрузку, умноженную на 100, при которой innwatch будет переводить innd в режим ожидания;
- `innwatchhiload` — определяет среднюю загрузку, умноженную на 100, при которой innwatch будет переводить innd в режим throttle;
- `innwatchloload` — средняя загрузка, умноженная на 100, при которой innwatch будет возвращать innd в нормальный режим;
- `innwatchspoolspace` — задает размер свободного места на устройстве, хранящем articles и overview, в единицах inndf, при достижении которого innwatch переводит innd в режим throttle;
- `innwatchbatchspace` — задает размер свободного места на устройстве, хранящем исходящие сообщения, в единицах inndf, при достижении которого innwatch переводит innd в режим throttle;
- `innwatchlibspace` — задает размер свободного места на устройстве, хранящем файлы db-history, active, в единицах inndf, при достижении которого innwatch переводит innd в режим throttle;
- `docnfsstat` — определяет, запускать ли cnfsstat (данный параметр нужен только при использовании метода хранения статей CNFS);

❒ /etc/news/innfeed.conf — конфигурационный файл для программы innfeed. Более подробную информацию следует искать в документации к серверу новостей;

❒ /etc/news/innreport.conf — конфигурационный файл для программы innreport. Более подробную информацию следует искать в документации к серверу новостей;

❒ /etc/news/innwatch.ctl — конфигурационный файл для программы innwatch. Каждая строка определяет одну проверку, состоит из семи полей, разделенных одним символом, и начинается с того же символа. Разделитель полей един для всей строки и выбирается из списка: восклицательный знак, запятая, двоеточие, `@`, точка с запятой или вопросительный знак; в зависимости от того, какой знак из перечисленных ранее не встречается внутри полей в этой строке;

❒ /etc/news/moderators — файл, который хранит имя модерируемой группы и электронный адрес модератора. Когда nnrpd или inews получает статью от клиента и выясняется, что она послана в модерируемую группу, то вместо того, чтобы послать ее innd, он посылает ее по электронной почте модератору этой группы. В данном файле задаются шаблоны для определения адреса модератора по имени группы. Каждая строка состоит из

двух полей, разделенных двоеточием. В первом поле указывается шаблон имени группы, а во втором — электронный адрес модератора конференции;

- /etc/news/news2mail.cf — конфигурационный файл для программы news2mail;
- /etc/news/newsfeeds — файл содержит информацию о том, какие статьи и каким образом необходимо пересылать на соседние NNTP-узлы. Для каждого узла, с которым вы обмениваетесь новостями, должно быть соответствующее описание в этом файле.

  Каждая строка представляет собой отдельное правило, состоящее из 4-х полей, разделенных двоеточиями:

  - *<имя_сайта>*/*<список_исключений_через_запятую>* — первым сайтом в файле должен быть сайт с именем ME. Если он имеет список шаблонов групп, то этот список добавляется в начало списков остальных сайтов:
    - *<Имя_сайта>* получателя записывается в журнал; если имя сайта уже встречается в `Path:`, то статья на него не посылается; для локальных имен (программ обработки типа overchan, archive и т. д) рекомендуется добавлять восклицательный знак в конце, чтобы не пересечься с реальным именем сайта; в качестве имени сайта получателя обычно выбирается то имя, которое этот сайт вставляет в `Path:` при обработке статьи;
    - *<Список_исключений>* — список имен сайтов через запятую; для каждого имени делается аналогичная проверка — не встречается ли он в `Path:`. Часто используются имена генераторов управляющих сообщений: cyberspam, spewcancel, bincancel;
  - *<список_шаблонов_имен_групп_через_запятую>*/*<список_областей_распределения_через_запятую>*
    - *<Список_шаблонов>* определяет, какие группы будут посылаться на сайт получателя. Восклицательный знак в начале шаблона означает отрицание. Наибольший приоритет имеет последнее соответствие. Если вместо `!` использовать `@`, то статья из соответствующей группы не будет посылаться на данный сайт, даже если она отсылается в группу, подлежащую посылке;
    - область распространения дополнительно ограничивает список рассылаемых статей — если статья имеет заголовок `Distribution:` и определен список областей распространения для данного сайта получателя, то они должны соответствовать друг другу. Правила записи аналогичны правилам записи шаблонов. Если статья имеет несколько областей распространения, то используется логическое "ИЛИ".

- *<список_флагов>*
  - `<size` — статья посылается, если ее размер меньше указанного числа байтов;
  - `>size` — статья посылается, если ее размер больше указанного числа байтов;
  - `Ac` — не посылать управляющие сообщения;
  - `AC` — посылать только управляющие сообщения;
  - `Ad` — только статьи с заголовком `Distribution:`;
  - `Ae` — если заголовок статьи `Newsgroups:` содержит только те группы, которые имеются в списке активных групп;
  - `Ap` — не проверять наличие имени сайта получателя в `Path:` до отсылки сообщения;
  - `F`*<имя_файла>* — задает имя файла для спула;
  - `G`*<число>* — посылать статью, если она послана не более чем в указанное число групп;
  - `H`*<число>* — посылать статью, только если в `Path:` накопилось не более указанного числа хостов;
  - `I`*<размер>* — величина внутреннего буфера, после которого данные начинают сбрасываться в файл;
  - `Nm` — только модерируемые группы;
  - `Nu` — только немодерируемые группы;
  - `P`*<приоритет>* — число от 0 до 20, которое будет назначено программе или каналу;
  - `O`*<шаблон>* — требуется наличие заголовка `X-Trace`, и первое поле в нем должно соответствовать шаблону;
  - `S`*<размер>* — если в очереди к данному сайту находится больше указанного размера байтов, то innd переходит в режим спулинга — сбрасывает во временный файл;
  - `T`*<тип>* — способ передачи статей на сайт: `c` — канал, `f` — файл, `l` — только запись в журнал (очень удобно собирать статистику), `p` — программа;
  - `W`*<поле>* — если передача происходит через файл или канал, то здесь указывается, какую информацию туда записывать. Можно использовать несколько флагов. Поля будут записаны в указанном порядке и будут разделяться пробелами. Программы понимают только поле `*` (`b` — размер статьи в байтах, `f` — полное имя файла статьи, `g` — имя первой группы, `h` — hash-ключ Message-ID, `m` —

Message-ID, `n` — имя файла статьи относительно спула, `p` — время посылки статьи, `s` — откуда пришла статья, `t` — время получения статьи, `*` — имена всех сайтов, получающих данную статью, `D` — значение заголовка `Distribution:` ("`?`" если не было), `H` — все заголовки, `N` — заголовок `Newsgroups:`, `P` — заголовок `Path:`, `R` — данные для репликации);

- *<параметры>* — формат зависит от способа посылки статей на сайт. Перечислим способы посылки статей:
  - журнал — делается только запись в журнале /var/log/news/news;
  - файл — для каждой статьи в файл, определяемый полем *<параметры>*, записывается одна строка. По умолчанию, имя файла — outgoing/*имя_сайта*;
  - программа — для каждой статьи запускается новый экземпляр программы;
  - канал — в поле *<параметры>* задается полное имя программы, которая запускается при старте innd. На каждую статью запущенный процесс получает одну строку на стандартный ввод. Стандартный вывод, ошибки, UID и GID — как для случая программы. Если процесс уже запущен, он перезапускается. Если процесс не удается запустить, то образуется спул в outgoing/*имя_сайта*;
  - exploder — особый подтип канала, кроме обычных статей на него могут быть посланы команды. Команда предваряется восклицательным знаком. Автоматически генерируются следующие команды: `newgroup` *<имя_группы>*, `rmgroup` *<имя_группы>*, `flush`, `flush` *<имя_сайта>*;
  - `funnel` — слияние нескольких потоков в один. Поле *<параметры>* определяет реального получателя;

❒ /etc/news/nnrp.access — файл определяет права доступа к данному NNTP-узлу. Все строки состоят из пяти полей, разделенных двоеточием, и имеют следующий формат:

<шаблон_хостов>:*<права_доступа>*:*<имя_пользователя>*:*<пароль>*:↳
<шаблон_имен_групп>

- *<шаблон_хостов>* — задает шаблон для сравнения с хостом клиента и может использовать как имена, так и адреса с сетевой маской;
- *<права_доступа>* — перечень букв, которые определяют права клиента, зашедшего с соответствующего адреса:
  - `R` — клиент имеет право на чтение;
  - `P` — клиент имеет право на посылку;
  - `N` — клиент может использовать команду `NEWNEWS`, несмотря на глобальный запрет;

  - L — клиент может посылать статьи в группы с запретом на локальную посылку;
  - *<полное_имя_файла>* — формат файла такой же, как и основного, права доступа уточняются, исходя из него;
- *<имя_пользователя>* — пустое, если аутентификация клиента не нужна;
- *<пароль>* — пустой, если аутентификация клиента не нужна;
- *<шаблон_имен_групп>* — список шаблонов имен групп через запятую, к которым клиент должен иметь доступ;

❒ /etc/news/nnrpd.track — файл позволяет nnrpd записывать в журнал доступа определенную строку текста вместо имени или адреса хоста клиента. Состоит из строк вида:

<шаблон_имен_или_адресов_хостов>:*<строка_идентифицирующая_пользователя>*

❒ /etc/news/nntpsend.ctl — файл определяет список хостов, на которые nntpsend будет рассылать статьи, если имя хоста не указано явно при запуске. Каждая строка определяет отдельный хост и имеет вид:

<сайт>:fqdn:*<размер><параметры>*

- *<сайт>* — имя, указанное в newsfeeds;
- fqdn — полное доменное имя хоста, на который должны быть посланы статьи;
- *<размер>* — размер для обрезания пакета заданий, если он станет слишком большим;
- *<параметры>* — параметры для innxmit;

❒ /etc/news/overview.ctl — файл применяется для создания файла истории сообщений overview при использовании новых способов хранения статей;

❒ /etc/news/overview.fmt — файл определяет, какие заголовки будут храниться в файле истории сообщений overview;

❒ /etc/news/passwd.nntp — в этом файле хранятся пароли для доступа к NNTP-серверам;

❒ /etc/news/storage.conf — файл определяет параметры для нестандартных методов хранения статей. Для каждого класса определяется своя структура хранения.

Файл active

Этот файл содержит список групп новостей, которые принимает локальный сервер. Все статьи, опубликованные в группы новостей, которые не указаны в файле active, игнорируются локальным сервером новостей. Строки в этом файле имеют следующий формат:

<имя> <старшая_метка> <младшая_метка> <флаги>

где:

- *<имя>* — имя группы новостей;
- *<старшая_метка>* — номер самой новой статьи в данной группе новостей на локальном сервере. Это число увеличивается при получении новых статей;
- *<младшая_метка>* — номер самой старой статьи в данной группе новостей на локальном сервере. Это число изменяется в результате удаления старых статей на диске;
- *<флаги>* — это поле определяет один из шести возможных флагов:
  - `y` — для данной группы новостей разрешена локальная публикация;
  - `n` — для данной группы новостей не разрешена локальная публикация;
  - `m` — данная группа модерируемая, и все публикации должны быть одобрены модератором;
  - `j` — статьи из данной группы новостей не хранятся на локальном сервере, а только передаются через него;
  - `x` — статьи не могут посылаться в данную группу новостей;
  - `=news.group` — статьи для данной группы новостей помещаются локально в группу news.group.

Основные операции, которые должен время от времени выполнять администратор, включают в себя добавление новых групп, удаление ненужных групп, изменение флагов текущих групп новостей. Все эти операции должны находить свое отображение в файле active. Существует два основных подхода к выполнению указанных ранее операций с группами новостей.

- *Первый подход* — использование соответствующих подкоманд команды `ctlinnd` — `newgroup`, `rmgroup` и `changegroup`;
- *Второй подход* — непосредственное редактирование файла active. Такой подход удобен для операций с большим количеством групп.

Файлы базы данных и журналы

Список файлов базы данных и их стандартное размещение приведено далее.

- /var/lib/news/.news.daily;
- /var/lib/news/active;
- /var/lib/news/active.times;
- /var/lib/news/distributions;
- /var/lib/news/history;
- /var/lib/news/newsgroups;
- /var/lib/news/subscriptions.

Список файлов журналов и их стандартное размещение приведено далее.

- /var/log/news;
- /var/log/news/OLD;
- /var/log/news/news.crit;
- /var/log/news/news.err;
- /var/log/news/news.notice.

Сами статьи находятся в следующих файлах:

- /var/spool/news/archive;
- /var/spool/news/articles;
- /var/spool/news/incoming;
- /var/spool/news/incoming/bad;
- /var/spool/news/innfeed;
- /var/spool/news/outgoing;
- /var/spool/news/overview;
- /var/spool/news/uniover.

Настройка списка получаемых групп новостей

Для настройки получаемых групп новостей необходимо получить список новостей, которые провайдер предоставляет клиентам. Приведем один из способов получения списка:

```
getlist -h newsserver.our.pro > active.provider
```

Созданный этой командой файл active.provider содержит список групп новостей, на которые предоставляет провайдер. Выберем из списка те группы, на которые мы действительно хотим подписаться, и пропишем их в нашем файле active. Например, если вы хотите подписаться на конференцию relcom.humor, добавьте в этот файл примерно следующее:

```
relcom.humor 0000000000 0000000001 y
```

Если вы хотите принимать все (или почти все) группы новостей, которые предоставляет провайдер, то файл active можно получить из active.provider, выполнив для него следующие команды (обнуляются два средних поля каждой строки):

```
#!/bin/sh
sed < active.provider > active \
-e 's/^\([^ ]*\) [0-9]* [0-9]* \([^ ]*\)$/\1 0000000000 0000000000 \2/'
```

Нужный файл active готов (он содержит строки для всех групп, которые поддерживает наш сервер), но надо сообщить и провайдеру о нашем выборе (чтобы он знал, какие группы новостей ему нужно пересылать на наш хост).

Даже если провайдер пропишет нас в своей конфигурации сервера новостей, он не сможет пересылать нам новости по NNTP. Мы должны дать ему разрешение на это. Для чего добавим строчку в файл hosts.nntp:

```
newsserver.our.provider:
```

Здесь надо заметить, что мы полагаемся на провайдера — знаем, что он будет снабжать нас только теми конференциями, о которых мы его попросили. Если же вы не доверяете своим NNTP-соседям, то можно указать конкретно шаблон конференций, которые вы принимаете на локальный диск от конкретного NNTP-соседа. Например, мы хотим принимать от newsserver.our.badprovider только relcom-группы новостей:

```
newsserver.our.badprovider::relcom.*
```

Отредактируем файл newsfeeds, указав всех NNTP-соседей, которых мы хотим снабжать статьями. Не забудем указать в этом файле своего провайдера. Далее приведены два примера этого файла.

- В первом случае мы планируем снабжать статьями хост newsserver.our.provider по NNTP:

  ```
  ME:*, !junk, !control*, !local*/!local::
  newsserver.our.provider:*, !junk, !control*, !local*:Tf,
  Wnm:newsserver.our.provider
  ```

- Во втором случае мы хотим снабжать этот же хост по UUCP (имя этой UUCP-системы `provider`), используя программу sendbatch:

  ```
  ME:*, !junk, !control*, !local*/!local::
  provider/newsserver.our.provider:*, !junk, !control*, !local*:Tf, Wnb:
  ```

Затем назначим различные глобальные параметры сервера новостей (имя сервера, имя домена) и параметры, используемые при формировании заголовков статей, публикуемых у нас. Эта информация хранится в файле inn.conf.

Определимся теперь с клиентами нашего сервера новостей (хосты, которые через программу чтения новостей общаются с нашим сервером). Например, мы хотим ограничить пространство пользования ресурсами нашего сервера новостей своей интранет-сетью (192.168.1.0/255.255.255.0) и нашей внешней сетью (домен our.domain), причем пользователям этих сетей мы разрешаем и читать новости, и публиковать их на нашем сервере. При этом надо помнить о партнерах из домена partner.domain (правда, им нечего делать в наших локальных конференциях). Ну а для остальных поместим первым правило, запрещающее любой доступ. Для этого добавим в файл nnrp.access строки:

```
*:: -no- : -no- :!*
192.168.1.*:Read Post:::*
*.our.domain:Read Post:::*
*.partner.domain:Read Post:::*, !local*
```

Как только мы начнем получать статьи на локальный диск, надо будет следить за сроком их хранения на диске и удалять старые. К счастью, за нас

это будет делать программа expire, а от нас требуется только дать ей соответствующие указания в файле expire.ctl (ну и конечно, запускать механизм очистки). В этом файле следует указать:

- срок хранения идентификаторов статей в файле history (это делается для того, чтобы не принимать заново удаленные статьи);
- срок хранения самих тел статей.

Приведенный далее пример показывает, что запись об идентификаторе статей хранится в файле history 14 дней после удаления тела этих статей, тела статей из локальных телеконференций хранятся в системе от 5 до 7 дней (по умолчанию 6), а для всех остальных телеконференций тела хранятся от 3 до 5 дней (по умолчанию 4 дня).

```
/remember/:14
*:A:3:4:5
local*:A:5:6:7
```

Заметим, что значение по умолчанию (образец `*`) должно фигурировать раньше, чем строки для отдельных групп, поскольку применяется последнее соответствие образцу в первом поле.

Важным шагом после редактирования конфигурационных файлов является проверка корректности сделанных нами изменений. Система INN имеет ряд средств, помогающих нам в решении этой задачи. Вот некоторые из них:

- Для поиска ошибок в файле newsfeeds можно дать такую команду:

  ```
  innd -s
  ```

  Например, если вы получили в ответ следующее:

  ```
  Found 1 errors --see syslog
  ```

  то это значит, что командой обнаружена одна ошибка, о которой сообщается через syslog в файлах news.err и news.notice.

- Для проверки файла active на наличие неверных строк можно дать нижеприведенную команду:

  ```
  expire -n -x -t
  ```

  Например, если в ответ получено следующее:

  ```
  /var/news/etc/active: line 5 wrong number of fields
  ```

  то это значит, что вы ошиблись с количеством полей в 5-й строке данного файла (их должно быть 4). Однако это не лучший способ проверки файла active. В частности, expire не замечает отсутствие флага для группы новостей (в отличие от inncheck).

Итак, обратим внимание на inncheck — Perl-сценарий, предназначенный для проверки всех рассматриваемых нами конфигурационных файлов. Помимо проверки файлов на наличие синтаксических ошибок, он может осу-

ществлять проверку прав доступа к файлам и их владельцев. Возвращаясь к примеру, рассмотренному ранее (отсутствие флага в конце строки файла active), inncheck сообщит вам об этой ошибке:

```
/var/news/etc/active:5: ends with whitespace
```

Запущенный без параметров, inncheck проверит синтаксис всех файлов (которые может проверить), с выводом на экран сообщений об ошибках. Если мы укажем опцию `-v` (режим verbose), то inncheck расскажет нам о том, что он просматривает. Мы можем ограничить работу inncheck проверкой синтаксиса конкретного файла, дав команду `inncheck <имя_файла>`. Для того чтобы проверить корректность прав доступа к файлам и корректность владельцев и групп файлов, можно дать команду `inncheck -perm`.. Аналогичную информацию, да еще и с указанием того, какие команды надо выполнить, чтобы устранить ошибки, дает команда `inncheck -f -perm`..

Последний шаг настройки — периодически запускать программу отправки статей с нашей машины, программу чистки каталога статей и обобщения LOG-файлов. Для этого отредактируем таблицу заданий пользователя news для демона cron:

```
crontab -u news -e
```

Ваш редактор (определенный переменной окружения `EDITOR`) откроет файл /var/cron/tabs/news. Ежедневно в 4 часа утра мы будем запускать сценарий news.daily, в функции которого входит обобщение и ротация файлов регистрации, прогон программы expire и др. Далее, в 1-ю и 28-ю минуту каждого часа мы будем запускать программу nntpsend для отправки потоков статей по NNTP нашим соседям.

```
0    4  *  *  *   /usr/news/bin/news.daily > /dev/null 2>&1 &
1, 28 *  *  *  *  /usr/news/bin/nntpsend > /dev/null 2>&1 &
```

Наконец, если мы планируем отправлять потоки новостей по UUCP на UUCP-систему provider, то в 37-ю минуту каждого часа из cron будем вызывать программу sendbatch:

```
37  *  *  *  *  /usr/news/bin/sendbatch -c provider > /dev/null 2>&1 &
```

Журналирование пакета INN

Пакет INN применяет стандартный способ — систему журнализации событий syslog. Помимо этого, можно использовать дополнительные журналы сообщений, в частности:

- news.crit — содержит сообщения о критических ошибках, требующих внимания от администратора сервера новостей;

- news.err — содержит сообщения о фатальных ошибках сервера;
- news.notice — используется для записи информации о соединении удаленных NNTP-хостов, активности клиентов, в этом же файле информируют о своей работе программы ctlinnd, innxmit, rnews.

Система INN имеет помимо LOG-файлов, поддерживаемых системой syslog, встроенные LOG-файлы — errlog и news (по умолчанию они расположены в каталоге /var/log/news):

- файл errlog содержит стандартный вывод и стандартные ошибки любых программ, порождаемых демоном innd;
- файл news регистрирует все статьи, поступающие к innd для обработки.

Помимо перечисленных ранее файлов регистрации, несколько программ системы INN ведут собственные файлы регистрации (expire.log, send-uucp.log, nntpsend.log и др.).

Программы пакета INN

Поскольку пакет INN очень велик, то в этом разделе приведены некоторые программы, имеющие отношение к пакету с небольшими комментариями.

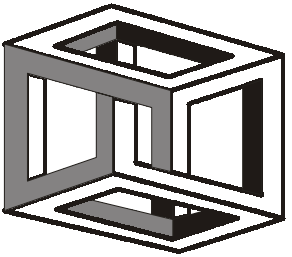
- /usr/bin/actived — вспомогательный демон для nnrpd, хранит в памяти проиндексированный файл active;
- /usr/bin/actmerge — утилита, позволяющая произвести слияние двух файлов active;
- /usr/bin/actsync — утилита для синхронизации, сравнения или слияния файлов active;
- /usr/bin/archive — утилита для создания архивной копии части статей;
- /usr/bin/batcher — программа разбивает на пакеты указанного размера список статей, подготовленных для посылки на хост;
- /usr/bin/controlchan — программа позволяет передать обработку управляющих сообщений из innd внешней программы;
- /usr/bin/convdate — утилита для преобразования формата времени;
- /usr/bin/ctlinnd — интерфейс для управления работающим innd;
- /usr/bin/cvtbatch — преобразует Usenet-пакеты в формат INN;
- /usr/bin/expire — утилита для удаления старых статей, не прерывая работы innd;
- /usr/bin/expireindex — удаление старых статей из списка заголовков статей группы;
- /usr/bin/expireover — удаление старых статей из списка статей группы;
- /usr/bin/fastrm — быстрое удаление группы файлов;

- /usr/bin/getlist — получение списков от NNTP-сервера;
- /usr/bin/grephistory — быстрое извлечение статьи по ее индексу;
- /usr/bin/inncheck — проверка конфигурационных файлов;
- /usr/bin/innd — основной сервер, принимающий данные и изменяющий базу данных;
- /usr/bin/inndstart — пусковая программа для innd;
- /usr/bin/innreport — обработка журналов;
- /usr/bin/innstat — выдать состояние сервера;
- /usr/bin/innwatch — мониторинг сервера inn;
- /usr/bin/innxbatch — послать статьи в формате Usenet другому NNTP-серверу;
- /usr/bin/innxmit — пересылка пакета статей другому NNTP-серверу;
- /usr/bin/mailpost — поместить письмо в группу news;
- /usr/bin/makeactive — восстановление файла active по спулу;
- /usr/bin/news.daily — подготовка ежедневного отчета;
- /usr/bin/news2mail — превращение статей в письма;
- /usr/bin/nnrpd — отдельный процесс, предоставляющий клиентам доступ к статьям;
- /usr/bin/nntpsend — оболочка для innxmit;
- /usr/bin/overchan — заполнение данных списка заголовков статей группы;
- /usr/bin/parsecontrol — анализ управляющих сообщений;
- /usr/bin/pgpverify — проверка управляющих сообщений;
- /usr/bin/scanlogs — обработка журналов;
- /usr/bin/send-nntp — подготовка и рассылка пакетов с помощью innxmit;
- /usr/bin/sendxbatches — подготовка и рассылка пакетов с помощью innxbatch;
- /usr/bin/writelog — запись в журнал inn.

Литература и ссылки

- RFC977 — Network News Transfer Protocol — Протокол обмена сетевыми новостями.
- RFC1036 — Standard for interchange of USENET — Стандарт обмена сообщениями в USENET.
- **antonio.mccinet.ru/net/nntp.html** — Протокол новостей (NNTP).

- **ief.tup.km.ua/docs/Linux/NAG/nag19.html** — Описание NNTP.
- **malik.bishkek.su/doc/UNIX/innd/inn.htm** — Савин Ю. Сервер новостей InterNetNews (INN).
- **www.bog.pp.ru/work/inn.html** — конфигурирование сервера INN.
- **www.isc.org/products/INN** — официальный сайт INN.
- **www.logic.ru/Russian/soft/ligs/node382.html** — Система электронных новостей и USENET.
- **www.mibsoftware.com/userkt/inn/0346.htm** — утилиты для пакета INN.
- **www.switch.ch/switch/netnews/wg/newstools.html** — утилиты для пакета INN.



ГЛАВА 10

Web-сервер Apache

В качестве HTTP-сервера в UNIX-сообществе в основном используется Web-сервер Apache, который распространяется по лицензии GNU. По статистическим данным более половины всех Web-серверов в Сети созданы на базе сервера Apache.

Чем же привлекателен этот сервер? Во-первых, большое количество возможностей — использование CGI-скриптов, шифрования, доступ по паролю, перекодирование страниц "на лету", поддержка виртуальных хостов и многое другое. Во-вторых, малая требовательность к ресурсам и большая производительность. В-третьих, многоплатформенность — Apache есть для Linux, для различных клонов UNIX, для Windows. В-четвертых, он бесплатный и с открытым исходным кодом. Список можно продолжать. Конечно, есть и недостатки, к примеру, некоторые сложности с конфигурированием. Но в целом — этот сервер не зря получил столь большую популярность.

Второе место по количеству установок занимает Web-сервер Microsoft IIS, который теперь входит в стандартную поставку Windows 2000 сервер и Windows 2003 сервер.

Microsoft IIS является мощным сервером, который по основным параметрам находится на уровне сервера Apache. Однако у Microsoft IIS также есть недостатки — и основной из них — платформозависимость. Он функционирует только под управлением операционной системы семейства Windows NT/2000/XP/2003. Также Microsoft IIS не предназначен для обработки большого количества одновременных запросов (порядка десятков тысяч запросов в секунду).

В качестве альтернативы для Linux-платформы можно использовать Web-сервер TUX, который тесно интегрирован с ядром Linux, что позволило резко увеличить количество обрабатываемых запросов за единицу времени.

Однако у этого сервера есть несколько минусов, в том числе:

- платформозависимость;
- неустоявшийся код;
- мало дополнительных возможностей по сравнению с Apache.

Конфигурация

Установка сервера для дистрибутивов, использующих RPM-пакеты, стандартна — необходимо скачать нужный пакет и произвести установку сервера командой

```
rpm -I <имя_пакета>
```

Конфигурирование сервера достаточно сложно — несколько сотен команд и параметров, некоторые крайне редко используются. Поэтому далее рассматриваются наиболее распространенные директивы и их параметры.

Замечание

В том случае, если производится переконфигурирование на рабочем сервере Apache, вам необходимо дать указание серверу перечитать конфигурационные файлы. Сервер перечитывает конфигурационные файлы при запуске либо при получении сигнала `-HUP` или `-USR1`. Если сервер Apache находится в работе, то при изменении конфигурации его рекомендуется перезапустить командой `kill -USR1`, поскольку при таком перезапуске сервера текущие соединения завершаются обычным образом, а следующие клиенты работают уже с новыми конфигурационными файлами.

Конфигурация сервера задается в файлах httpd.conf, srm.conf, access.conf и .htaccess. Файл httpd.conf предназначен для общей конфигурации сервера, srm.conf содержит описание доступных ресурсов, а access.conf — права доступа к ресурсам. Однако в современных версиях сервера любая директива конфигурации может лежать в любом из этих файлов. Сейчас де-факто все директивы конфигурации содержатся в файле httpd.conf.

Некоторые модули могут иметь свои отдельные файлы конфигурации (например, mod_charset требует файлы, хранящие таблицы перекодировки).

Используемые обозначения

Далее показаны обозначения, используемые при описании параметров конфигурации сервера:

- `S` — директива действует на поведение сервера целиком;
- `V` — действует, если запрос касается данного виртуального хоста;
- `D` — определяет свойства только данного каталога;
- `A` — определяет свойства для всех каталогов.

Права доступа и свойства объекта

Права доступа к данному каталогу и его свойства определяются следующими директивами:

- `DA allow from {host}`

  Определяет, с каких хостов разрешен доступ к данному каталогу:

  - `all` — для всех;
  - *<доменное имя>* — с тех хостов, имя которых заканчивается этой строкой;
  - *<полный IP-адрес>*;
  - *<частичный IP-адрес>* — 1, 2 или 3 байта IP-адреса;
  - `a.b.c.d/e.f.g.h` — сеть/сетевая маска;
  - `a.b.c.d/nnn` — сеть/подсеть.

- `DA allow from env=`*<имя_переменной>*

  Доступ разрешается, только если определена соответствующая переменная окружения.

- `D AllowOverride {None | All | AuthConfig | FileInfo | Indexes | Limit | Options}`

  Определяет, какие директивы из .htaccess в данном каталоге могут перекрывать конфигурацию сервера.

- `D AuthName` *<домен_авторизации>*

  Определяет, какой домен авторизации клиент должен использовать при определении имени и пароля.

- `DA deny from {host}`

  Определяет, с каких адресов запрещен доступ к данному каталогу:

  - `all` — для всех;
  - *<доменное имя>* — с тех хостов, имя которых заканчивается этой строкой;
  - *<полный IP-адрес>*;
  - *<частичный IP-адрес>* — 1, 2 или 3 байта IP-адреса;
  - `a.b.c.d/e.f.g.h` — сеть/сетевая маска;
  - `a.b.c.d/nnn` — сеть/подсеть.

- `DA deny from env=`*<имя_переменной>*

  Доступ не разрешается, если определена соответствующая переменная окружения.

- SV <Directory *имя_каталога*> ... </Directory>

  Внутри этой пары тегов определяются права и свойства данного каталога. В качестве имен используется полный путь к каталогу.

- SV <DirectoryMatch *регулярное_выражение*> ... </DirectoryMatch>

  Внутри пары тегов определяются права и свойства данного каталога. В качестве имени используется регулярное выражение.

- SV DocumentRoot <*путь*>

  Определяет, где находится корневой каталог документов сервера или виртуального сервера.

- SVDA ErrorDocument <*код_ошибки*> <*документ*>

  Определяет, какой документ выдавать в случае ошибки с указанным кодом.

- SVA <Files *имя_файла*> ... </Files>

  Внутри пары тегов определяются права и свойства файлов. Может находиться внутри секции Directory или .htaccess.

- SVA <FilesMatch *имя_файла*> ... </FilesMatch>

  Внутри пары тегов определяются права и свойства файлов, в качестве имен используется регулярное выражение. Может находиться внутри секции Directory или .htaccess.

- SVDA <Limit {*метод*}> ... </Limit>

  Эта пара тегов для группы директив, управляющих доступом. Метод — GET, POST, PUT, DELETE, CONNECT или OPTIONS.

- SV <Location URL> ... </Location>

  Пара тегов для определения свойств и прав доступа для данного URL.

- SV <LocationMatch URL> ... </LocationMatch>

  Пара тегов для определения свойств и прав доступа для данного URL (регулярное выражение).

- SVDA Options [+|-]option ...

  Определяет возможности сервера в данном каталоге:

  - ALL — все кроме MultiView;
  - ExecCGI — разрешается выполнение CGI;
  - FollowSymLinks — разрешено ходить по символьным ссылкам;
  - Includes — использовать SSI (Server Side Include);
  - IncludesNOEXEC — использовать SSI, кроме exec и include CGI;
  - Indexes — генерировать список содержимого каталога, если отсутствует файл index.html;

- `MultiViews` — определять представление ресурса в зависимости от предпочтений клиента;
- `SymLinksIfOwnerMatch` — следовать по символьным ссылкам, только если владелец целевого файла совпадает с владельцем ссылки.

❒ `DA order option`

Определяет очередность, в которой применяются директивы `allow` и `deny`:

- `deny, allow` — первой применяется директива `deny`, затем `allow` (начальное состояние — доступ разрешен);
- `allow, deny` — первой применяется директива `allow`, затем `deny` (начальное состояние — запрещено);
- `mutual-failure` — доступ только с тех хостов, которые перечислены в `allow` и не перечислены в `deny`.

❒ `DA require entity-name entity entity...`

Какой пользователь может иметь доступ к каталогу;

- `user {userid}` — только пользователи с данными именами;
- `group {group-name}` — только пользователи из данной группы;
- `valid-user` — любой аутентифицированный пользователь.

❒ `DA satisfy [all|any]`

Если для ограничения доступа используется логин/пароль и IP-адрес, то сервер будет требовать соответствия обоих критериев (`all`) или любого из них (`any`). По умолчанию — `all`.

Общие характеристики сервера

Общие характеристики сервера определяются следующими директивами:

❒ `SV ErrorLog filename | syslog:facility`

Определяет, куда выводить сообщения об ошибках.

❒ `SV Group <`*группа*`>`

Определяет, с правами какой группы будет обрабатываться запрос.

❒ `SVD HostNameLookups on | off | double`

Указывает, определять ли имя клиента по его IP-адресу.

❒ `SVDA <IfDefine [!]parameter-name> ... </IfDefine>`

Условная конфигурация, если параметр (не) определен.

❒ `SVDA <IfModule [!]module-name> ... </IfModulee>`

Условная конфигурация, если модуль (не) включен в состав сервера.

- `S Include <имя_файла>`

  Вставляет содержимое файла в состав конфигурационного файла в данном месте.

- `S KeepAlive on | off`

  Обслуживает несколько запросов, не прерывая TCP-соединения с клиентом.

- `SV LogLevel emerg | alert | crit | error | warn | notice | info | debug`

  Определяет, что писать в журнал ошибок.

- `S MaxClients <число>`

  Определяет максимальное количество одновременно обслуживаемых клиентов.

- `S MaxKeepAliveRequests <число>`

  Определяет, максимальное количество запросов.

- `S MaxRequestsPerChild <число>`

  Определяет максимальное количество одновременно обслуживаемых запросов одним процессом.

- `S MaxSpareServers <число>`

  Определяет максимальное число процессов, не осуществляющих в данный момент соединения.

- `S MinSpareServers <число>`

  Определяет минимальное число процессов, не осуществляющих в данный момент соединения.

- `S Port <номер_порта>`

  Определяет, по какому порту производится соединение (по умолчанию — 80 порт).

- `SV RLimitCPU <Мягкое ограничение (нижняя граница)> <жесткое ограничение (верхняя граница)>`

  Задает максимальное число секунд CPU для любого процесса. Оба параметра могут быть числом или словом `max`.

- `SV RlimitMEM <Мягкое ограничение (нижняя граница)> <жесткое ограничение (верхняя граница)>`

  Задает максимальное число байтов, которое может использовать каждый процесс. Оба параметра могут быть числом либо словом `max`.

- `SV RlimitNPROC <Мягкое ограничение (нижняя граница)> <жесткое ограничение (верхняя граница)>`

  Задает максимальное число процессов, которое может запустить каждый пользователь. Оба параметра могут быть числом либо словом `max`.

- `SV ServerAdmin <email-адрес>`

  Электронный адрес администратора Web-сервера.

- `SV ServerName <имя>`

  Полное доменное имя, используется для перенаправления.

- `S ServerRoot <полное-имя-каталога>`

  Указывает место, где лежат все файлы сервера по умолчанию.

- `SVDA ServerSignature Off | On | Email`

  Определяет, какую информацию включать в конце документов, генерируемых сервером:

  - `Off` — отсутствие информации;
  - `On` — имя сервера и версия;
  - `EMail` — имя сервера, версия и почтовый адрес администратора.

- `S ServerTokens Minimal|OS|Full`

  Определяет, что сервер сообщает о себе в заголовке `Server`.

- `S ServerType standalone | initd`

  Определяет тип сервера — постоянно находящийся в оперативной памяти или вызываемый демоном initd.

- `S StartServers <число>`

  Определяет, сколько дочерних процессов запускать при начальном старте сервера.

- `S TimeOut <секунд>`

  Количество секунд, определяющее тайм-аут.

- `SVDA UseCanonicalName on|off`

  Используется при генерации URL, ссылающихся на этот же сервер:

  - `On` — использовать имя, определенное в `ServerName` и `Port`;
  - `Off` — использовать параметры из запроса пользователя.

- `SV User uid`

  Определяет, с правами какого пользователя будет работать сервер.

Виртуальные серверы

Виртуальные серверы — в последнее время — повсеместно используются для Web-сайтов с относительно небольшой нагрузкой (десятки запросов в секунду). Идея очень проста — на мощном сервере Apache конфигурируется таким образом, что на одном IP-адресе находятся несколько Web-серверов

с разными символическими именами и разным физическим пространством для организации структуры Web-страниц.

Общие характеристики виртуальных серверов определяются следующими директивами:

- `S NameVirtualHost` *адрес*`[:`*порт*`]`

  Задает пару соответствия виртуальный хост — адрес/порт.

- `V ServerAlias host1 host2 ...`

  Задает альтернативные имена для виртуального хоста.

- `V ServerPath <`*путь*`>`

  Все запросы, которые начинаются с `<`*путь*`>` будут обслуживаться этим виртуальным сервером.

- `S <VirtualHost {`*адрес*`[:`*порт*`]}> ... </VirtualHost>`

  Пара тегов определяет описание виртуального сервера. Адрес и порт определяют адрес, по которому он будет отзываться. Внутри используются любые директивы с признаком `V`.

Преобразование адресов

Преобразование адресов определяется следующими директивами:

- `SV Alias <`*URL*`> <`*каталог-имя_файла*`>`

  Запрос, начинающийся с `<`*URL*`>`, будет отображен на файл, начинающийся с `<`*каталог-имя_файла*`>`.

- `SV AliasMatch <`*регулярное_выражение*`> <`*каталог-имя_файла*`>`

  Аналогично директиве `Alias`, но сравнение производится в соответствии с регулярным выражением.

- `SV ScriptAlias <`*url-path*`> <`*каталог-имя_файла*`>`

  Аналогично директиве `Alias`, но дополнительно пометить каталог как содержащий CGI.

- `SV ScriptAliasMatch <`*регулярное_выражение*`> <`*каталог-имя_файла*`>`

  Аналогично директиве `AliasMatch`, но дополнительно пометить каталог как содержащий CGI.

Преобразование HTTP-заголовков

Преобразование HTTP-заголовков определяется следующими директивами:

- `SVDA MetaFiles on/off`

  Включает/выключает преобразование для данного каталога.

- `SVDA MetaDir <каталог>`

  Определяет имя каталога, в котором лежат метафайлы.

- `SVDA MetaSuffix <суффикс>`

  Определяет суффикс, который добавляется к имени файла, чтобы найти метафайл для него.

- `SVDA ExpiresActive on|off`

  Определяет, посылать ли заголовок `Expire` (срок хранения документа в кэше).

- `SVDA Header unset header`

  Предписывает удалить заголовок.

Безопасность

Безопасность сервера определяется следующими директивами:

- `DA AuthGroupFile <имя_файла>`

  Определяет имя файла, в котором хранится список групп пользователей.

- `DA AuthUserFile <имя_файла>`

  Определяет имя файла, в котором хранится список пользователей.

- `D AuthType [Basic | Digest]`

  Определяет тип аутентификации.

- `DA AuthAuthoritative on | off`

  Если установлено `off`, то в процессе авторизации, если отсутствует имя пользователя в текущей базе данных, происходит обращение к модулю аутентификации нижнего уровня.

- `DA AuthDBMGroupFile <имя_файла>`

  Аналогично `AuthGroupFile`, но использует `dbm`.

- `DA AuthDBMUserFile <имя_файла>`

  Аналогично `AuthUserFile`, но использует `dbm`.

Индекс каталога

Индекс каталога определяется следующими директивами:

- `SVDA AddAlt <string_file> <file>...`

  Определяет, какой текст показывать вместо пиктограммы, если на стороне клиента отключена загрузка картинок.

- `SVDA AddDescription <string_file> <file>...`

  Определяет текстовое описание файла.

- `SVDA AddIcon <icon_name> <name>...`

  Определяет, какую картинку показать для файла, соответствующего *<name>*.

- `SVDA DefaultIcon <url>`

  Определяет, какая картинка будет использоваться, если нет соответствующей.

- `SVDA DirectoryIndex <local-url> <local-url>...`

  Задает имя файла (относительно запрашиваемого каталога), в котором находится индексный файл каталога.

- `SVDA HeaderName <имя_файла>`

  Определяет, что в качестве заголовка индекса будет вставлен указанный файл.

- `SVDA IndexIgnore <имя_файла> <имя_файла>...`

  Определяет список файлов, которые надо скрывать.

- `SVDA IndexOptions [+|-]option [+|-]option ...`

  Определяет параметры сортировки и оформления:

  - `FancyIndexing` — сортировка по столбцам;
  - `IconHeight=<pixels>` — высота пиктограммы;
  - `IconWidth=<pixels>` — ширина пиктограммы;
  - `NameWidth=[n | *]` — ширина колонки.

- `SVDA ReadmeName <имя_файла>`

  В конец индекса будет вставлен указанный файл (сначала ищется файл *<имя_файла>*.html, затем просто *<имя_файла>*).

Перекодировка (русификация)

Для перекодирования документов из одной кодовой страницы в другую используются приведенные далее директивы.

Определение кодировки и таблиц перекодировки:

- `SV CharsetDecl <имя_кодировки> [ S ]`

  Флаг `S` подавляет выдачу `charset=...` клиенту.

- `SV CharsetRecodeTable <из_какой> <в_какую> <имя_файла_с_таблицей> [ <имя_файла_с_обратной_таблицей> ]`

  Задает, из какой кодировки в какую производится перекодирование.

- `SV CharsetWideRecodeTable <из_какой> <в_какую> <имя_файла_с_таблицей>`

  Используется для перекодировок из символа в строку, например, для транслитерации.

- `SVDLA CharsetAlias <официальное_имя> <синоним>...`

  Определяет синонимы для имени кодировки.

Определение кодировки хранения:

- `SVDLA CharsetSourceEnc <имя_кодировки>`

  Определяет, в какой кодировке хранятся документы.

- `SVDLA CharsetByExtension <имя_кодировки> .ext1 ...`

  Разрешает определение кодировки по расширению.

- `SVDLA CharsetProcessType <mime-type>`

  Определяет, какие типы файлов надо обрабатывать; всегда обрабатываются — text/*.

Определение кодировки клиента:

- `SVDLA CharsetPriority <имя_кодировки1>...`

  Определение приоритета, если клиент задает несколько `Accept`.

- `SVDLA CharsetBrokenAccept Agent-Substring <accept_charset_string>`

  Игнорировать данный заголовок `Accept` от данного клиента — использовать другие механизмы для определения типа клиентской кодировки.

- `SVDLA CharsetSelectionOrder <Rule1>...`

  Приоритет способов определения кодировки клиента:

  - `Portnumber` — по номеру порта;
  - `Hostname` — если каноническое имя хоста начинается с имени кодировки или его синонима, то выбирается данная кодировка;
  - `URIHostname` — если имя в заголовке `Host:` начинается с имени кодировки или его синонима, то выбирается данная кодировка;
  - `EnvVariable` — по переменной `FORCE_CHARSET`, определенной внешними модулями;
  - `Dirprefix` — по началу имени каталога;
  - `Useragent` — по HTTP-заголовку `User-Agent`.

- `SVDLA CharsetDefault <имя_кодировки>`

  Принимается в качестве кодировки клиента, если все остальные способы не помогли.

- `SVDLA CharsetByPort <имя_кодировки> <номер_порта>`

  Определяет кодировку по номеру порта, к которому произошло подключение.

Дополнительная обработка специфических случаев:

- `SVDLA AddHandler <strip-meta-http> .ext1 ...`

  Удалять теги "`META HTTP-EQUIV=... charset=...`" из HTML-файлов перед передачей их клиенту.

- `SVDLA CharsetBadAgent <шаблон>...`

  Для клиентских программ, подпадающих под шаблон, не будет выдаваться строка `charset=` в HTTP-заголовке `Content-type`.

- `SVDLA CharsetErrReject On | Off`

  Если клиент запрашивает неизвестную кодировку в `Accept/Accept-charset` выдавать сообщение об ошибке или попытаться определить правильную кодировку.

- `SVDLA CharsetDisable On | Off`

  Выключить модуль для данного сервера/каталога.

- `SVDLA CharsetRecodeFilenames On | Off`

  Перекодировать имена файлов.

- `SVDLA CharsetOverrideExpires On | Off`

  Если включен — заменять заголовки `Expires`, сгенерированные другими модулями, на свои.

- `SVDLA CharsetDisableForcedExpires On | Off`

  Если выключен — сервер выдает заголовок `Expires: 1 Jan 1970` для того, чтобы документ не кэшировался, если его кодировка определилась по `User-Agent` или `Accept-charset`.

- `SVDLA CharsetRecodeMethodsIn <метод1>...`

  Включить обработку запроса для данного метода: `GET`, `POST`, `PUT`, `ALL`, `NONE`.

- `SVDLA CharsetRecodeMethodsOut <метод1>...`

  Включить обработку ответа для данного метода: `GET`, `POST`, `PUT`, `ALL`, `NONE`.

Это далеко не все параметры, используемые при конфигурации сервера Apache. Для более полного описания конфигурационных директив смотрите документацию, идущую в комплекте с сервером Apache.

Файл access.conf

В access.conf содержатся директивы, описывающие права доступа к каталогам и файлам Web-сервера. Обычно создается каталог /www/<*имя_сервера*>/, потому что при такой организации проще ориентироваться в структуре файлов.

Файл access.conf содержит секции `Directory`, `Location` и `Files`, которые ограничены одноименными директивами. В параметрах этих директив могут использоваться символы "?" и "*", а также регулярные выражения, предваряемые тильдой "~". В секции `Directory` помещаются инструкции, относящиеся к определенному каталогу на диске, в секции `Location` — относящиеся к виртуальному пути, в секции `Files` — относящиеся к файлу или группе файлов.

```
<Directory /www/room.ru>
# директивы, относящиеся ко всем документам, хранящимся
в каталоге /www/room.ru и вложенных в него
</Directory>

<Location /cgi-bin>
# директивы, относящиеся ко всем документам, доступным по
адресу http://<имя_сервера>/cgi-bin/ <путь_к_файлу>
</Location>

<Files /www/room.ru/form.htm>
# директивы, относящиеся к файлу form.htm из каталога
 /www/room.ru
</Files>
```

Различие между секциями `Directory` и `Location` состоит в том, что первая относится к каталогам на диске, вторая — к виртуальному пути (URL), который браузер запрашивает у Web-сервера. И в той, и в другой могут присутствовать директивы `order`, `allow` и `deny`, которые позволяют ограничить доступ к каталогу или URL с различных машин.

При отсутствии специальных требований к безопасности можно указать `Options All` в секции `<Directory /www>`, иначе нужно описать параметры каждого каталога отдельно.

Приведем пример файла access.conf:

```
## access.conf — Apache HTTP server configuration file

<Directory />
Options FollowSymLinks
AllowOverride None
</Directory>
```

```
<Directory /www>
Options All
AllowOverride All
order allow,deny
allow from all
</Directory>
```

Файл srm.conf

Файл srm.conf содержит директивы, связанные с общими настройками структуры каталогов сервера. Обычно они не изменяются.

Файл httpd.conf

Конфигурационный файл httpd.conf является основным и содержит настройки, связанные с работой Web-сервера, виртуальных серверов, а также всех его программных модулей. Кроме того, именно в нем настраивается перекодирование русских букв при передаче от сервера к клиенту и обратно.

Директива `Port`, помещенная в самом начале файла, определяет номер порта для HTTP-сервера; по умолчанию это 80. При необходимости можно приписать серверу другой порт или несколько портов.

Директива `HostnameLookups` с параметром `on` или `off` включает или отключает преобразование численных IP-адресов клиентов, получивших данные с сервера, в доменные имена.

Директивы `User` и `Group` задают пользователя, который будет администрировать сервер. С точки зрения безопасности нежелательно указывать здесь существующего пользователя, имеющего доступ к каким-либо другим ресурсам или файлам. Лучше создать отдельного пользователя и группу специально для HTTP-сервера.

Директивы `ServerRoot`, `ErrorLog`, `CustomLog` определяют корневой каталог HTTP-сервера, путь к журналу регистрации ошибок (error_log) и путь к общему журналу обращений к серверу (access_log).

Настройка виртуальных серверов в файле httpd.conf

Обычно на одном физическом Web-сервере размещаются несколько так называемых "виртуальных" Web-серверов. Это делается по нескольким причинам — экономия денег, экономия IP-адресов, проще с администрированием.

Однако оборотной стороной медали является больший вред при взломе сервера, поскольку теперь при взломе сервера злоумышленник получает доступ к десяткам или даже сотням Web-сайтов.

Виртуальные серверы могут иметь один и тот же IP-адрес и разные доменные имена, а могут иметь и разные IP-адреса. Для описания адресов и доменных имен виртуальных серверов служат директивы `ServerName`, `ServerAlias`, `NameVirtualHost` и `VirtualHost`. Они необходимы, только если вам нужно установить более одного виртуального сервера.

Директива `ServerName`, находящаяся вне секций `VirtualHost`, определяет имя основного сервера, корневой каталог которого задан директивой `DocumentRoot` в файле srm.conf. Виртуальные серверы наследуют настройки основного; при необходимости специальной настройки соответствующие директивы помещаются в секции `VirtualHost`, относящейся к данному серверу.

Приведем фрагмент конфигурационного файла для виртуальных серверов с различными IP-адресами:

```
...
ServerName www.root.ru

<VirtualHost 192.168.0.2>
DocumentRoot /www/root.ru
ServerName www.root.ru
ErrorLog /var/log/error_log.root.ru
CustomLog /var/log/access_log.root.ru combined
...
</VirtualHost>

<VirtualHost 192.168.11.6>
DocumentRoot /www/r.ru
ServerName www.r.ru
ErrorLog /var/log/error_log.r.ru
CustomLog /var/log/access_log.r.ru combined
...
</VirtualHost>
```

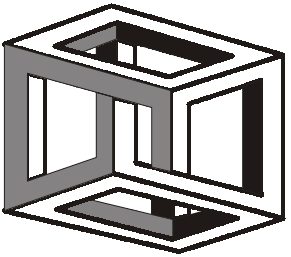
Представим фрагмент конфигурационного файла для виртуальных серверов с одинаковыми IP-адресами:

```
...
ServerName www.root.ru
NameVirtualHost 192.168.0.2

<VirtualHost 192.168.0.2>
DocumentRoot /www/root.ru
ServerName www.root.ru
ErrorLog /var/log/error_log.root.ru
```

```
CustomLog /var/log/access_log.root.ru combined
...
</VirtualHost>

<VirtualHost 190.168.0.2>
DocumentRoot /www/r.ru
ServerName www.r.ru
ServerAlias *.r.ru
ErrorLog /var/log/error_log.r.ru
CustomLog /var/log/access_log.r.ru combined
...
</VirtualHost>
```

Литература и ссылки

- Apache-Overview-HOWTO — обзор Web-сервера Apache.
- Building a Secure RedHat Apache Server HOWTO — настройка безопасного Apache.
- _fastcgi mini-HOWTO — инсталляция Apache Web-сервера с поддержкой модулей mod_perl, mod_ssl и php (с поддержкой модулей Apache+DSO+mod_ssl+mod_perl+php+mod_auth_nds+mod_auth_mysql+mod).
- Linux Apache SSL PHP/FI frontpage mini-HOWTO — настройка Web-сервера.
- **http://apache.lexa.ru** — сервер группы разработчиков русского модуля Apache.
- **http://bog.pp.ru/work/apache.html** — Apache: HTTP-сервер. Установка, настройка и русификация.
- **http://www.apache.org** — официальный сервер Apache.
- **http://www.cs.ifmo.ru/education/documentation/rapacheman/index.shtml** — Подстрешный А. Работа с Web-сервером Russian Apache.



ГЛАВА 11

Proxy-сервер

Что такое Proxy-сервер? Это специальный сервер, который все данные, получаемые с Web-серверов и проходящие через него, сохраняет у себя в кэше на определенное время (обычно сутки). При вторичном обращении к Web-серверу Proxy-сервер вместо того, чтобы заново получать с Web-сервера данные, выдает клиенту информацию, ранее сохраненную в кэше. При этом сервер предварительно проверяет, не истек ли срок хранения информации в кэше, и не изменилась ли информация на Web-сервере

Чем больше пользователей используют Proxy-сервер, тем более существенной становится его помощь.

Многие Proxy-серверы обладают еще одним интересным свойством — они могут обмениваться информацией с другими Proxy-серверами, что существенно ускоряет доступ к данным, хранящимся на удаленных или сильно загруженных серверах.

Proxy-сервер предоставляет следующие возможности:

- централизованный выход в Интернет через один сервер в сети;
- локальное хранение часто просматриваемых документов для увеличения скорости загрузки страниц;
- возможность регулировать пропускную способность канала в зависимости от его нагрузки;
- авторизованный доступ в Интернет;
- возможность обмена данными кэша с соседними Proxy-серверами;
- замена или удаление нежелательной информации (например, баннеров).

Однако не все данные могут быть корректно получены через Proxy-серверы. Это касается, прежде всего, динамически формируемой информации. Но многие современные Proxy-серверы имеют большое количество настроек и

обладают массой интеллектуальных алгоритмов, позволяющих в большинстве случаев корректно получать самую свежую информацию.

Наиболее распространенным Proxy-сервером, доступным под лицензией GNU, является Squid.

Squid

Squid — это высокопроизводительный кэширующий Proxy-сервер, поддерживающий протоколы FTP, gopher и HTTP.

Поддерживаемые функции Squid:

- Proxy и кэширование HTTP, FTP;
- Proxy для SSL;
- иерархия кэшей;
- ICP, HTCP, CARP, Cache digests;
- прозрачное Proxy;
- WCCP;
- гибкий контроль доступа;
- HTTP-серверное ускорение;
- SNMP;
- кэширование DNS-запросов;
- возможность ограничения трафика.

Рассмотрим некоторые из этих функций подробнее.

Протокол ICP

Протокол ICP используется в иерархии кэшей для поиска объектов в дереве кэшей Squid-серверов. Если ваш Squid не находит нужного документа, то посылает ICP-запрос другим Squid-серверам, входящим в вашу иерархию Proxy-серверов. Эти серверы отвечают ICP ответами `HIT` (попадание) или `MISS` (промах). После получения ответов, ваш сервер решает, при помощи какого кэша Proxy-сервера получить необходимые ему данные.

Cache digest

Cache digest — компактная форма представления списка содержимого кэша Proxy-сервера. Proxy-серверы могут обмениваться этой информацией с другими Proxy-серверами для избежания необходимости делать ICP-запросы (экономия трафика). В качестве ключей объектов используется протокол шифрования MD5.

Иерархия кэшей

Иерархия кэшей — это структура кэширующих Proxy-серверов, расположенных логически как родительский/дочерний и братский узлы таким образом, что кэши, ближайшие к интернет-каналу, являются родителями тем Proxy-серверам, которые находятся дальше от точки доступа в Интернет. В случае, когда кэш запрашивает объект от родителя, и у того в кэше необходимый объект отсутствует, родительский Proxy-сервер получает объект из Интернета, кэширует его и передает дочернему. Таким образом, при помощи иерархии достигается максимальная разгрузка канала.

Кроме родительских/дочерних отношений, Squid поддерживает понятие братских кэшей — находящихся на одном уровне иерархии. Каждый Proxy-сервер в иерархии независимо ни от кого решает, откуда получать необходимый объект — напрямую из Интернета, от родительского или братского кэша.

Алгоритм получения запрошенного объекта

Приведем алгоритм получения запрошенного объекта пакетом Squid.

1. Разослать ICP-запросы всем братским кэшам.
2. Дождаться всех ответов, пришедших в течение заданного времени:
   - получив первый ответ `HIT` (попадание), получить объект;
   - взять объект от первого родительского кэша, ответившего `MISS` (зависит от настройки);
   - получить объект из Интернета при отсутствии объекта в братских кэшах.

Конфигурирование пакета Squid

Основной конфигурационный файл пакета Squid — /etc/Squid.conf. Размер этого файла достаточно велик, поскольку он содержит множество параметров, начиная с номера порта для ICP-запросов и заканчивая правилами доступа к информации. Далее приведены параметры конфигурации Squid-сервера, разбитые на типы. Однако приведенный список не является полным и исчерпывающим, поскольку он содержит только основные параметры конфигурации, позволяющие настроить сервер для стандартных условий использования.

Сетевые параметры

Сетевые параметры Proxy-сервера обычно имеют несколько настроек.

- Порт для запросов клиентов Proxy-сервера:

```
http_port 3128
```

- Порт для ICP-запросов. В том случае, если не предполагается использовать иерархию Proxy-серверов — необходимо указать нулевой порт:

  `icp_port 3130`

- Порт для общения с соседями ICP — через TCP-протокол:

  `htcp_port 4827`

- К каким multicast-группам (соседи-серверы squid) подсоединяться для получения ICP, если используется multicast:

  `mcast_groups 239.128.16.128 224.0.1.20`

- По умолчанию режим пассивного FTP включен, но если Squid находится за брандмауэром, то необходимо выключить:

  `passive_ftp on | off`

Соседи

Squid может обмениваться информацией с другими Squid-серверами, которых принято называть соседями.

- Каждый сосед описывается отдельной строкой:

  `cache_peer hostname type proxy-port icp-port options`

  - Параметр `type` имеет следующие значения:
    - `parent` — старший в иерархии;
    - `sibling` — одного уровня.
  - Параметр `options` имеет следующие значения:
    - `proxy-only` — объекты, взятые с указанного узла, не хранить у себя в кэше;
    - `weight=<число>` — указывает приоритет хоста, чем значение больше, тем больше приоритет;
    - `ttl=<число>` — время жизни пакета используется при настройке multicast;
    - `no-query` — не посылать ICP-запросы;
    - `default` — самый старший в иерархии;
    - `round-robin` — определяет родительские кэши, используемые по очереди;
    - `multicast-responder` — данный сосед является членом multicast-группы;
    - `no-digest` — не запрашивать от этого соседа cashe digest;

  - `login=<user>:<password>` — определение имени и пароля для случая, если старший в иерархии Proxy-сервер требует аутентификации;
  - `connect-timeout=<число>` — время ожидания ответа от соседей;

- `cache_peer_domain host domain [domain...]` — ограничить запросы к данному соседу данным списком доменов;
- `icp_query_timeout <milisec>` — время ожидания ответа в миллисекундах;
- `mcast_icp_query_timeout <milisec>` — ожидание ответа на регулярные multicast-опросы;
- `dead_peer_timeout <seconds>` — время ожидания ответа от соседа, по истечении которого считается, что сосед отсутствует в сети;
- `hierarchy_stoplist` — список строк (через пробел), при встрече которых в URL запрос не будет кэшироваться; по умолчанию cgi-bin;
- `no_cache deny <имя>-ACL` — определяет список объектов, которые не будут кэшироваться.

Размер кэша

Раздел предназначен для определения параметров кэша — размера, использования, времени хранения информации и т. п.

- `cache_mem 8 MB` — объем оперативной памяти, используемой для хранения обрабатываемых объектов;
- `cache_swap_high 95` — при достижении данного уровня заполнения кэша (в процентах) начинается ускоренный процесс очистки кэша от устаревших объектов;
- `cache_swap_low 90` — процесс удаления старых объектов заканчивается, если достигнут данный уровень (в процентах);
- `maximum_object_size 4096 KB` — максимальный размер кэшируемого объекта;
- `minimum_object_size 0 KB` — минимальный размер кэшируемого объекта; файлы меньшего размера не сохраняются;
- `ipcache_size 1024` — размер кэша для IP-адресов;
- `ipcache_high 95` — верхний уровень заполнения IP-кэша для алгоритма удаления старых объектов;
- `ipcache_low 90` — нижний уровень заполнения IP-кэша для алгоритма удаления старых объектов.

Имена и размеры файлов

В этом разделе определяются имена и размеры используемых файлов.

- `cache_dir <тип> Directory-Name Mbytes Level-1 Level2` — определяет имя, размер и количество подкаталогов на первом и втором уровне кэша на

диске — каждый кэшируемый объект записывается в отдельный файл, файлы хранятся в двухуровневой иерархии каталогов;

- `cache_access_log /usr/local/squid/logs/access.log` — место хранения журнала обращений к кэшу;
- `cache_log /usr/local/squid/logs/cache.log` — место хранения журнала запусков процессов;
- `cache_store_log /usr/local/squid/logs/store.log` — место хранения журнала записи объектов в дисковый кэш;
- `emulate_httpd_log on|off` — производить ли эмуляцию формата журнала HTTPD;
- `mime_table /usr/local/squid/etc/mime.conf` — таблица типов MIME;
- `log_mime_hdrs off` — в журнал access записываются полученные HTTP-заголовки;
- `useragent_log` *<имя-файла>* — в этот файл будут записываться строки User-agent из HTTP-заголовков;
- `debug_options` *<раздел>*, *<уровень>* — уровень отладки; `ALL` — для всех разделов; по умолчанию `ALL,1`;
- `log_fqdn off` — позволяет определять и записывать в журнал полные доменные имена источника запроса.

Параметры внешних программ

Как и большинство серьезных программ, Squid позволяет воспользоваться внешними программами для выполнения некоторых действий. К примеру — сбор статистики или обработка трафика.

- `ftp_user` *<email-адрес>* — будет подставляться вместо пароля при анонимном доступе к FTP-серверам; по умолчанию — `Squid@`, вызывает проблемы с серверами, которые проверяют синтаксис адреса.
- `cache_dns_program /usr/local/squid/bin/dnsserver` — местоположение программы, кэширующей DNS-запросы.
- `dns_children 5` — число процессов, которые делают DNS lookup (получение по IP-адресу доменного имени и наоборот).
- `dns_nameservers` *<список-IP-адресов>* — используется вместо списка DNS-серверов, определенного в /etc/resolv.conf.
- `redirect_program none` — позволяет подключить программу преобразования URL при каждом запросе.
- `redirect_children 5` — параметр определяет, сколько процессов преобразования URL запускать параллельно.

- `redirect_rewrites_host_header on` — разрешает или запрещает изменение поля `Host:` в заголовке запроса; по умолчанию Squid переписывает поле `Host:` в заголовках преобразованных запросов.
- redirector_access acl — какие запросы направлять через редиректор; по умолчанию — все.
- `authenticate_program none` — позволяет производить аутентификацию клиентов, делающих запросы; программа должна в цикле читать строку "имя пароль" выдавать `OK` или `ERR`; должен быть определен параметр `ACL proxy_auth`.
- `authenticate_children 5` — сколько параллельных процессов будут заниматься аутентификацией.
- `authenticate_ttl 3600` — сколько секунд кэшировать результаты работ программы аутентификации.
- `authenticate_ip_ttl` *<число>* — необходимо установить 0, чтобы с нескольких адресов не смогли воспользоваться одним именем.

Тонкая настройка кэша

С помощью нижеприведенных параметров можно произвести тонкую настройку параметров кэша.

- `wais_relay_host localhost` — куда перенаправлять WAIS-запросы.
- `wais_relay_port 8000` — куда перенаправлять WAIS-запросы.
- `request_header_max_size 10KB` — максимальный размер заголовка.
- `request_body_max_size 1 MB` — максимальный размер объекта.
- `refresh_pattern [-i] regex MIN_AGE percent MAX_AGE[options]` — используется для определения, не устарел ли объект в кэше.

  Имя объекта сравнивается по очереди с регулярными выражениями в строках `refresh_pattern` до первого совпадения, параметры из соответствующей строки используются в алгоритме проверки. По умолчанию, регулярные выражения различают прописные/строчные буквы, чтобы игнорировать это различие, используется ключ `-i`. `MIN_AGE` и `MAX_AGE` — время жизни объекта в минутах. По умолчанию:

  - `refresh_pattern ^ftp: 1440 20% 10080`
  - `refresh_pattern ^gopher: 1440 0% 1440`
  - `refresh_pattern. 0 20% 4320`

  Более подробную информацию смотрите в документации на Squid.
- `reference_age 1 month` — максимальное время хранения неиспользуемого объекта до его удаления.

- `quick_abort_min 16 KB` — если клиент оборвал запрос, а осталось докачать всего `min KB`, то Squid произведет докачку объекта.
- `quick_abort_max 16 KB` — если клиент оборвал запрос, и осталось качать больше `max KB`, то Squid прекратит получение объекта.
- `quick_abort_pct` *<число>* — если клиент оборвал запрос, и уже получено больше чем *<число>* процентов объекта, то Squid докачает объект.
- `negative_ttl 5 minutes` — время кэширования негативных ответов (например "connection refused ", "404 not found") — число задает их время жизни в кэше.
- `positive_dns_ttl 6 hours` — время кэширования положительных DNS-ответов — число задает их время жизни в кэше.
- `negative_dns_ttl 5 minutes` — время кэширования негативных DNS-ответов — число задает их время жизни в кэше.
- `range_offset_limit 0 KB` — если клиент делает запрос с середины объекта, то:
  - `1` — вынуждает Squid загрузить весь объект в кэш до того, как начать передачу клиенту;
  - `0` — означает, что Squid никогда не будет грузить больше, чем клиент запросил;
  - число, отличное от `1` — если начало запроса меньше этого числа, то Squid будет грузить весь объект.

Время ожидания

В этом разделе задаются различные временные параметры Squid.

- `connect_timeout 120 seconds` — время ожидания соединения с сервером.
- `siteselect_timeout 4 seconds` — максимальное время на выбор URL.
- `read_timeout 15 minutes` — сколько времени разрешается ждать следующего байта от сервера.
- `request_timeout 30 seconds` — сколько разрешается ждать запроса после установления соединения.
- `client_lifetime 1 day` — сколько времени разрешать клиенту быть присоединенным к Squid. (Соединение обрывается, даже если происходит передача данных!)
- `half_closed_clients on` — разрешать наполовину закрытые соединения, например, чтение есть, а запись уже закрыта.
- `shutdown_lifetime 30 seconds` — сколько времени продолжать обслуживание после получения сигнала `SIGTERM` или `SIGHUP`.

ACL

Этот раздел определяет правила доступа пользователей к группам файлов и хостов. С помощью ACL (Access Control List, список контроля доступа) можно очень гибко настроить доступ к различным сайтам. Определение списка доступа производится с помощью следующей команды:

```
acl <имя> <тип> <строка>
```

где *<имя>* — имя правила, *<тип>* — тип объекта, *<строка>* — регулярное выражение (шаблон для сравнения), по умолчанию чувствительное к регистру букв.

Параметр *<тип>* может принимать следующие значения:

- IP-адреса клиентов:

  ```
  src ip-address/netmask...
  ```

- диапазон адресов:

  ```
  src addr1-addr2/netmask...
  ```

- получение IP-адреса по URL:

  ```
  srcdomain foo.com...
  ```

- если в URL использовался IP, то делается попытка определить имя домена, если не удалась, то подставляется слово `none`:

  ```
  dstdomain foo.com...
  ```

- получение IP-адреса клиента по URL с использованием регулярных выражений:

  ```
  srcdom_regex [-i] <строка>...
  ```

- если в URL применялся IP, то делается попытка определить имя домена, используя регулярные выражения:

  ```
  dstdom_regex [-i] <строка>...
  ```

- регулярное выражение для всего URL:

  ```
  url_regex [-i] <строка>
  ```

- регулярное выражение для Path-части URL:

  ```
  urlpath_regex [-i] <строка>
  ```

- определение безопасных портов:

  ```
  port <порт>...
  ```

- сопоставление заголовка User-Agent:

  ```
  browser [-i] regexp
  ```

- ограничение числа соединений с одного и того же IP:

  `maxconn <число>`

Права доступа

Права доступа определяются следующими строками:

- `http_access allow|deny [!]aclname...` — кому разрешать доступ к Proxy по HTTP;
- `icp_access allow|deny [!]aclname...` — кому разрешать доступ к Proxy по ICP;
- `miss_access allow|deny [!]aclname...` — кому разрешить получать ответ `MISS`;
- `cache_peer_access cache-host allow|deny [!]aclname...` — ограничить запросы к данному соседу;
- `proxy_auth_realm Squid proxy-caching web server` — строка текста, которая будет выдана на экран клиента при запросе имени/пароля доступа к кэшу.

Параметры администрирования

Параметры администрирования определяются следующими строками:

- `cache_mgr email` — почтовый адрес, на который будет послано письмо, если у Squid возникнут проблемы;
- `cache_effective_user nobody` — если запускается Squid от имени `root`, то заменить UID на указанный;
- `cache_effective_group nogroup` — если запускается Squid от группы `root`, то заменить GID на указанный;
- `visible_hostname <имя-хоста>` — это имя будет упоминаться в сообщениях об ошибках;
- `unique_hostname <уникальное-имя>` — если нескольким кэшам дали одно и то же `visible_hostname`, необходимо определить каждому из них уникальное имя;
- `hostname_aliases <имя>...` — список синонимов для имени хоста.

Параметры для работы в режиме ускорителя HTTP-сервера

Параметры для работы в режиме ускорителя HTTP-сервера определяются следующими строками:

- `httpd_accel_host hostname` — если нужна поддержка виртуальных хостов, в частности для прозрачного кэширования (transparent proxy), то вместо имени указать `virtual`;

- `httpd_accel_port port` — порт для HTTP-сервера;
- `httpd_accel_with_proxy on|off` — кэширование для ускоряемого сервера;
- `httpd_accel_uses_host_header on|off` — для работы в прозрачном режиме требуется включить (on), иначе виртуальные серверы не будут правильно кэшироваться.

Разное

Опишем параметры Squid, не вошедшие в предыдущие разделы:

- `dns_testnames internic.net microsoft.com` — список имен хостов, на примере которых проверяется работоспособность DNS;
- `logfile_rotate 10` — данный параметр задает количество старых копий при ротации;
- `append_domain.r.ru` — добавляется к имени хоста, если в нем нет ни одной точки;
- `tcp_recv_bufsize 0 bytes` – `0` означает, что надо использовать размер буфера по умолчанию;
- `err_html_text` *<строка>* — подставляется в шаблоны текстов сообщений об ошибках;
- `deny_info err_page_name acl` — запросы, не прошедшие проверку в `http_access`, проверяются на соответствие `ACL`, выдается соответствующее сообщение об ошибке из файла `page_name`;
- `memory_pools on|off` — эта переменная определяет политику использования захваченной памяти:
  - `on` — однажды захваченная, но не используемая память не отдается обратно в систему;
  - `off` — позволяет освобождать захваченную память;
- `memory_pools_limit` *<байт>* — максимальное количество неиспользуемой памяти, которое Squid будет удерживать, если `0` — удерживать все, что было захвачено;
- `forwarded_for on|off` — если включено, то Squid будет вставлять IP-адрес или имя в заголовки перенаправляемых HTTP-запросов: `X-Forwarded-For: 192.1.2.3`; если выключено, то `X-Forwarded-For: unknown`;
- `log_icp_queries on|off` — записываются ли в журнал ICP-запросы;
- `icp_hit_stale on|off` — возвращать ли ответ `ICP_HIT` для устаревших объектов;
- `cachemgr_passwd` *<password> <action> <action>*... — задание пароля для действий по администрированию Squid; чтобы запретить действие — по-

ставьте пароль `disable`; чтоб разрешить действие без проверки пароля — поставьте пароль `none`, кроме действий `config` и `shutdown`; полную информацию смотрите в документации на Squid;

- `store_avg_object_size 13 KB` — предполагаемый средний размер объекта, используемый для расчетов;
- `store_objects_per_bucket 20` — число объектов на хэш-корзину;
- `client_db on|off` — сбор статистики о клиентах;
- `netdb_low 900` — нижняя граница для базы данных измерения ICMP;
- `netdb_high 1000` — верхняя граница для базы данных измерения ICMP;
- `netdb_ping_period 5 minutes` — минимальное время между посылок Ping-пакетов в одну и ту же сеть;
- `query_icmp on|off` — должны ли соседи в ICP-ответы включать ICMP-данные;
- `test_reachability on|off` — если включить, то ответ `ICP_MISS` будет заменяться на `ICP_MISS_NOFETCH`, если сервер отсутствует в базе данных ICMP или RTT равен нулю;
- `buffered_logs on|off` — при включении запись в журнал буферизуется;
- `always_direct allow|deny [!]aclname...` — запросы, удовлеворяющие данным ACL, не кэшировать, а всегда направлять к первоисточнику;
- `never_direct allow|deny [!]aclname...` — запросы, удовлеворяющие данным ACL, всегда кэшировать;
- `anonymize_headers allow|deny header_name...` — перечень заголовков, которые нуждаются в анонимизации;
- `fake_user_agent none` — если заголовок `User-Agent` фильтруется с помощью анонимизатора, то подставляется эта строка;
- `minimum_retry_timeout 5 seconds` — если сервер имеет несколько IP-адресов, то тайм-аут соединения делится на их количество;
- `maximum_single_addr_tries 3` — сколько раз пытаться соединиться с сервером, имеющим один IP-адрес; если сервер имеет несколько IP-адресов, то каждый из них будет опробован один раз;
- `snmp_port 3401` — порт, который слушает Squid для SNMP-запросов;
- `snmp_access allow|deny [!]aclname...` — определяет, кто будет допущен к SNMP-порту;
- `offline_mode on|off` — если включить, то Squid будет брать объекты только из кэша и не будет пытаться обращаться к первоисточникам;
- `uri_whitespace strip` — что делать с запросами, имеющими пробелы в URI;

Возможные варианты:

- `strip` — удалять пробелы;
- `deny` — сообщать Invalid Request (ошибочный запрос);
- `allow` — передавать как есть;
- `encode` — кодировать в соответствии с RFC1738 и передавать дальше;
- `chop` — остаток после первого же пробела отбрасывать;

- `mcast_miss_addr адрес` — по этому multicast-адресу посылается сообщение при каждом "непопадании" в кэш;
- `mcast_miss_port порт` — этот порт используется для посылки сообщения;
- `strip_query_terms on` — удалять параметры запроса перед записью в журнал;
- `ignore_unknown_nameservers on` — игнорировать сообщения от DNS-серверов, с которыми Squid не работает.

Пример конфигурации Squid

Как вы уже заметили, опций для конфигурации Squid очень много. Для быстрой настройки Proxy-сервера можно воспользоваться приведенными далее параметрами. Конечно, они не являются идеальными, наверняка тонкая настройка поможет вам оптимизировать сервер как с точки зрения увеличения производительности, так и с точки зрения безопасности.

Возьмем стандартный файл Squid.conf и отредактируем только приведенные далее строки:

- `http_port 3128` — номер порта, на котором Squid будет слушать команды от клиентов;
- `hierarchy_stoplist cgi-bin, chat` — слова в URL, при обнаружении которых Proxy-сервер будет не кэшировать объекты, а напрямую перенаправлять запрос серверу;
- `cache_mem 16 MB` — сколько оперативной памяти Squid может забрать под свои нужды. Чем больше выделенной памяти, тем быстрее будут обрабатываться запросы (зависит от количества клиентов);
- `maximum_object_size 16384 KB` — максимальный размер объектов, которые будут сохранены в кэше. Размер специфичен для ваших задач и объема жесткого диска;
- `cache_dir /usr/local/Squid/cache 2048 16 256` — указывает Proxy-серверу, где сохранять кэшируемые файлы. Под кэш выделяется 2 Гбайт и создается 16 и 256 каталогов 1-го и 2-го уровня;
- `ftp_user anonymous@r.ru` — задает Proxy-серверу, под каким паролем регистрироваться на анонимных FTP-серверах;

- `negative_ttl 1 minutes` — время жизни страничек с ошибкой;
- `positive_dns_ttl 6 hours` — время жизни удачного преобразования DNS-имен в IP-адреса;
- `negative_dns_ttl 5 minutes` — время жизни неудачного преобразования DNS-имен в IP-адреса.

Дальнейшие настройки касаются разграничения прав пользователей.

Сначала необходимо определить ACL (Access Control List, список контроля доступа). Закомментируем все строчки в файле Squid.conf, начинающиеся на `acl`. Запишем свои правила. К примеру:

- `acl users proxy_auth vanya til petya nina` — этой строчкой мы указываем Proxy-серверу правило, по которому разрешаем пускать вышеперечисленных пользователей с использованием авторизующей программы через Squid;
- `acl BANNER url_regex banner reklama linkexch banpics us\.yimg\.com [\./]ad[s]?[\./]` — это правило определяет адреса, содержащие рекламу. Интересно для тех, кто хочет отказаться от получения разнообразных баннеров. Позволяет экономить сетевой трафик;
- `http_access deny !users` — эта строка запрещает доступ всем пользователям, кроме тех, которые перечислены в группе `users`;
- `http_access deny BANNER` — запрещаем доступ к URL, удовлетворяющим правилу `BANNER` (убираем рекламу);
- `proxy_auth_realm Vasy Pupkina proxy-caching web server` — строка, которая выводится в окно с логином/паролем;
- `cache_mgr vasya@r.ru` — если у клиента возникает проблема, выводится HTML-страница с сообщением и адресом электронной почты администратора, в нашем случае `vasya@r.ru`;
- `cache_effective_user nobody` — с правами какого пользователя выполняется Proxy-сервер;
- `cache_effective_group nogroup` — с правами какой группы выполняется Proxy-сервер;
- `client_db on` — параметр разрешает собирать статистику по клиентам.

Поскольку стандартной настройки в такой сфере, как использование канала, места на винчестере, оперативной памяти просто не может быть, более тонкие настройки и ограничения вы должны обдумать и определить сами.

Создание иерархии Proxy-серверов

Чтобы разместить кэш в иерархии, нужно воспользоваться директивой `cache_host`.

В приведенной далее части Squid.conf сервера r.ru настройки сделаны так, что кэш сервера получает данные с одного родительского и с двух братских кэшей:

```
cache_host petya.ru parent 3128 3130
cache_host monya.ru sibling 3128 3130
cache_host gesha.ru sibling 3128 3130
```

Директива `cache_host_domain` позволяет задавать для каждого определенного домена или группы доменов как братский, так и родительский кэш. Приведенный далее пример показывает что kesha.ru получает данные из доменов ru, au, aq, fj, nz, а gesha.ru — из доменов uk, de, fr, no, se, it.

```
cache_host kesha.ru parent 3128 3130
cache_host gesha.ru parent 3128 3130
cache_host uc.cache.nlanr.net sibling 3128 3130
cache_host bo.cache.nlanr.net sibling 3128 3130
cache_host_domain kesha.ru.ru.au.aq.fj.nz
cache_host_domain gesha.ru.uk.de.fr.no.se.it
```

Transparent proxy

Transparent proxy — Proxy-сервер, настроенный таким образом, что его использование прозрачно для пользователей. То есть пользователям не придется что-либо настраивать в своих браузерах. Для этого необходимо решить следующие задачи:

- добиться, чтобы все HTTP-запросы пользователей попали на компьютер, где работает ваш HTTP Proxy-сервер;
- добиться, чтобы эти запросы попадали собственно к Proxy-серверу;
- добиться того, чтобы ваш Proxy-сервер их правильно обработал.

Выполнить первый пункт можно разными способами. Самый простой путь — поставить Proxy-сервер и маршрутизатор на один сервер, через который проходит весь трафик.

Чтобы HTTP-запросы пользователей попали к HTTP Proxy-серверу, необходимо таким образом настроить маршрутизатор (брандмауэр), чтобы транзитные пакеты, предназначенные для порта 80, попадали на вход Proxy-сервера. Если Proxy-сервер должным образом настроен, он правильно обработает полученные запросы. В Squid.conf добавляются следующие строки:

```
httpd_accel www.your.domain 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Ключи запуска Squid

Помимо конфигурационного файла, поведением программы Squid можно управлять с помощью ключей командной строки. Приведем некоторые из них:

- `-a` — указывает порт для входных HTTP-запросов;
- `-d` — выводит отладочную информацию на устройство stderr (обычно — текущая консоль);
- `-f` *<имя_файла_конфигурации>* — позволяет использовать альтернативный конфигурационный файл (удобно для отладки сервера);
- `-h` — выводит краткую справку по программе Squid;
- `-k` — позволяет посылать Squid следующие управляющие сигналы:
  - `reconfigure` — посылка сигнала `HUP`. Используется для прочтения измененного конфигурационного файла;
  - `rotate` — позволяет произвести ротацию журналов (сигнал `USR1`);
  - `shutdown` — прервать выполнение программы с корректным завершением (сигнал `TERM`);
  - `interrupt` — немедленно завершить работу программы (сигнал `INT`);
  - `kill` — "убить" приложение (`KILL`);
  - `debug` — начать/закончить полную трассировку (сигнал `USR2`);
  - `check` — проверка (сигнал `ZERO`);
- `-u` — задает порт для входных ICP-запросов;
- `-v` — выводит версию программы;
- `-z` — создает дисковый кэш при первом запуске (Важно!);
- `-D` — предписывает не производить DNS-тест при запуске;
- `-F` — восстанавливает после сбоя не в фоновом режиме (ускорение восстановления);
- `-N` — предписывает не становиться фоновым процессом;
- `-V` — включает поддержку виртуальных хостов для режима акселерации;
- `-X` — включает отладку при разборе конфигурационного файла;
- `-Y` — включает быстрое восстановление после сбоев.

Первый раз Squid нужно запускать с ключом `-z`:

```
Squid -z
```

При этом программа создаст дерево кэшей. Этой же командой необходимо воспользоваться с том случае, если вам требуется очистить кэш Proxy-сервера.

Для закрытия текущих файлов журналов и создания новых (чистых) файлов используется команда:

```
Squid – k rotate
```

Файлы журналов Squid

В этом разделе описываются файлы журналов, с помощью которых можно посмотреть историю обращения пользователей к определенным сайтам, облегчить процесс отладки Proxy, посмотреть статистику используемых пользователями браузеров и т. п.

Файл access.log

Файл access.log используется для хранения информации о всех подключениях к Proxy-серверу. Запись добавляется, когда клиент закрывает соединение. Для сервера с большим трафиком файл может за день увеличиться на десятки мегабайт. К примеру, при трафике 10 тыс. запросов в сутки объем журнала увеличивается примерно на 2 Мбайт.

Единицей информации о соединении является строка. Строка состоит из десяти полей. Приведем описание полей:

- `timestamp` — время в UNIX-формате (время с 1 января 1970 года в миллисекундах);
- `elapsed` — затраченное время в миллисекундах;
- `client IP address` — IP-адрес клиента, пославшего запрос;
- `type/HTTP` — результат запроса, где `type`:
  - `TCP_HIT` — верная копия объекта нашлась в кэше;
  - `TCP_MISS` — запрашиваемый объект не был в кэше;
  - `TCP_EXPIRED` — объект есть в кэше, но он устарел;
  - `TCP_CLIENT_REFRESH` — клиент запросил принудительное обновление объекта;
  - `TCP_REFRESH_HIT` — объект в кэше был старым, был сделан запрос к источнику и источник ответил "объект не изменился";
  - `TCP_REFRESH_MISS` — объект в кэше был старым, был сделан запрос к источнику и тот вернул обновленное содержание;
  - `TCP_IMS_HIT` — клиент выдал запрос, объект оказался свежим в кэше;
  - `TCP_IMS_MISS` — клиент выдал запрос для просроченного объекта;
  - `TCP_REF_FAIL_HIT` — объект в кэше устарел, но запросить новую копию не удалось;

- `TCP_SWAPFAIL` — объект должен находиться в кэше, но его не смогли извлечь;
- `TCP_DENIED` — отказ;

- `size` — количество байтов, переданных клиенту;
- `method` – метод передачи информации; `GET`, `HEAD`, `POST` для TCP-запросов или `ICP_QUERY` для UDP-запросов;
- `URL` — адрес запрашиваемого объекта;
- `ident "-"` — если недоступен;
- `hierarchy data/Hostname` — результат запросов к братским/родительским кэшам:
  - `PARENT_HIT` — UDP-запрос к родительскому кэшу (parent) вернулся с подтверждением;
  - `PARENT_UDP_HIT_OBJECT` — объект оказался в родительском кэше и поместился в UDP-ответе;
  - `DIRECT` — объект был запрошен с оригинального сервера;
- тип содержимого (MIME-тип/подтип).

Файл store.log

Файл store.log используется для хранения информации о всех кэшируемых объектах Proxy-сервера. Единицей информации о соединении является строка. Она состоит из одиннадцати полей:

- `Time` — время в UNIX-формате (время с 1 января 1970 года в миллисекундах);
- `Action` — действие:
  - `RELEASE` — удален из кэша;
  - `SWAPOUT` — сохранен на диск;
  - `SWAPIN` — был на диске, загружен в память;
- `HTTP reply code` — код ответа HTTP-сервера;
- `HTTP Date` — дата создания объекта;
- `HTTP Last-Modified` — время последней модификации объекта;
- `HTTP Expires` — срок жизни объекта;
- `HTTP Content-Type` — тип объекта;
- `HTTP Content-Length` — размер объекта;
- реально полученное число байтов. В том случае, если не совпадает с предыдущим полем, объект не сохраняется;

- `HTTP method` — метод передачи информации (`GET`, `HEAD`, `POST`);
- `Access key` — ключ доступа (обычно URL).

Файл useragent.log

Предназначен для хранения информации о том, какими пользовательскими агентами (Web-браузерами) пользуются клиенты. Малоинтересен в практическом плане. Разве что для получения статистики по частоте использования тех или иных Web-браузеров.

Нестандартные применения

Функциональность программы Squid не ограничивается только функцией Proxy-сервера. У нее есть достаточно много других интересных применений. В этом разделе мы рассмотрим только некоторые из них.

Борьба с баннерами

Наверняка вам встречались Web-страницы, на которых нужной информации было от силы на килобайт, а рекламных баннеров (зачастую анимированных) — пять-шесть. Хорошо когда канал большой и бесплатный. Когда же пользуешься обычном коммутируемым соединением да еще платишь за соединение из своего кармана, каждый килобайт начинаешь считать. В этом случае можно настроить локальный сервер Squid таким образом, чтобы не происходила закачка ненужных баннеров. Для этого необходимо предпринять следующие действия:

Вариант 1

Простой вариант. На месте баннеров показываются разорванные (или перекрещенные прямоугольники) картинки (неполученные файлы).

1. Определяем сайты баннерных сетей и создаем для них регулярные выражения.
2. Создаем в каталоге /usr/local/Squid/etc следующие файлы:
   - banners_path_regex — содержит по одному регулярному выражению на строку;
   - banners_regex — содержит по одному регулярному выражению на строку;
   - banners_exclusion — это строки, трактуемые в предыдущих файлах как баннеры, но изменять которые не рекомендуется.
3. В Squid.conf добавляем следующие правила:

```
acl banners_path_regex urlpath_regex
"/usr/local/Squid/etc/banners_path_regex"
acl banners_regex url_regex "/usr/local/Squid/etc/banners_regex"
```

```
acl banners_exclusion url_regex "/usr/local/Squid/etc/banners_exclusion"
http_access deny banners_path_regex !banners_exclusion
http_access deny banners_regex !banners_exclusion
```

Вариант 2

Замена рекламных баннеров на свою картинку, которая находится на локальном для Proxy-сервера компьютере.

1. Определяем сайты баннерных сетей и создаем для них регулярные выражения.
2. На своем сервере создаем "заменитель" рекламных картинок — файл mybanner.gif.
3. Настраиваем редиректор в Squid.conf — redirect_program /usr/local/Squid /bin/ banner.pl.
4. Создаем прстой скрипт на Perl — banner.pl:

```
#!/usr/bin/perl
$|=1;
while (<>)
{
 s@регулярное-выражение@http://www.myhost.org/mybanner.gif@;
 print;
}
```

Разделение внешнего канала

Часто бывает так, что у вас есть внешний канал (скажем, 128 Кбит) и несколько групп пользователей с определенным приоритетом. Требуется, чтобы группа 1 имела одну фиксированную ширину наружного канала (скажем, 64 Кбит), а группа 2 и 3 — ширину наружного канала по 32 Кбит. Для решения этой непростой задачи мы также можем воспользоваться Squid.

Поясним используемую терминологию.

- Пул — набор групп "емкостей" определенного класса.
- Группа "емкостей" — часть пула, привязанная к хосту, сети или общая для всех.
- "Емкость" ограниченного объема — та, в которую с определенной скоростью вливается внешний трафик, и из которой он раздается клиенту.

Определены три класса пулов:

- одна "емкость" на всех из этого класса;
- одна общая "емкость" и 255 отдельных для каждого хоста из сети класса С;

- 255 "емкостей" для каждой сети класса B и отдельная "емкость" для каждого хоста.

Пример конфигурации Squid для трех классов пулов:

```
delay_pools 3  # 3 poolы
delay_class 1 1 # 1 pool 1 класса
delay_class 2 1 # 2 pool 1 класса
delay_class 3 3 # 3 pool 3 класса
delay_access 1 allow staff
delay_access 1 deny all
delay_access 2 allow students
delay_access 2 deny all
delay_access 3 allow college
delay_access 3 deny all
delay_parameters 1 640000/640000
delay_parameters 2 64000/64000
delay_parameters 3 64000/64000 32000/64000 6400/32000
```

Строка, определяющая максимальную ширину виртуального канала, имеет следующий вид:

```
delay_parameters pool total_rest/total_max net_rest/net_max ind_rest/ind_max
```

где:

- `pool` — номер пула, для которого определяются каналы;
- `total` — ширина канала на всех;
- `net` — ширина канала на подсеть;
- `ind` — ширина канала на отдельный адрес;
- `rest` — скорость заполнения (байт/сек);
- `max` — объем "емкости" (байт).

Обработка статистики

В стандартную поставку пакета Squid входят скрипты, написанные на Perl и позволяющие создавать отчеты о работе программы Squid:

- access-extract.pl — скрипт получает на стандартный ввод журнал access.log и выдает на стандартный вывод промежуточный результат;
- access-summary.pl — скрипт получает на вход результат работы access-extract.pl и делает из него красивый отчет.

Программа Squid Cache and Web Utilities (SARG)

Программа Squid Cache and Web Utilities обрабатывает журналы Squid и составляет на их базе отчеты. С ее помощью можно получить следующую информацию:

- количество работавших пользователей;
- время их работы;
- трафик по каждому пользователю;
- использование кэша каждым пользователем;
- список Web-серверов, посещаемых пользователем;
- итоговые цифры по трафику и времени.

Существуют также дополнительные отчеты: Top sites и Useragents.

Отчет генерируется за период, интервал которого берется из LOG-файла Squid. В отчете, кроме зарегистрированных пользователей Squid, отражается информация о попытках незаконного вхождения в сеть и неправильных наборах пароля.

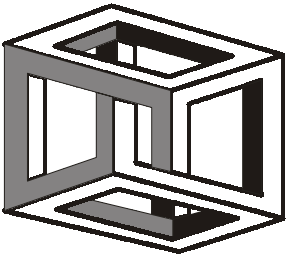
Если не производится ротация журналов Squid, то SARG генерирует отчеты нарастающим итогом. Отчеты хранятся в стандартном формате HTML-страниц, поэтому их можно просматривать через браузер, копировать и распечатывать. Также отчеты можно генерировать в определенный каталог или получать по почте.

Программа MRTG

Данная программа позволяет получать при соответствующей настройке отчеты о работе Squid. Вывод осуществляется в виде HTML-страниц.

Литература и ссылки

- **http://linux.webclub.ru/security/proxy/Squid.html** — Паскаль И. Настройка Squid.
- **http://karjagin.narod.ru/solaris/Squid-faq-rus.html** — русский перевод Squid-FAQ.
- **http://Squid.org.ua** — зона особого внимания: сайт, полностью посвященный программе Squid.
- **http://www.bog.pp.ru/work/Squid.html** — Bog BOS: Squid (кэширующий Proxy для HTTP): установка, настройка и использование.
- **http://www.nitek.ru/~igor/Squid** — борьба с баннерами.
- **http://www.nlanr.net/Cache/ICP/ICP-id.txt** — протокол Internet Cache Protocol.
- **http://www.Squid-cache.org** — официальный сайт Squid.



ГЛАВА 12

Синхронизация времени через сеть, настройка временной зоны

Для грамотно настроенной сети предприятия характерен учет всяких "мелочей", особенно когда эти "мелочи" таковыми не являются. В частности — системные дата/время компьютера. При разнообразных "разборах полетов" приводят различные журналы действий пользователя, и при неправильной настройке системного времени грош цена таким данным. Можно конечно руками производить изменения даты/времени, однако через пару-тройку недель вам это наскучит. Особенно неприятно, когда компьютеров несколько десятков, и время у всех должно быть синхронизировано. Для синхронизации системного времени создателями Интернета был предусмотрен специальный сервис — сетевой протокол времени (Network Time Protocol, NTP).

Сетевой протокол времени

Протокол NTP предназначен для синхронизации клиента или сервера точного времени с другим сервером точного времени или эталонным источником времени (радио, атомные часы и тому подобные устройства). Для локальной сети служба NTP способна обеспечить точность до миллисекунды, а для распределенной сети (в частности Интернета) достижима точность синхронизации порядка нескольких десятков миллисекунд. Последний стандарт этого протокола предусматривает криптографическую защиту передаваемых данных, одновременное подключение к нескольким серверам точного времени для достижения более точной синхронизации времени и повышения отказоустойчивости системы и многое другое.

Структура сети серверов точного времени многоуровневая. Главные серверы точного времени, напрямую подключенные к источнику эталонного времени, образуют первый уровень, серверы точного времени, присоединенные непосредственно к главным серверам, образуют второй уровень и т. д.

В качестве сетевого протокола используется протокол UDP, порт 123. Для увеличения надежности и точности получаемых данных применяется фильтрация, селекция и комбинация пакетов на принципах максимальной вероятности, а также несколько резервных серверов и путей передачи.

Для передачи и хранения времени используется беззнаковое 64-битовое число с фиксированной точкой, которое хранит число секунд в формате UTC. Старшие 32 бита — число секунд, младшие 32 бита — дробная часть секунд. Достижимая точность — 232 пикосекунды. 0 означает неопределенное время.

Классы обслуживания

Служба точного времени имеет несколько классов обслуживания клиентов:

- `multicast` — предназначен для использования в быстрой локальной сети со множеством клиентов, где отсутствует необходимость в высокой точности. Принцип действия — один или более NTP-серверов рассылают широковещательное сообщение, клиенты определяют время, предполагая, что задержка составляет несколько миллисекунд. Сервер не принимает ответных NTP-сообщений;
- `procedure-call` — предназначен для получения высокоточного времени. NTP-клиент посылает запрос на сервер точного времени, который обрабатывает запрос и немедленно посылает ответ. Сервер не синхронизируется с клиентом;
- `symmetric` — предназначен для использования серверами точного времени. Представляет собой динамически реконфигурируемую иерархию серверов точного времени. Каждый сервер точного времени синхронизирует своих соседей и синхронизируется своими соседями в соответствии с правилами выбора соседей. Активный режим используется серверами точного времени низшего уровня с заранее определенными адресами соседей, пассивный режим применяется серверами точного времени, близкими к первому уровню и взаимодействующими с соседями с заранее неизвестными адресами.

Обеспечение достоверности данных

Алгоритм функционирования сервера точного времени подразумевает несколько способов для обеспечения достоверности данных.

- Если в течение восьми последовательных интервалов опроса от соседнего сервера точного времени не было сообщений, то этот сервер считается недостижимым.
- Осуществляется проверка времени:
  - если время передачи совпадает со временем предыдущего сообщения — дублированный пакет;

- если время отправки сообщения не совпадает со временем, содержащимся в пакете, сервер считает, что он получил фальшивый пакет.

- Используется алгоритм защиты от очень старых сообщений.
- Аутентификатор состоит из ключа и шифрованной контрольной суммы, которая создается с использованием алгоритма шифрования DES.

Формат NTP-пакета

Пакет NTP включает следующие поля:

- `LI` (leap indicator) — в конце суток должна быть вставлена секунда для синхронизации атомных и астрономических часов;
- `VN` — номер версии протокола;
- `mode` — режим работы сервера точного времени;
- `stratum` — уровень сервера;
- `precision` — точность часов сервера;
- `poll interval` — интервал запросов. Используется наименьшее значение из двух интервалов: своего сервера и сервера, отвечающего на запросы;
- `synchronization distance` — полный цикл обмена сообщениями до первичного источника;
- `synchronization dispersion` — дисперсия задержек синхронизации;
- `reference clock identifier` — тип источника времени;
- `reference timestamp` — время последнего изменения источника времени;
- `originate timestamp` — время соседа, когда было отправлено последнее NTP-сообщение;
- `receive timestamp` — местное время получения последнего NTP-сообщения;
- `transmit timestamp` — местное время отправки текущего сообщения;
- `authenticator` (96 bit) — ключ и шифрованная контрольная сумма сообщения.

Рекомендуемая конфигурация

Рекомендуемая конфигурация подразумевает наличие трех местных серверов точного времени, соединенных между собой, каждый из которых подключен к двум различным внешним серверам. Клиенты службы точного времени подключаются к каждому местному серверу точного времени.

Стандарты

Стандарты, используемые для протокола NTP, приведены в табл. 12.1.

Таблица 12.1. *Стандарты протокола NTP*

Стандарт	Название	Примечание
RFC1128	Measured performance of the Network Time Protocol in the Internet system (Измерение производительности сетевого протокола времени в Интернете)	
RFC1129	Internet time synchronization: The Network Protocol (Синхронизация времени через Интернет: сетевой протокол времени)	Описывает процесс синхронизации времени
RFC1165	Network Time Protocol (NTP) over the OSI Remote Operations Service (Сетевой протокол времени и взаимодействие с моделью OSI)	
RFC1305	Network Time Protocol (v3) (Сетевой протокол времени, третья версия)	Отменил стандарты RFC1119, RFC1059, RFC958
RFC2030	Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI (Простой сетевой протокол времени, четвертая версия для IPv4, IPv6 и OSI)	Упрощенный по сравнению с NTP протокол рекомендуется к использованию там, где нет необходимости для прецизионной синхронизации времени

Сервер xntpd

Для UNIX-платформы, в том числе и Linux, существует сервер точного времени, носящий название xntpd. Этот сервер полностью реализует стандарт RFC1305 и имеет расширенные возможности, которые планируется включить в следующую версию стандарта. Входит в стандартную поставку большинства дистрибутивов Linux. Установка тривиальна. Файл конфигурации — /etc/ntp.conf.

Конфигурация сервера

Поскольку варианты конфигурирования сервера зависят от класса обслуживания, сервер имеет достаточно много настроек, которые в основном содержатся в конфигурационном файле /etc/ntp.conf.

Класс *symmetric*

Этот класс предназначен для конфигурирования сервера точного времени в режиме `symmetric`.

```
peer <address> [key <key>] [version <version>] [prefer]
[minpoll <minpoll>] [maxpoll <maxpoll>]
```

Здесь:

- *<address>* — адрес симметричного сервера;
- *<key>* — 32-битовый ключ для поля аутентификации (по умолчанию отсутствует);
- `prefer` — предпочитать данный сервер при прочих равных условиях;
- *<minpoll>* — минимальный интервал запросов (секунды, 2 в степени *<minpoll>* в диапазоне от 4 (16 с) до 14 (16 384 с), по умолчанию 6 (64 с));
- *<maxpoll>* — максимальный интервал запросов (секунды, 2 в степени *<maxpoll>*, по умолчанию 10—1024 с).

Класс *procedure-call*

Этот класс предназначен для конфигурирования сервера точного времени в режиме `procedure-call`.

```
server address [key <key>] [version <version>] [prefer] [mode <mode>]
```

- *<address>* — адрес сервера;
- *<key>* — 32-битовый ключ для поля аутентификации (по умолчанию отсутствует);
- *<mode>* — режим.

Класс *multicast*

Предназначен для настройки режима `multicast`. Обычно используется в локальных сетях.

- `broadcast <address> [key <key>] [version <version>] [ttl <ttl>]`
  - *<address>* — адрес симметричного сервера;
  - *<key>* — 32-битовый ключ для поля аутентификации (по умолчанию отсутствует);
  - *<version>* — версия протокола;
  - *<ttl>* — время жизни пакета;
- `broadcastclient [<address>] <address>` — адрес клиента, получающего информацию;

- `broadcastdelay <секунд>` — позволяет самостоятельно указать задержку в распространении пакета.

Общие параметры

Опишем общие параметры настройки сервера xntpd:

- `driftfile <driftfile>` — определяет файл, в котором хранится и извлекается при запуске сдвиг частоты местных часов;
- `enable/disable auth/monitor/pll/pps/stats` — включить/выключить режим работы:
  - `auth` — с неупомянутыми соседями общаться только в режиме аутентификации;
  - `monitor` — разрешить мониторинг запросов;
  - `pll` — разрешать настраивать частоту местных часов по NTP;
  - `stats` — разрешить сбор статистики;
- `statistics loopstats` — при каждой модификации локальных часов записывает строчку в файл loopstats, формат которого имеет вид:
  - номер модифицированного Юлианского дня;
  - секунды с полуночи (UTC);
  - смещение в секундах;
  - смещение частоты в миллионных долях;
  - временная константа алгоритма дисциплинирования часов;
- `statistics peerstats` — каждое общение с соседом записывается в журнал, хранящийся в файле peerstats, формат которого имеет вид:
  - номер модифицированного Юлианского дня;
  - секунды с полуночи (UTC);
  - IP-адрес соседа;
  - статус соседа, шестнадцатеричное число;
  - смещение, сек.;
  - задержка, сек.;
  - дисперсия, сек.;
- `statistics clockstats` — каждое сообщение от драйвера локальных часов записывается в журнал, хранящийся в файле clockstats;
- `statsdir <имя-каталога-со-статистикой>` — задает имя каталога, в котором будут находиться файлы со статистикой сервера;

- `filegen [file <filename>] [type <typename>] [flag <flagval>] [link | nolink] [enable | disable]` — определяет алгоритм генерации имен файлов, которые состоят из:
  - префикс — постоянная часть имени файла, задается либо при компиляции, либо специальными командами конфигурации;
  - имя файла — добавляется к префиксу без косой черты, две точки запрещены, может быть изменена ключом `file`;
  - суффикс — генерируется в зависимости от *<typename>*:
    - `none` — обычный файл;
    - `pid` — при каждом запуске xntpd создается новый файл (к префиксу и имени файла добавляются точка и номер процесса);
    - `day` — каждый день создается новый файл (к префиксу и имени файла добавляется .yyyymmdd);
    - `week` — каждую неделю создается новый файл (к префиксу и имени файла добавляется .yyyywww);
    - `month` — каждый месяц создается новый файл (к префиксу и имени файла добавляется .yyyymm);
    - `year` — каждый год создается новый файл (к префиксу и имени файла добавляется .yyyy);
    - `age` — новый файл создается каждые 24 часа (к префиксу и имени файла добавляются .a и 8-значное количество секунд на момент создания файла от момента запуска xntpd);
    - `link/nolink` — по умолчанию создается жесткая ссылка от файла без суффикса к текущему элементу набора (это позволяет обратиться к текущему файлу из набора, используя постоянное имя);
    - `enable/disable` — разрешает/запрещает запись в соответствующий набор файлов;
- `restrict numeric-address [ mask <numeric-mask> ] [flag] ...` — задает ограничение доступа: пакеты сортируются по адресам и маскам, берется исходный адрес и последовательно сравнивается, от последнего удачного сравнения берется флаг доступа:
  - нет флагов — дать доступ;
  - `ignore` — игнорировать все пакеты;
  - `noquery` — игнорировать пакеты NTP 6 и 7 (запрос и модификация состояния);
  - `nomodify` — игнорировать пакеты NTP 6 и 7 (модификация состояния);

- `notrap` — отказать в обеспечении mode 6 trap-сервиса (удаленная журнализация событий);
- `lowpriotrap` — обслуживать ловушки, но прекращать обслуживание, если более приоритетный клиент потребует этого;
- `noserve` — обслуживать только запросы mode 6 и 7;
- `nopeer` — обслуживать хост, но не синхронизироваться с ним;
- `notrust` — не рассматривать как источник синхронизации;
- `limited` — обслуживать только ограниченное количество клиентов из данной сети;
- `ntpport/non-ntpport` — модификатор алгоритма сравнения адресов (сравнение успешно, если исходный порт равен/не равен 123), алгоритм сортировки ставит эту строку в конец списка.

- `clientlimit limit` — для флага `limited` определяет максимальное количество обслуживаемых клиентов (по умолчанию 3);
- `clientperiod` *<секунд>* — сколько секунд считать клиента активным и учитывать при определении количества обслуживаемых клиентов;
- `trap host-address [port` *<port-number>*`] [interface` *<interface-address>*`]` — задать хост и порт, которые будут вести журнал;
- `setvar` *<variable>* — установка дополнительных переменных;
- `logfile` *<имя-файла>* — использовать файл *<имя-файла>* для ведения журнала вместо syslog;
- `logconfig` *<keyword>* — управление количеством сообщений, сбрасываемых в журнал. Ключевое слово может быть предварено символами равно (установка маски), минус (удаление класса сообщений), плюс (добавление); ключевое слово образуется слиянием класса сообщений (`clock`, `peer`, `sys`, `sync`) и класса событий (`info`, `event`, `statistics`, `status`); в качестве суффикса или префикса может использоваться слово `all`.

Обеспечение безопасности сервера

Если сервер точного времени используется только внутренней локальной сетью, желательно закрыть порт 123 для доступа извне, чтобы избежать возможной атаки типа denial of service (отказ в обслуживании), поскольку это грозит неправильным функционированием сервера. Помимо этого необходимо использовать шифрование трафика.

Программы и утилиты, относящиеся к службе точного времени

В этом разделе приведено несколько утилит, используемых в различных операционных системах для синхронизации с серверами точного времени. Таких программ много, особенно для Windows, поэтому описаны только некоторые из них.

ntpdate

Эта утилита позволяет установить время на компьютере, используя список NTP-серверов.

Используемые ключи:

- `-B` — только плавный сдвиг, даже если смещение больше 128 мс;
- `-b` — всегда использовать settimeofday;
- `-d` — отладка;
- `-p` *<число>* — число запросов к каждому серверу (от 1 до 8, по умолчанию 4);
- `-q` — только запрос времени;
- `-s` — использовать syslog вместо stdout;
- `-t` *<timeout>* — время ожидания ответа (по умолчанию 1 сек.);
- `-u` — использовать непривилегированный порт.

ntpq

Утилита для получения состояния NTP-сервера и его изменения (использует NTP mode 6).

ntptrace

Утилита для поиска серверов первого уровня.

Используемые ключи:

- `-r` *<число>* — количество запросов (по умолчанию 5);
- `-t` *<секунд>* — время ожидания ответа (по умолчанию 2).

xntpd

Собственно демон точного времени. Используемые параметры при запуске:

```
xntpd [-aAbdm ] [-c <config-file>] [-f <drift-file>] [-k <key-file>]
   [-l <log-file>] [-p <pid-file>] [-r <broadcast-delay>] [-s <stats-dir]
   [-t <key>] [-v <variable>] [-V <variable>]
```

Здесь:

- `-a` — разрешить аутентификацию;
- `-A` — запретить аутентификацию;
- `-b` — широковещательные сообщения;
- `-c <config-file>` — конфигурационный файл (по умолчанию /etc /ntp.conf);
- `-d` — отладка;
- `-f <drift-file>` — файл, хранящий смещение часов (по умолчанию /etc /ntp.drift);
- `-k <key-file>` — файл ключей (по умолчанию /etc/ntp.keys);
- `-l <log-file>` — файл протокола (по умолчанию syslog).

xntpdc

Утилита для запроса состояния NTP-сервера и его изменения. Применяется только для Xntpd-серверов. Использует NTP mode 7.

Публичные NTP-серверы

Список публичных серверов точного времени можно найти в Интернете. В любом случае вам придется протестировать серверы из этого списка, чтобы определить задержки и качество соединения. Попробуйте сначала получить список серверов точного времени вашего провайдера (провайдеров). В списке литературы приведена ссылка на список серверов точного времени первого и второго уровней, можно попробовать синхронизироваться от них.

Клиентские программы для синхронизации времени

Сам по себе сервер точного времени бесполезен, если у пользователей отсутствует программное обеспечение для синхронизации даты/времени. В настоящее время практически для всех операционных систем есть программы получения времени с серверов NTP. Некоторые из них приведены далее.

UNIX/Linux

Для этих операционных систем можно на компьютере установить сервер xntpd и настроить его для получения точного времени. У этого решения есть как достоинства, так и недостатки. Положительным моментом является то, что мы можем максимально точно синхронизировать время и построить отказоустойчивую конфигурацию. Отрицательный момент — достаточно сложное конфигурирование сервера и относительно большой объем занимаемой оперативной памяти компьютера.

Более простой вариант — воспользоваться утилитой ntpdate. Она небольшая по размерам и простая в конфигурировании. С ее помощью можно получить достаточно точное время — расхождение порядка 100 миллисекунд. Для синхронизации времени достаточно выполнить следующую команду:

```
ntpdate -B <ntp> <ntp2> <ntp3>
```

где *<ntp>*, *<ntp2>*, *<ntp3>* — адреса серверов точного времени. Достаточно добавить эту строчку в таблицу заданий crontab и вы всегда будете иметь на компьютере точное время.

Apple

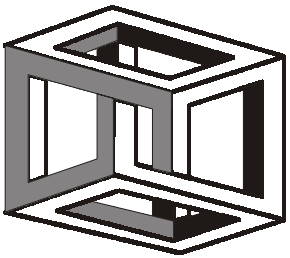
Для компьютеров фирмы Apple есть клиент NTP, называющийся macntp.

Windows

Для операционной системы Windows существует несколько десятков клиентов службы точного времени. В частности программа AboutTime, которую можно получить по адресу **http://www.listsoft.ru/programs/536/**. Или программа AnalogX Atomic TimeSync, получить которую можно по адресу **www.analogx.com/contents/download/network/ats.htm**.

Литература и ссылки

- **www.bog.pp.ru/work/ntp.html** — Богомолов С. Bog BOS: Network Time Protocol.
- **www.bog.pp.ru/work/xntpd.html** — Богомолов С. Bog BOS: xntpd (UNIX-сервер NTP — Network Time Protocol).
- **www.ntp.org** — страница, посвященная xntp.
- **cisco.opennet.ru/docs/RUS/lasg/time.html** — сетевые сервисы: NTP.
- **www.psn.ru/net/servis/ntp.shtml** — статья "Как пользоваться службой NTP?".
- **www.tomsknet.ru/ftp/docs/rfc/rfc1305.txt** — Network Time Protocol (Version 3) Specification, Implementation and Analysis (RFC1305).



ЧАСТЬ III

Сетевые ресурсы. Взаимодействие с другими операционными системами

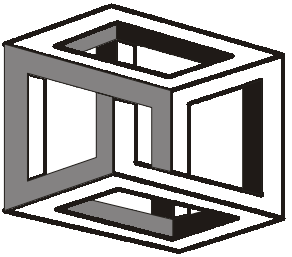
Данная часть является в какой-то мере продолжением *части II*, но, тем не менее, она стоит несколько особняком от предыдущего материала. Поскольку мы живем не в идеальном мире, нам приходится идти на компромиссы — наша сеть не является гомогенной средой и волей-неволей приходится взаимодействовать с различными операционными системами. В этой части мы ознакомимся с NFS (сетевой файловой системой UNIX), научимся предоставлять и получать доступ к каталогам операционных систем Windows и Novell.

Глава 13. NFS — сетевая файловая система

Глава 14. Сервер Samba для клиентов Windows

Глава 15. Mars — клиентам Novell

ГЛАВА 13



NFS — сетевая файловая система

NFS (Network File System, сетевая файловая система) — это программное обеспечение, позволяющее предоставлять в общее сетевое использование дисковое пространство хоста, включать в локальное дерево каталогов сетевые ресурсы, предоставленные в общее пользование. NFS позволяет работать с удаленной файловой системой через разные типы сетевых соединений: локальные сети, беспроводные сети, модемные соединения. NFS разработана и используется давно — еще с начала 80-х годов. Ее реализация и поддержка присутствует в любой версии UNIX и Windows.

Не будем пока заострять внимание на безопасности и надежности NFS, а перейдем сразу к ее процессу установки и настройки на Linux-компьютере.

Установка и настройка NFS-сервера

Как и большинство сетевых служб — NFS является клиент-серверной системой. И как обычно — настройка серверной части сложнее, чем клиентской. С нее мы и начнем.

Установить NFS не составляет проблем, поскольку в большинстве дистрибутивов она входит в стандартную поставку. После установки пакета необходимо произвести его настройку и конфигурирование.

К сожалению (для администратора системы), для функционирования NFS необходимо наличие службы RPC (Remote Procedure Called, служба вызова удаленных процедур), в Linux это пакет portmap.

Для указания хостов, которые имеют право доступа к службе RPC, используется конфигурационный файл /etc/hosts.allow, а для запрета доступа к службе — файл /etc/hosts.deny.

Для указания, какой каталог файловой системы можно предоставить в пользование удаленным компьютерам (экспортировать) и каким компьютерам можно монтировать эти каталоги, используется файл /etc/exports.

Этот файл содержит строки в следующем формате:

```
Полный_путь_к_каталогу
    ↳ Имя_хоста_которому_разрешено_монтирование_каталога (Права_доступа)
```

К примеру:

```
/home/boss/documents  zam(ro)
```

Здесь хост с именем `zam` может читать файлы (`ro`) из каталога `/home/boss/documents`, если вам нужна возможность записи, воспользуйтесь опцией (`rw`). Помимо этой возможности управления доступом можно воспользоваться NIS или LDAP.

После того как определены хосты, которым можно иметь доступ к экспортируемым ресурсам, приступим к тестированию NFS. Для нормального функционирования NFS необходимо наличие следующих демонов: portmapper, mountd и nfsd. Проверить наличие этих демонов несложно:

```
rpcinfo -p.
```

Результат программы должен быть следующим:

```
program vers proto   port
  100000    2   tcp    111  portmapper
  100000    2   udp    111  portmapper
  100005    1   udp    745  mountd
  100005    1   tcp    747  mountd
  100003    2   udp   2049  nfs
  100003    2   tcp   2049  nfs
```

В том случае, если выполнение программы завершилось ошибкой, необходимо убедиться, что у вас установлены и запущены ранее перечисленные программы и правильно заполнены файлы hosts.allow и hosts.deny.

При изменении файла exports необходимо заставить демон nfsd перечитать файл exports. Это можно сделать, выполнив следующие команды:

```
killall -HUP /usr/sbin/mountd
killall -HUP /usr/sbin/nfsd
```

Установка и настройка NFS-клиента

Установка NFS-клиента не вызывает проблем, в большинстве дистрибутивов он входит в стандартную поставку. Единственно, перед установкой клиента необходимо убедиться, что ядро собрано с поддержкой файловой системы NFS.

После установки клиента попробуем подмонтировать удаленную файловую систему.

Пусть мы хотим смонтировать /home/boss/documents, расположенного на хосте boss. Это делается с помощью команды:

```
mount -o rsize=1024,wsize=1024 boss:/home/boss/documents /mnt/docs
```

Если вместо монтирования файловой системы команда `mount` выдаст сообщение об ошибке:

```
mount: boss:/home/boss/documents failed, reason given by server:
Permission denied
```

то в файле exports не разрешен доступ к этому ресурсу для хоста, на котором пытаются смонтировать удаленный каталог.

Чтобы прекратить использование файловой системы, необходимо выполнить:

```
umount /mnt
```

Для автоматического монтирования файловой системы NFS при загрузке системы, в файл /etc/fstab необходимо добавить следующую строку:

```
boss:/home/bosss/documents /mnt/docs nfs rsize=1024, wsize=1024,
↳hard, intr 0 0
```

Опции монтирования

В этом разделе мы более подробно рассмотрим некоторые опции монтирования удаленной файловой системы.

Поскольку работа с удаленными источниками данных априори на несколько порядков менее надежна, необходимо тщательно подходить к опциям монтирования удаленной файловой системы, чтобы не возникало проблем (вплоть до потери реакции системы) при обрывах сетевого соединения. Также грамотно настроенные параметры монтирования помогут увеличить производительность сетевого доступа к удаленной файловой системе.

rsize

Опция `rsize` позволяет задавать размер блока чтения (в байтах). По умолчанию используется размер блока 8192 байт.

wsize

Опция `wsize` позволяет задавать размер блока записи (в байтах). По умолчанию используется размер блока 8192 байт.

Применяя эти параметры можно добиться максимальной производительности на медленных и неустойчивых соединениях (например, модемных).

soft

NFS-клиент будет сообщать об ошибке программе, которая пытается получить доступ к файлу, расположенному на файловой системе, смонтированной через NFS. Не рекомендуется использовать эту опцию, поскольку она может привести к появлению испорченных файлов и потерянных данных.

hard

При использовании опции монтирования `hard` программа, осуществляющая доступ к файлу на смонтированной по NFS файловой системе, просто приостановит выполнение при разрыве связи с сервером. Процесс не может быть прерван или "убит" до тех пор, пока вы явно не укажите опцию `intr`.

timeo=n

Параметр *n* задает задержку в десятых долях секунды до посылки первой ретрансляции после тайм-аута RPC. По умолчанию эта величина равна 0,7 секунды. После первого тайм-аута, время тайм-аута удваивается после каждого тайм-аута, пока не будет достигнута величина в 60 секунд, или произойдет ретрансляция, заданная параметром `retrans`, вызвав главный тайм-аут.

retrans=n

Величина *n* задает количество не основных тайм-аутов и ретрансляций, которые должны произойти до возникновения главного тайм-аута. По умолчанию эта величина равна 3. Когда возникает главный тайм-аут, то файловые операции либо прерываются, либо выдается сообщение "server not responding".

Безопасность NFS

Основная проблема NFS заключается в том, что клиент, если не задано, будет доверять серверу, и наоборот. Это значит, что если запись администратора сервера NFS взломана, то также легко может быть взломана запись администратора клиентской машины. И наоборот.

Безопасность клиента

Для обеспечения безопасности на клиентской стороне необходимо запретить выполнение программ с установленным битом `suid` в файловой системе NFS. Для этого необходимо в опциях монтирования добавить опцию `nosuid`. Также рекомендуется запретить выполнение файлов на смонтированной файловой системе с помощью опции `noexec`, однако это не всегда

возможно, поскольку файловая система может содержать программы, которые необходимо выполнять.

Безопасность сервера

Для исключения возможности доступа с удаленного хоста к файлам, владельцем которого является пользователь root, необходимо указать опцию `root_squash` в файле exports:

```
/home/boss/documents zam(rw,root_squash)
```

Таким образом, если пользователь с UID 0 на стороне клиента попытается получить доступ, то файловый сервер выполнит подстановку UID пользователя nobody на сервере. Это означает, что администратор клиента не сможет получить доступ к файлам, к которым имеет доступ только администратор сервера. Аналогично с изменением файлов.

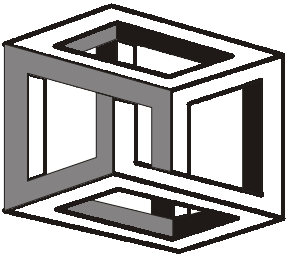
Помимо этого, вы можете запретить доступ определенным пользователям или группе пользователей удаленного компьютера. За более подробной информацией обратитесь к руководству по NFS.

Также необходимо с помощью файлов hosts.allow и hosts.deny разрешить доступ только тем хостам, которые будут использовать ваши экспортируемые ресурсы.

И наконец, защитите порты nfs, mountd и portmap с помощью firewall на вашем маршрутизаторе. Nfsd использует порт с номером 2049 и протоколы UDP и TCP. Portmapper использует порт номер 111 и протоколы TCP и UDP. Mountd использует порты 745 и 747 и протоколы TCP и UDP.

Литература и ссылки

- Справочные страницы man: NFS, portmap, mountd, nfsd, exports.
- NFS HOWTO.



ГЛАВА 14

Сервер Samba для клиентов Windows

В современном мире, как бы ни хотелось некоторым фирмам и личностям, невозможно продуктивно работать и существовать без взаимодействия и сотрудничества с конкурентами/соперниками. Вряд ли в вашей фирме есть только компьютеры под управлением Linux. Почти наверняка найдется парочка и с Windows, причем разных семейств. Само собой, хочется все это хозяйство "подружить", сделать более-менее прозрачный доступ к различным сетевым ресурсам.

Наверняка читатель знает, что *сетевое окружение* в понятиях Microsoft Windows достаточно сложная система, позволяющая организовывать совместный доступ к дискам, каталогам, принтерам, организовывать домен, использовать Active Directory и некоторые другие "прелести".

При этом достаточно много всяких тонкостей и ограничений возникает из-за различной реализации поддержки сети в разных версиях Windows. Кое-что исправляется патчами (patch), а некоторые ограничения невозможно исправить из-за особенностей архитектуры Windows различных версий.

Для корректного сосуществования Linux и Windows в мире UNIX существует пакет Samba, предназначенный для взаимодействия с клиентами сети Microsoft Windows.

Этот пакет позволяет Linux-системе выступать в качестве файл- и принт-сервера в сети Microsoft Windows, а также позволяет компьютеру под управлением Linux выступать в качестве первичного контроллера домена (Primary Domain Controller, PDC) сети Windows. Помимо этого есть Samba-клиент для операционной системы Linux, позволяющий Linux-клиенту подключаться к ресурсам, предоставляемым серверами сети Microsoft Windows.

Такая объединенная схема дает ряд преимуществ:

- поскольку в целом операционная система Linux устойчивее Windows 9*x* (а часто и Windows 2000/XP), повышается надежность функционирования системы;

- отпадает необходимость приобретать лицензионную Windows для организации сервера печати и доступа к данным;
- если у вас уже есть Linux-сервер, представляется рациональным нагрузить его дополнительной работой;
- сервер Samba имеет возможность мониторинга и удаленного управления как через SSH, так и через Web-интерфейс, предоставляемый пакетом SWAT (Samba Web-based Administrative Tool);
- и, наконец, по последним тестам на небольшом сервере SAMBA 3 работает значительно быстрее, чем Windows Server 2003.

Установка сервера Samba проблем не вызывает — достаточно при инсталляции Linux выбрать установку Samba. Если вы не установили Samba при инсталляции дистрибутива командой `rpm -i sambaXXX.rpm`, сервер будет установлен на вашем компьютере.

В том случае, если вы хотите установить самую последнюю версию пакета, и она досталась вам в виде TGZ-архива, содержащего исходный текст, процесс установки несколько растянется.

1. Сначала необходимо распаковать архив, содержащий исходные коды Samba. Для этого надо выполнить команду:

   ```
   tar zxvf samba-X.X.X.tar.gz
   ```

   где *X.X.X* — версия пакета.

2. После этого следует перейти в каталог /source, где находятся исходные коды. Также там есть и файл Readme, в котором подробно рассказано, как сконфигурировать, произвести компиляцию и установить пакет Samba.

3. Выполнить следующую команду:

   ```
   configure --with-smbmount --prefix=/opt/samba --with-msdfs
   ```

   Эта команда производит конфигурирование файла Makefile.

Замечание

Команда указывает компилятору компилировать утилиту smbmount, которая служит для монтирования SMB-ресурсов в файловую структуру Linux, включает поддержку Microsoft DFS и указывает каталог /opt/samba в качестве места установки после компиляции. Конечно, есть еще много параметров, которые можно назначить. Подробную информацию о них следует смотреть в документации к пакету Samba или вызывать командой `configure --help`.

4. Затем необходимо набрать в командной строке `make` и нажать клавишу <Enter>. Этой командой запускается процесс компиляции программного пакета.

5. Если в ходе работы программы make не появились сообщения об ошибках, то следующее, что следует сделать, — выполнить команду `make install`. Эта команда установит пакет Samba в ее родной каталог (если действовать в точности по инструкции, то файлы попадут в каталог /opt/samba).

Теперь на очереди конфигурирование сервера Samba.

Файл конфигурации smb.conf

Самое трудное, с чем можно столкнуться при настройке сервера Samba, — это создание (или редактирование) файла конфигурации. Все файлы конфигурации Samba находятся в каталоге /etc/samba. Вот список файлов, которые обычно содержатся в этом каталоге:

- lmhosts — содержит список хостов и соответствующих им адресов;
- smbpasswd — содержит пароли пользователей сервера Samba;
- smbusers — файл, предназначенный для хранения списка пользователей, которым разрешен доступ к ресурсам Samba;
- smb.conf — главный конфигурационный файл сервера.

Примеры конфигурационных файлов, поставляемых с пакетом, находятся в каталоге /examples. В большинстве случаев их можно использовать в качестве базы.

В листинге 14.1 приведен файл smb.conf сервера Samba, который успешно функционирует на одном из серверов.

Листинг 14.1. Пример файла smb.conf

```
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options (perhaps too
# many!) most of which are not shown in this example
# Any line which starts with a ; (semi-colon) or a # (hash)
# is a comment and is ignored. In this example we will use a #
# for commentry and a ; for parts of the config file that you
# may wish to enable
# NOTE: Whenever you modify this file you should run the command
"testparm"
# to check that you have not many any basic syntactic errors.
#
#====================== Global Settings ==============================
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
   workgroup = Kontora
```

```
# server string is the equivalent of the NT Description field
   server string = Kontora Samba Server

# This option is important for security. It allows you to restrict
# connections to machines which are on your local network. The
# following example restricts access to two C class networks and
# the "loopback" interface. For more examples of the syntax see
# the smb.conf man page
hosts allow = 192.168.10. 193.166.17.
# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
#   printcap name = /etc/printcap
#   load printers = yes

# It should not be necessary to spell out the print system type unless
# yours is non-standard. Currently supported print systems include:
# bsd, sysv, plp, lprng, aix, hpux, qnx
   printing = lprng

# Uncomment this if you want a guest account, you must add this to
/etc/passwd
# otherwise the user "nobody" is used
;  guest account = pcguest

# this tells Samba to use a separate log file for each machine
# that connects
   log file = /var/log/samba/%m.log

# Put a capping on the size of the log files (in Kb).
   max log size = 0

# Security mode. Most people will want user level security. See
# security_level.txt for details.
   security = user
# Use password server option only with security = server or
# security = domain
;   password server = <NT-Server-Name>

# Password Level allows matching of _n_ characters of the password for
# all combinations of upper and lower case.
;  password level = 8
;  username level = 8

# You may wish to use password encryption.
# Please read ENCRYPTION.txt, Win95.txt and WinNT.txt in the Samba
# documentation.
```

```
# Do not enable this option unless you have read those documents
encrypt passwords = yes
smb passwd file = /etc/samba/smbpasswd

# The following are needed to allow password changing from Windows to
# update the Linux sytsem password also.
# NOTE: Use these with 'encrypt passwords' and 'smb passwd file' above.
# NOTE2: You do NOT need these to allow workstations to change only
#        the encrypted SMB passwords. They allow the Unix password
#        to be kept in sync with the SMB password.
;  unix password sync = Yes
;  passwd program = /usr/bin/passwd %u

# Unix users can map to different SMB User names
;  username map = /etc/samba/smbusers

# Using the following line enables you to customise your configuration
# on a per machine basis. The %m gets replaced with the netbios name
# of the machine that is connecting
;   include = /etc/samba/smb.conf.%m

# Most people will find that this option gives better performance.
# See speed.txt and the manual pages for details
   socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

# Configure Samba to use multiple interfaces
# If you have multiple network interfaces then you must list them
# here. See the man page for details.
interfaces = 192.168.10.0/24

# Configure remote browse list synchronisation here
#  request announcement to, or browse list sync from:
#  a specific host or from / to a whole subnet (see below)
;   remote browse sync = 192.168.3.25 192.168.5.255
# Cause this host to announce itself to local subnets here
;   remote announce = 192.168.1.255 192.168.2.44

# Browser Control Options:
# set local master to no if you don't want Samba to become a master
# browser on your network. Otherwise the normal election rules apply
;   local master = no

# OS Level determines the precedence of this server in master browser
# elections. The default value should be reasonable
;   os level = 33
```

```
# Domain Master specifies Samba to be the Domain Master Browser. This
# allows Samba to collate browse lists between subnets. Don't use this
# if you already have a Windows NT domain controller doing this job
;   domain master = yes

# Preferred Master causes Samba to force a local browser election on
#startup and gives it a slightly higher chance of winning the election
;   preferred master = yes

# Enable this if you want Samba to be a domain logon server for
# Windows95 workstations.
;   domain logons = yes

# if you enable domain logons then you may want a per-machine or
# per user logon script
# run a specific logon batch file per workstation (machine)
;   logon script = %m.bat
# run a specific logon batch file per username
;   logon script = %U.bat

# All NetBIOS names must be resolved to IP Addresses
# 'Name Resolve Order' allows the named resolution mechanism to be
# specified the default order is "host lmhosts wins bcast".
name resolve order = wins lmhosts bcast

# Windows Internet Name Serving Support Section:
# WINS Support - Tells the NMBD component of Samba to enable it's WINS
Server
wins support = yes

# WINS Server - Tells the NMBD components of Samba to be a WINS Client
#  Note: Samba can be either a WINS Server, or a WINS Client, but NOT
both
;   wins server = w.x.y.z

# WINS Proxy - Tells Samba to answer name resolution queries on
# behalf of a non WINS capable client, for this to work there must be
# at least one  WINS Server on the network. The default is NO.
;   wins proxy = yes

# DNS Proxy - tells Samba whether or not to try to resolve NetBIOS names
# via DNS nslookups. The built-in default for versions 1.9.17 is yes,
# this has been changed in version 1.9.18 to no.
   dns proxy = no

# Case Preservation can be handy - system default is _no_
# NOTE: These can be set on a per share basis
```

```
;  preserve case = no
;  short preserve case = no
# Default case is normally upper case for all DOS files
default case = lower
# Be very careful with case sensitivity — it can break things!
case sensitive = no

client code page = 866
character set = koi8-r
printer driver file=/home/samba/hplj1200/printers.def
#============================ Share Definitions ========================
[homes]
   comment = Home Directories
   browseable = no
   writable = yes
   valid users = yura tol katya slava vova lena alst

[comm]
        comment = Common place
        path = /home/samba/comm
        valid users = root slava tol yura katya vova lena alst
        public = no
        writable = yes
        printable = no
        create mask = 0775
        directory mask= 0775
        force group = office
[hp]
   comment = HP LaserJet 1200 Series PCL6
   path = /var/spool/samba
   printer = lp
   public = no
   printable = yes
   printer driver=HP LaserJet 1200 Series PCL6
   printer driver location=\\%h\printer$

[printer$]
  path=/home/samba/hplj1200
  public=yes
  browseable=yes

# This one is useful for people to share files
[tmp]
   comment = Temporary file space
```

```
path = /tmp
read only = no
public = yes
```

Как видно из примера, конфигурационный файл разбит на разделы. Каждый раздел начинается с заголовка, такого как `[global]`, `[homes]` и т. д. Структурой конфигурационный файл сильно напоминает INI-файлы операционной системы Windows. Символы `#` и `;` используются в качестве признаков комментария.

Секция *[global]*

Секция `[global]` назначает переменные, которые Samba будет использовать для определения доступа ко всем ресурсам. Рассмотрим переменные секции `[global]`.

- `workgroup = Kontora`

  Переменная `workgroup` содержит имя NT-домена или имя рабочей группы, к которой будет принадлежать сервер Samba.

- `netbios name = bw`

  Переменная `netbios name` задает имя сервера для отклика по протоколу NetBIOS. Не делайте его таким же, как и имя рабочей группы.

- `server string = Kontora Samba Server`

  Переменная `server string` содержит описание сервера (комментарий).

- `hosts allow = 192.168.10. 197.64.17.`

  Переменная `hosts allow` содержит список IP-адресов компьютеров и сетей, разделенных пробелом, которые имеют право подключаться к ресурсам вашего сервера Samba.

- `printing = lprng`

  Переменная `printing` определяет тип системы печати; поддерживается `bsd`, `sysv`, `plp`, `lprng`, `aix`, `hpux`, `qnx`.

- `guest account = pcguest`

  Переменная используется, если вы хотите разрешить гостевой вход на Samba-сервер. Соответствующего пользователя также придется завести в Linux-системе. Однако по соображениям безопасности не рекомендуется разрешать гостевой вход.

- `log file = /var/log/samba/%m.log`

  Переменная `log file` указывает серверу создавать LOG-файлы отдельно для каждого пользователя; заодно указывает каталог, где будут создаваться файлы.

- `max log size = 0`

  Переменная `max log size` определяет максимальный размер LOG-файла.

- `security = user`

  Переменная `security` используется для задания уровня безопасности системы; обычно применяется уровень `user`, также используют уровень `share`, `server` и `domain`.

- `password server = <NT-Server-Name>`

  Переменная `password server` используется только совместно с `security = server` или `security = domain`; задает имя сервера паролей.

- `password level` и `username level`

  Переменные `password level` и `username level` позволяют задать количество символов пароля и имени пользователя.

- `encrypt passwords = yes`

  Переменная `encrypt passwords` позволяет использовать пересылку паролей пользователей в зашифрованном виде; если задать `encrypt passwords = no`, то пароли пользователей будут пересылаться в незашифрованном виде, что очень плохо с точки зрения безопасности.

- `smb passwd file = /etc/samba/smbpasswd`

  Переменная `smb passwd file` задает путь и имя файла, содержащего пароли пользователей; поскольку принципы хранения пароля в Linux не позволяют его расшифровать, приходится создавать отдельный файл паролей для пользователей Samba.

- `local master = yes`

  Переменная `local master` позволяет серверу Samba стать мастер-браузером.

- `preferred master = yes`

  Переменная `preferred master` позволяет серверу Samba сразу же при запуске устроить перевыборы мастера с наибольшим шансом для себя.

Замечание

Протокол NetBIOS в принципе предназначен для одноранговой локальной сети, т. е. такой сети, где все компьютеры равноправны. Тем не менее в NetBIOS предусмотрен специальный компьютер, называемый *мастер* (master), который ведет список компьютеров, подключенных к сети, их разделяемые ресурсы и вновь подключаемые компьютеры. Именно от мастера вновь подключающиеся компьютеры получают список компьютеров в сети и их доступные ресурсы.

- `dns proxy = yes`

  Разрешает серверу сопоставлять NetBIOS-имена с IP-адресом при помощи DNS.

- `username map = /etc/samba/smbusers`

  Переменная `username map` позволяет задать файл пользователей Samba, в котором ставится соответствие имя Linux-пользователя имени Samba-пользователя; обычно в качестве имени пользователя Samba используется имя Linux-пользователя.

- `socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192`

  Переменная `socket options` используется для тонкой настройки сетевых параметров, позволяющих несколько улучшить производительность сервера.

- `interfaces = 192.168.10.0/24`

  Переменная `interfaces` указывает серверу, с какой сетевой картой (сетью) он имеет дело; используется в том случае, если на сервере установлено несколько сетевых карт из разных локальных сетей.

- `name resolve order = wins lmhosts bcast`

  Переменная `name resolve order` определяет порядок получения имен.

- `wins support = yes`

  Переменная `wins support` указывает, что сервер Samba выступает в роли WINS-сервера.

- `wins server = w.x.y.z`

  Переменная `wins server` определяет IP-адрес WINS-сервера; если установлено `wins support = yes`, то использование переменной `wins server` запрещено.

- `default case = lower`

  Переменная `default case` определяет регистр имен файлов, создаваемых на ресурсах Samba.

- `case sensitive = no`

  Переменная `case sensitive` определяет чувствительность к регистру символов.

- `client code page = 866`

  Переменная `client code page` задает кодовую страницу клиента; для DOS-клиента — `866`.

- `character set = koi8-r`

  Переменная `character set` задает набор символов, используемых сервером.

- `printer driver file=/home/samba/hplj1200/printers.def`

  Переменная `printer driver file` определяет имя драйвера принтера.

- `time server = true`

  Эта переменная предписывает серверу показывать клиентам Windows, что он выступает для них в роли сервера точного времени.

Секция *[homes]*

Секция `[homes]` позволяет удаленным пользователям получить доступ к своим домашним каталогам на Linux-машине. Для этого пользователь должен быть зарегистрирован в Linux-системе. Рассмотрим переменные секции `[homes]`.

- `comment = Home Directories`

  Эта переменная просто комментирует содержимое данной секции.

- `browseable = no`

  Переменная `browseable` запрещает просматривать каталог посторонним пользователям.

- `writable = yes`

  Переменная `writable` разрешает записывать в домашний каталог.

- `valid users = yura katya vova alst`

  Переменная `valid users` задает список пользователей, для которых разрешен доступ к своим домашним каталогам; в принципе параметр не обязательный.

Секция *[comm]*

Секция `[comm]` отвечает за каталог, доступный всем пользователям Samba. Это своего рода аналог FTP, куда могут записывать и откуда могут читать пользователи. Разберем подробнее данную секцию.

- `comment = Common place`

  Эта переменная просто комментирует содержимое данной секции.

- `path = /home/samba/comm`

  Переменная `path` определяет каталог, который используется для совместного доступа.

- `valid users = root yura katya vova alst`

  Переменная `valid users` содержит список пользователей, которым разрешен доступ к общему ресурсу.

- `public = no`

  Запрещает остальным пользователям получать доступ к данному ресурсу.

- `writable = yes`

  Разрешает запись в общий ресурс.

- `printable = no`

  Указывает, что разделяемый ресурс не является печатающим устройством.

- `create mask = 0775`

  Маска для создания файлов на разделяемом ресурсе.

- `directory mask= 0775`

  Маска для создания каталогов на разделяемом ресурсе.

- `force group = office`

  Переменная `force group` определяет, что файлу, создаваемому или копируемому на общий ресурс, принудительно задается принадлежность к группе `office`, для того чтобы любой пользователь мог изменить или удалить файл.

Секция *[tmp]*

Секция `[tmp]` предназначена для создания разделяемого ресурса, в который могли бы записывать все пользователи. Как видно из приведенного далее описания, она отличается от секции `[comm]` отсутствием списка пользователей и значением переменной `public`.

```
comment = Temporary file space
path = /tmp
read only = no
public = yes
```

Пароли пользователей

Сервер Samba подразумевает использование нескольких типов безопасности. В частности, переменная `encrypt password` определяет, какой механизм авторизации будет использован. Если переменной `encrypt password` присвоено значение `no`, то авторизация пользователей производится исходя из учетных записей Linux, хранящихся в файлах /etc/passwd и /etc/shadow. При таком типе авторизации пользователя пароли передаются по сети в незашифрованном виде, что несколько упрощает настройку, но резко снижает безопасность системы. В дополнение к этому, такой тип авторизации требует изменений в системном реестре в Windows 95, Windows 98, Windows NT. Далее приведены изменения, которые необходимо внести в системный реестр.

- Windows 95

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000001
```

- Windows 98

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\VNETSUP]
"EnablePlainTextPassword"=dword:00000001
```

- Windows NT

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Rdr\Parameters]
"EnablePlainTextPassword"=dword:00000001
```

- Windows 2000

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services
  \LanmanWorkStation\Parameters]
"EnablePlainTextPassword”=Data: 0x01
```

В том случае, если переменной `encrypt password` присвоено значение `yes`, авторизация пользователя происходит с использованием файла /etc/samba /smbpasswd, а передача паролей — в зашифрованном виде.

Почему для использования шифрованных паролей необходимо создавать отдельную базу паролей пользователей Samba? Все дело в методе хранения пароля. Windows-системы хранят зашифрованный пароль, и при аутентификации пользователя производят сверку паролей. Linux *не хранит* пароль как таковой. В файле shadow хранится так называемый хэш (hash) пароля, а в последних версиях Linux — контрольная сумма пароля, рассчитанная по алгоритму MD5. И при аутентификации пользователя сравниваются хэши паролей. Особенность хэша — он необратим, т. е., зная хэш, невозможно по нему восстановить пароль. Поэтому приходится отдельно для Samba заводить базу паролей пользователей. Для администратора системы это представляет некоторое неудобство — еще один повод забыть прописать пользователя, а с другой стороны — за все надо платить.

Добавление пользователей Samba

Для добавления пользователей в файл /etc/samba/smbpasswd необходимо наличие самого файла /etc/samba/smbpasswd. Должна также существовать учетная запись пользователя в Linux-системе. Если эти условия соблюдены, следует:

1. Воспользоваться программой smbpasswd для создания учетной записи `smbpasswd -a user_name`;
2. Активировать учетную запись `smbpasswd -e user_name`.

Эту операцию придется произвести с каждым пользователем. Существуют скрипты, позволяющие перебросить пользователей из файла passwd в файл smbpasswd. Но они только перебрасывают пользователей, а пароли для них все равно придется заводить вручную. Еще один недостаток этих скриптов — после них придется удалять пользователей типа nobody, root, news и т. п.

Для монтирования ресурсов, предоставляемых сервером Samba, используются команды `smbclient` и `smbmount`. Обо всех возможностях этих команд можно узнать из соответствующих man-страниц, краткие сведения о команде `smbclient` вы получите в этой главе.

Принтеры

Принтер, установленный в системе с сервером Samba, предоставить в общее пользование Samba-клиентам очень просто. Все принтеры, которые определены в файле /etc/printcap, становятся доступными после того, как вы добавите следующую секцию в конфигурационный файл smb.conf:

```
[printers]
path = /var/spool/lpd
writeable = no
guest ok = no
printable = yes
```

Использование ресурсов Samba

Хотя сервер Samba позиционируется как средство доступа Windows-клиентов к ресурсам Linux-систем, тем не менее, в пакете есть средства для того, чтобы Linux-компьютеры могли также просматривать и монтировать SMB-ресурсы. И что особенно приятно, доступ к ресурсам Windows-сети можно получить и в том случае, когда сервером является машина с Windows!

Программа клиента SMB для Linux включена в дистрибутив Samba и называется smbclient. Она обеспечивает FTP-подобный интерфейс командной строки. Также существует пакет smbfs, который позволяет монтировать и размонтировать SMB-ресурсы.

Для того чтобы увидеть доступные SMB-ресурсы, выполните команду:

```
/usr/bin/smbclient -L host
```

где `host` — это имя машины, ресурсы которой вас интересуют. Данная команда вернет список имен доступных сервисов.

Пример команды smbclient:

```
  smbclient -L ziga
Server time is Sat Aug 19 19:58:27 1999
Timezone is UTC+2.0
Password:
Domain=[WORKGROUP] OS=[Windows NT 3.51] Server=[NT LAN Manager 3.51]

Server=[ZIGA] User=[] Workgroup=[WORKGROUP] Domain=[]

        Sharename      Type      Comment
        ---------      ----      -------
        ADMIN$         Disk      Remote Admin
        public         Disk      Public
        C$             Disk      Default share
        HP             Printer   HP6L

This machine has a browse list:

        Server               Comment
        ---------            -------
        HOP                  Samba 1.9.15p8
        ZIGA
```

Для использования сервиса, выполните следующую команду:

```
/usr/bin/smbclient service <password>
```

где service — имя хоста и сервиса. Например, если вы пытаетесь обратиться к каталогу, который доступен под именем public на машине, названной ziga, то имя сервиса должно представлять собой \\ziga\public. Поскольку в языке C обратный слэш является спецсимволом, то практически необходимо ввести такую строку:

```
/usr/bin/smbclient \\\\ziga\\public <mypasswd>
```

где <mypasswd> — ваш пароль.

В результате вы должны получить приглашение smbclient:

```
smb: \>
```

Для получения справки необходимо ввести h и нажать клавишу <Enter>:

```
smb: \> h
ls             dir            lcd            cd             pwd
get            mget           put            mput           rename
more           mask           del            rm             mkdir
md             rmdir          rd             prompt         recurse
```

```
translate      lowercase      print          printmode      queue
cancel         stat           quit           q              exit
newer          archive        tar            blocksize      tarmode
setmode        help           ?              !
smb: \>
```

Как видите, практически все команды дублируют команды FTP-клиента.

Утилита smbclient многое позволяет, однако она утомительна для использования. Если от Windows-сети нужен только доступ к дисковым ресурсам, рекомендуется воспользоваться пакетом Smbfs.

В пакет Smbfs входят утилиты smbmount и smbumount, которые работают подобно mount и umount. Также есть графическая утилита gnomba — подобная утилите **Сеть** Windows.

Конфигурирование Samba в качестве первичного контроллера домена

Ранее мы рассмотрели основные моменты конфигурирования сервера в качестве простого сервера, предоставляющего в пользование свои ресурсы. В этом разделе мы будем конфигурировать Samba таким образом, чтобы сервер выступал в качестве первичного контроллера домена сети Windows.

Конфигурирование Samba в качестве контроллера домена можно разделить на два больших шага:

- настройка Samba PDC;
- создание доверенных бюджетов машин и подключение клиентов к домену.

Есть несколько моментов, на которые следует обратить ваше внимание:

- необходимо включить шифрование паролей;
- сервер должен поддерживать `domain logons` и ресурс `[netlogon]`;
- для того чтобы клиенты Windows корректно определяли сервер как контроллер домена, сервер должен быть главным обозревателем сети (domain master browser).

Samba 2.2 не предоставляет полную реализацию отображения групп между группами Windows NT и UNIX (это достаточно сложно коротко объяснить), для информации о создании пользовательских бюджетов в стиле Domain Admins вы должны свериться с параметром `domain admin users` файла smb.conf.

В листинге 14.2 приведена часть файла конфигурации smb.conf, в которой прописаны параметры, позволяющие стать контроллером домена серверу Samba.

Листинг 14.2. Пример файла smb.conf

```
[global]
    ; основные настройки сервера
    netbios name = domain_pdc
    workgroup = test

    ; сервер должен выступать в роли domain и local master browser
    os level = 64
    preferred master = yes
    domain master = yes
    local master = yes

   ; название сервера-контроллера домена
    password server = domain_pdc
    ; разрешить работу с доверенными доменами
        allow trusted domains = yes
        ; поддержка правил доступа и ограничений в стиле NT
        nt acl support = yes

    ; настройки безопасности
    security = user

    ; для PDC требуется шифрование паролей
    encrypt passwords = yes

    ; поддержка domain logons
    domain logons = yes

    ; место, где помещать профили пользователей
    logon path = \\%N\profiles\%u
    ; местонахождение домашних каталогов пользователей
    ; и где они должны быть смонтированы
    logon drive = H:
    logon home = \\homeserver\%u

    ; указываем общий скрипт подключения для всех пользователей
    ; это относительный путь к [netlogon] ресурсу
    logon script = logon.cmd

; необходимый ресурс для контроллера домена
[netlogon]
    path = /usr/local/samba/lib/netlogon
    writeable = no
    write list = ntadmin
```

```
; ресурс для размещения профилей пользователей
[profiles]
    path = /export/smb/ntprofile
    writeable = yes
    create mask = 0600
    directory mask = 0700
```

Утилиты

Как и у других подобных проектов, для пакета Samba существует достаточно много сторонних утилит, позволяющих упростить конфигурирование и доступ к ресурсам.

Вот список утилит и программ, в той или иной мере относящихся к пакету Samba:

- smbstatus — утилита для мониторинга Samba;
- SWAT — инструмент для конфигурирования Samba через Web-интерфейс;
- smbpasswd — управление паролями Samba;
- testparm — проверка конфигурационного файла;
- testprns — проверка конфигурации принтера;
- smbtar — SMB-утилита резервного копирования;
- smbclient — клиент командной строки;
- Ksamba — KDE-программа, предназначенная для конфигурации;
- Smbedit — Win32-приложение для правки конфигурационного файла Samba;
- Webmin — универсальная программа конфигурации через Web-интерфейс, в том числе и Samba;
- GSMB — графический интерфейс для утилиты smbpasswd;
- SambaSentinel — графический интерфейс для утилиты smbstatus.

SWAT

SWAT (Samba Web Administration Tool) — одна из наиболее известных утилит для работы с сервером Samba через Web-интерфейс (рис. 14.1). Для доступа к SWAT в браузере необходимо набрать `localhost:901`. Далее, после ввода логина и пароля вы получаете доступ к программе SWAT.

SWAT охватывает практически все настройки Samba, которые доступны администратору через Web-интерфейс.

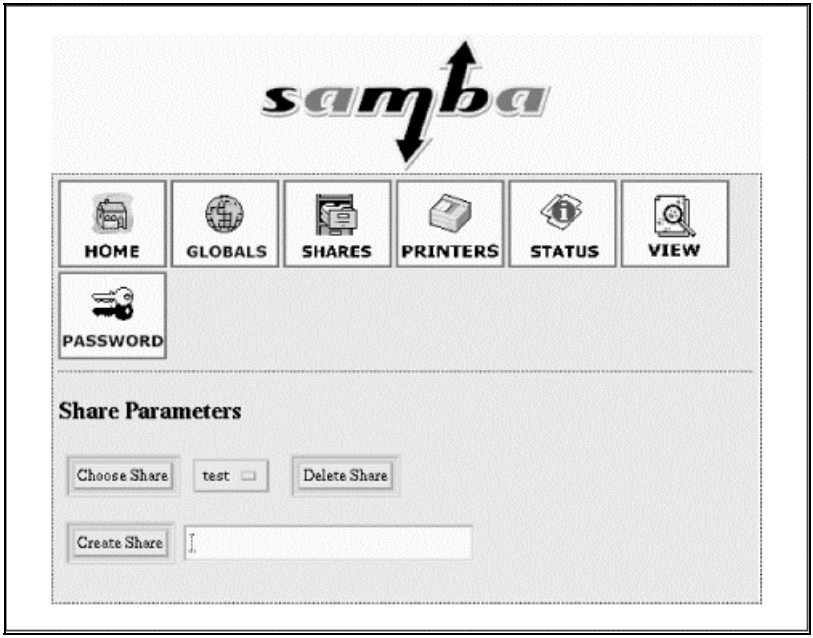


Рис 14.1. SWAT — страница глобальных переменных

Webmin

Webmin — программа с Web-интерфейсом, позволяющая конфигурировать множество служб и сервисов через Web (рис. 14.2). В частности, есть возможность настраивать сервер Samba.

Ksamba

Ksamba — программа для KDE-оболочки, предназначенная для конфигурации Samba. Достаточно удобная и понятная.

SambaSentinel

Графический интерфейс к утилите smbstatus. Позволяет производить мониторинг, удалять зависшие задачи и т. п.

Литература и ссылки

- **boombox.campus.luth.se/sambasentinel.php** — сайт проекта sambasentinel.
- **www.culte.org/projets/developpement/gsmb/** — официальный сайт проекта GSMB.

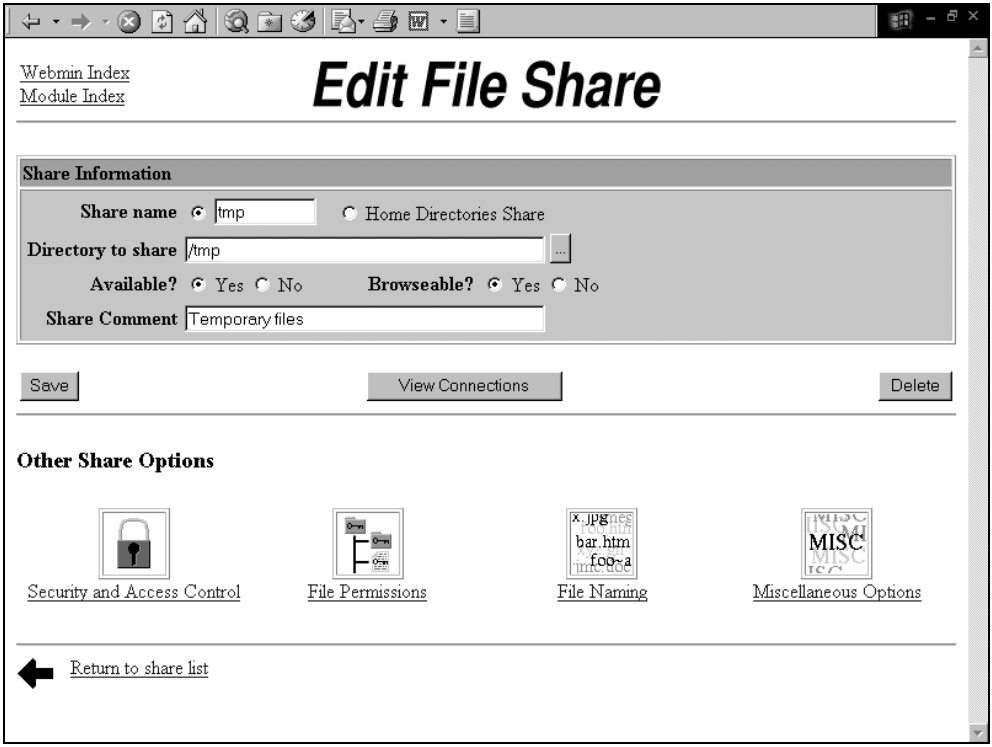
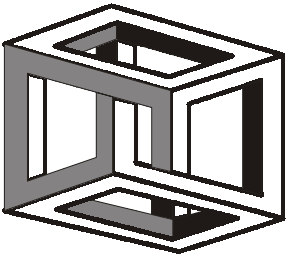


Рис. 14.2. Webmin — управление Samba

- **www.linuxcenter.ru/lib/soft/samba_pdc.phtml** — Как настроить Samba 2.2 в качестве основного контроллера домена (Primary Domain Controller, PDC).
- **www.linuxoid.ru/how_to/samba5.html** — Басин И. Samba за пять минут.
- **www.linux.org.ru/books/HOWTO/SMB-HOWTO.html** — SMB-HOWTO (русский перевод).
- **www.samba.org** — официальный сайт проекта Samba.
- **www.webmin.com** — официальный сайт проекта Webmin.

ГЛАВА 15



Mars — клиентам Novell

В последние несколько лет протокол TCP/IP стал стандартом де-факто при построении сетей, однако еще лет пять назад в корпоративной среде в качестве сетевого протокола повсеместно использовался протокол IPX, разработанный фирмой Novell.

Фирма Novell стояла у истоков локальной сети как таковой. До сих пор на большинстве сетевых карт написано "Ready for Novell" или "NE compatible" — что означает совместимость с сетевой картой производства фирмы Novell (которые не производятся уже лет 15). Флагманский продукт фирмы Novell — программное обеспечение для сервера, носящее название "Novell NetWare". Это программное обеспечение на просторах СНГ долгое время занимало монопольное положение при построении локальных сетей на базе недорогих компьютеров *x*86.

Наиболее популярным и чаще устанавливаемым программным обеспечением была Novell NetWare 3.*x*. Конечно, были выпущены Novell NetWare версии 4 и 5, однако из-за просчетов компании и других факторов (в частности увеличения популярности Windows NT и Linux) рыночная доля Novell NetWare стремительно сократилась. Тем не менее, во многих крупных учреждениях сохранились серверы, использующие Novell NetWare. Относительно недавно компания Novell заявила, что новая версия Novell NetWare будет функционировать на ядре Linux и даже прикупила производителя одного из дистрибутивов Linux. Но поскольку осталось еще много старых серверов Novell NetWare, мы должны уметь с ними взаимодействовать.

Термины, используемые в тексте

Для лучшего понимания текста приведем расшифровку некоторых используемых терминов. Поскольку идеология IPX отличается от TCP/IP, отдельные термины могут оказаться незнакомыми для широкой аудитории.

- 802.2 — это протокол IEEE, определяющий набор процедур управления логическими связями (Logical Link Control).
- 802.3 — это протокол IEEE, определяющий механизм множественного доступа к среде переноса с определением коллизий (Carrier Sense Multiple Access with Collision Detection, CSMA/CD). Базируется на оригинальном стандарте DIX Ethernet с некоторыми дополнениями. Важнейшее изменение — в кадре поле типа "идентификатор протокола" используется в качестве поля длины. IEEE 802.3 был спроектирован для того, чтобы переносить *только* фреймы IEEE 802.2, однако существуют реализации, которые используют этот тип для прямого переноса фреймов IPX.
- Bindery — специализированная база данных, сохраняющая сетевую конфигурационную информацию на файловом сервере Novell. Используется для получения информации о доступных серверах, маршрутизации и пользователях.
- Ethernet_II — это упрощенная версия оригинального стандарта DIX Ethernet. Протокол часто используется в среде Novell NetWare.
- Hardware address — адрес устройства. Число, уникально идентифицирующее хост в физической сети на уровне доступа к среде передачи данных.
- IPX (Internet Packet eXchange, межсетевой обмен пакетами) — протокол, используемый корпорацией Novell для обеспечения межсетевой поддержки NetWare.
- Сеть IPX — это набор оборудования, подключенного к одному и тому же сегменту локальной сети, и использующий один и тот же тип фрейма. Различные типы фреймов в одном и том же сегменте локальной сети считаются отдельными сетями. Каждой сети выделяется адрес, который должен быть уникальным во всей локальной сети. Клиентам IPX этот адрес выдается сервером при запуске.
- Сетевой адрес IPX — уникальное число, которое идентифицирует частную сеть IPX.
- Внутренняя сеть IPX — это виртуальная сеть IPX. Используется для обеспечения уникальной идентификации хоста IPX. Применяется для хостов IPX, которые существуют больше чем в одной физической сети IPX (файловые серверы, маршрутизаторы).
- NCP (NetWare Core Protocol, базовый протокол NetWare) — протокол сетевой файловой системы Novell NetWare.
- RIP (Routing Information Protocol, протокол маршрутной информации) — протокол, используемый для автоматического распространения сетевых маршрутов в сетях IPX.
- Route (маршрут) — путь прохождения пакета для достижения хоста назначения.

- SAP (Service Advertisement Protocol, протокол объявления сервисов) — протокол, который используется для объявления сетевых сервисов в Novell NetWare.
- SNAP (Sub Network Access Protocol, протокол доступа к подсетям) — спроектирован для использования поверх протоколов 802.3 и 802.2.

Linux и IPX

В этом разделе мы рассмотрим три варианта настройки Linux-системы:

- IPX-клиент;
- IPX-сервер;
- IPX-маршрутизатор.

Прежде чем приступить к настройке системы, необходимо убедиться, что ваше ядро операционной системы Linux скомпилировано с поддержкой протокола IPX.

Файлы в каталоге /proc, относящиеся к IPX

Существует несколько файлов, тем или иным образом касающихся поддержки IPX в Linux, которые располагаются в каталоге /proc.

- /proc/net/ipx_interface — этот файл содержит информацию о существующих интерфейсах IPX в вашей системе;
- /proc/net/ipx_route — этот файл содержит список маршрутов, существующих в таблице маршрутов IPX;
- /proc/net/ipx — этот файл содержит список сокетов IPX, которые открыты для использования на вашем компьютере.

Linux-утилиты IPX

Помимо пакета Mars_nwe, есть несколько утилит, позволяющих сконфигурировать поддержку IPX-протокола в операционной системе Linux.

- `ipx_interface` — эта команда используется для добавления, удаления или проверки IPX на существующем сетевом устройстве. Обычно сетевым устройством является устройство Ethernet. Например:

  ```
  ipx_interface add -p eth0 802.2 x39ab0222
  ```

- `ipx_configure` — эта команда разрешает или запрещает автоматическую установку конфигурации интерфейсов и первичного интерфейса. Например:

  ```
  ipx_configure --auto_interface=on --auto_primary=on
  ```

- `ipx_internal_net` — эта команда позволяет настраивать адрес внутренней сети. Например:

  ```
  ipx_internal_net add 0xab000000 1
  ```

- `ipx_route` — эта команда позволяет вручную модифицировать таблицу маршрутизации IPX. Например:

  ```
  ipx_route add 0x39ab0222 0x39ab0108 0x00608CC33C0F
  ```

IPX-клиент

Существует пакет ncpfs, который позволяет Linux эмулировать обычную рабочую станцию Novell для файловых сервисов. В этот пакет также входит утилита печати, которая позволяет использовать принт-сервер Novell. Пакет ncpfs предназначен для работы с файловыми серверами Novell 3.*x*. Для использования ncpfs с файловыми серверами Novell 4.*x* файловый сервер должен работать в режиме эмуляции Bindery.

Настройка сетевого программного обеспечения IPX

Существует два способа настройки сетевого программного обеспечения IPX:

- вы можете вручную настроить всю информацию о вашей сети IPX;
- можно позволить программному обеспечению определить для себя установки с помощью команды:

  ```
  ipx_configure --auto_interface=on --auto_primary=on
  ```

Проверка конфигурации

После настройки вашей сети IPX воспользуйтесь командой `slist`, для того чтобы увидеть список всех файловых серверов Novell в вашей сети.

Монтирование сервера или тома Novell

Для того чтобы смонтировать том файлового сервера Novell в файловую систему Linux, существует команда `ncpmount`. Для демонтирования смонтированных файловых систем Nowell используется команда `ncpumount`.

Посылка сообщения пользователю Novell

Для посылки сообщений пользователям Novell можно воспользоваться утилитой nsend. Пример:

```
nsend petr hello
```

Эта команда посылает сообщение `hello` пользователю, вошедшему под именем `petr`.

IPX-сервер

Существует, по меньшей мере, два пакета, которые позволяют операционной системе Linux выступать в качестве файлового сервера Novell — это Mars_nwe и Lwared. Эти пакеты позволяют осуществлять доступ к файлам на Linux-системе для пользователей, использующих клиентское программное обеспечение Novell NetWare.

Пакет Mars_nwe

Данный пакет разработан Martin Stover для обеспечения в операционной системе Linux работы файловых сервисов и сервисов печати для клиентов NetWare. Mars_nwe реализует подмножество полного Novell NCP для файловых сервисов, основанного на Bindery и сервисах печати.

Настройка сервера

Конфигурационный файл называется nwserv.conf и находится в каталоге /etc. Файл состоит из текстовых строк. Каждая строка разделена пробелами и начинается с числа, которое обозначает содержимое этой строки. Все символы, следующие за символом '#', считаются комментарием и игнорируются. В комплекте Mars_new есть пример конфигурационного файла (листинг 15.1).

Листинг 15.1. Пример конфигурационного файла

```
# Только том SYS является необходимым. Каталог, содержащий том SYS,
# должен содержать каталоги: LOGIN, PUBLIC, SYSTEM, MAIL.
# Опция 'i' регистр букв.
# Опция 'k' преобразует все имена в запросе NCP в нижний регистр
# Опция 'm' обозначает том как сменный
# Опция 'r' устанавливает том только для чтения
# Опция 'o' показывает, что том является единой файловой системой
# Опция 'P' разрешает командам использоваться как файлы
# Опция 'O' позволяет использовать пространство имен OS/2
# Опция 'N' разрешает использование пространства имен NFS
# По умолчанию в верхнем регистре.
# Синтаксис:
#    1 <Имя тома>  <Путь к тому>        <Опции>

1   SYS         /home/NetWare/SYS/                # SYS
1   DATA        /home/NetWare/DATA/      k        # DATA
1   CDROM       /cdrom                   kmr      # CDROM

# ИМЯ СЕРВЕРА
# если не настроено — будет использовано имя хоста.
```

```
# Синтаксис:
#    2 <Имя сервера>

2   LINUX_MARS

# АДРЕС ВНУТРЕННЕЙ СЕТИ
# Адрес внутренней сети IPX — это свойство, которое упрощает
# маршрутизацию IPX для многосетевых машин
# Синтаксис:
#    3 <Адрес внутренней сети> [<Номер узла>]
# или:
#    3 auto
# Если вы используете 'auto', тогда будет использован IP-адрес хоста.
3   0x49a01010  1

# СЕТЕВОЕ УСТРОЙСТВО(А)
# Этот раздел настраивает вашу сеть IPX. Если она у вас уже
# настроена, вам этот пункт не нужен. Это то же самое что и
# использование утилит ipx_configure/ipx_interface до запуска
# сервера.
# Синтаксис:
#    4 <Номер сети IPX> <имя устройства> <тип фрейма> [<ticks>]
#                          Frame types: ethernet_ii, 802.2, 802.3, SNAP

4  0x39a01010  eth0  802.3  1

# СОХРАНЯТЬ МАРШРУТЫ IPX ПОСЛЕ ОКОНЧАНИЯ РАБОТЫ СЕРВЕРА
# Синтаксис:
#    5 <флаг>
#       0 = не сохранять маршруты, 1 = сохранять маршруты

5 0

# ВЕРСИЯ NETWARE
# Синтаксис:
#    6 <версия>
#      0 = 2.15, 1 = 3.11

6 1

# ОБРАБОТКА ПАРОЛЯ
# Настоящие клиенты Novell для DOS поддерживают процедуру, которая
# шифрует пароли при их изменении. Вы можете выбрать, хотите ли вы,
# чтобы ваш сервер поддерживал эту процедуру или нет.
# Синтаксис:
#    7 <флаг>
#    <флаг> может быть:
#        0 force password encryption. (Клиенты не могут сменить пароль)
```

```
#        1 force password encryption, разрешить изменение нешифрованного
#        пароля
#        7 разрешаются нешифрованные пароли, но не пустые
#        8 разрешаются нешифрованные пароли, включая пустые
#        9 полностью нешифрованные пароли (не работает с OS/2)

7 1

# МИНИМАЛЬНЫЕ ПРАВА GID UID
# Синтаксис:
#    10 <gid>
#    11 <uid>
#    <gid> <uid> из /etc/passwd, /etc/groups

10  200
11  201

# ПАРОЛЬ АДМИНИСТРАТОРА (SUPERVISOR)
# Может быть убран после первого запуска сервера. Сервер зашифрует
# эту информацию в файл bindery после запуска. Вы должны избегать
# использования пользователя 'root' и вместо этого использовать
# другой идентификатор для администрирования файлового сервера mars
#
# Синтаксис:
#    12 <Идентификатор администратора> <имя пользователя UNIX> [<пароль>]

12  SUPERVISOR  terry  secret

# ЗАПИСИ ПОЛЬЗОВАТЕЛЕЙ
# Этот раздел ассоциирует идентификаторы NetWare с идентификаторами
# пользователей UNIX. Наличие пароля является опциональным.
# Синтаксис:
#    13 <Идентификатор пользователя> <имя пользователя в Unix> [<пароль>]

13  MARTIN martin
13  TERRY  terry

# НАСТРОЙКА СИСТЕМЫ "ЛЕНИВОГО" АДМИНИСТРИРОВАНИЯ
# Если у вас большое количество пользователей, и вы не хотите
# беспокоиться использованием индивидуального сопоставления
# пользовательских имен  между Linux_системой
# и Mars_nwe, то вы можете
# автоматически сопоставить идентификаторы Mars_nwe в имена
# пользователей Linux. Но в настоящее время нет способа использовать
# пароли Linux, так что все пользователи, настроенные таким способом,
# будут пользоваться единственным паролем, указанным здесь.
```

```
# Синтаксис:
#    15 <флаг> <общий пароль>
#    <флаг>:    0  — не делать автоматическое мапирование пользователей
#               1  — автоматически мапировать пользователей, не указанных
#                    выше
#               99 — автоматически мапировать всех пользователей

15  0  duzzenmatta

# ПРОВЕРКА РАБОТОСПОСОБНОСТИ
# Mars_nwe будет автоматически убеждаться, что определенные
# каталоги существуют, если установлен этот флаг
# Синтаксис:
#    16 <флаг>
#    <флаг> — 0 для нет, не делать, или 1 для да, делать проверку

16  0

# ОЧЕРЕДИ ПЕЧАТИ
# Этот раздел ассоциирует принтер NetWare с принтерами UNIX.
# Каталоги очередей должны быть созданы вручную до попытки печати.
# Каталоги очередей НЕ являются очередями lpd.
# Синтаксис:
#    21 <имя очереди> <каталог очереди> <команда печати UNIX>

21  EPSON  SYS:/PRINT/EPSON lpr -h
21  LASER  SYS:/PRINT/LASER lpr -Plaser

# ФЛАГИ ОТЛАДКИ
# Обычно они не нужны, но могут быть полезными, если вы ищите решение
# проблемы.
# Синтаксис:
#    <тема отладки> <флаг отладки>
#
#    100 = IPX KERNEL
#    101 = NWSERV
#    102 = NCPSERV
#    103 = NWCONN
#    104 = start NWCLIENT
#    105 = NWBIND
#    106 = NWROUTED
#                 0 = запрещает отладку, 1 = разрешает отладку

100 0
101 0
102 0
```

```
103 0
104 0
105 0
106 0

# ЗАПУСК NWSERV В ФОНОВОМ РЕЖИМЕ И ИСПОЛЬЗОВАНИЕ ФАЙЛА ПРОТОКОЛА
# Синтаксис:
#    200 <флаг>
#   0 = запуск NWSERV в нормальном режиме, не использовать файл протокола
#   1 = запуск NWSERV в фоновом режиме и использовать файл протокола

200  1

# ИМЯ ФАЙЛА ПРОТОКОЛА
# Синтаксис:
#    201 <файл протокола>

201  /tmp/nw.log

# ДОПОЛНЯТЬ ПРОТОКОЛ ИЛИ ПЕРЕЗАПИСЫВАТЬ
# Синтаксис:
#    202 <флаг>
#        0 = добавлять к существующему файлу протокола
#        1 = переписывать существующий файл протокола

202  1

# ВРЕМЯ ВЫКЛЮЧЕНИЯ СЕРВЕРА
# Этот раздел устанавливает время между выдачей команды SERVER DOWN
# и действительным выключением сервера.
# Синтаксис:
#    210 <время>
#        в секундах. (по умолчанию 10)

210  10

# ИНТЕРВАЛ МЕЖДУ  ШИРОКОВЕЩАТЕЛЬНЫМИ ПЕРЕДАЧАМИ МАРШРУТОВ
# Время в секундах между широковещательными передачами сервера.
# Синтаксис:
#    211 <время>
#        в секундах. (по умолчанию 60)

211  60

# ИНТЕРВАЛ ПРОТОКОЛИРОВАНИЯ МАРШРУТИЗАЦИИ
# Устанавливает, сколько широковещательных передач произойдет
# до протоколирования маршрутизационной информации.
```

```
# Синтаксис:
#    300  <число>

300  5

# ФАЙЛ ПРОТОКОЛА МАРШРУТИЗАЦИИ
# Устанавливает имя файла протокола маршрутизации
# Синтаксис:
#    301 <имя файла>

301  /tmp/nw.routes

# ДОБАВЛЕНИЕ/ПЕРЕЗАПИСЬ МАРШРУТНОЙ ИНФОРМАЦИИ
# Устанавливает, хотите ли вы добавлять информацию к существующему
# файлу протокола или перезаписывать его.
# Синтаксис:
#    302 <флаг>
#        <flag> — 0 для дополнения, 1 для создания/перезаписи

302  1

# WATCHDOG TIMING
# Устанавливает хронометраж для наблюдательных сообщений, чтобы
# убедиться, что сеть функционирует.
# Синтаксис:
#    310 <значение>
# = 0 — всегда посылать наблюдательные сообщения
# < 0 — (-ve) для запрета наблюдений
# > 0 — посылать наблюдательные сообщения при падении трафика < 'n' ticks

310  7

# ФАЙЛ СТАНЦИЙ
# Устанавливает имя для файла станций, который определяет, для каких
# машин этот файловый сервер будет выступать как первичный файловый
# сервер. Синтаксис этого файла описан в каталоге 'examples' исходного
# кода пакета.
# Синтаксис:
#    400 <имя файла>

400  /etc/nwserv.stations

# ОБРАБОТКА 'GET NEAREST FILESERVER'
# Устанавливает, как будет обрабатываться запрос SAP 'Get Nearest
# Fileserver' (получить ближайший файловый сервер).
```

```
# Синтаксис:
#    401 <флаг>
#        <флаг>: 0 — запретить запросы 'Get Nearest Fileserver'.
#                1 — файл 'stations' перечисляет исключаемые станции.
#                2 — файл 'stations' перечисляет включаемые станции.

401  2
```

Для запуска сервера достаточно выполнить команду:

```
nwserv
```

Пакет Lwared

Этот пакет разработан Ales Dryak. Он позволяет системе Linux функционировать в качестве файлового сервера Novell.

Сервер Lwared обеспечивает подмножество всех функций Novell NCP. Он включает функции сообщений, но не обеспечивает возможности печати. Сервер Lwared полагается на внешние программы для выполнения функций построения и обновления таблиц маршрутизации IPX и SAP.

Настройка и использование Lwared

Сначала необходимо настроить интерфейсы Ethernet для поддержки сетей IPX, которые будет использовать ваш сервер. Для того чтобы сделать это, необходимо знать сетевые адреса IPX для каждого из сегментов локальной вычислительной сети (ЛВС), какие устройства Ethernet находятся в системе, какой тип фреймов (`802.3`, `EtherII`) применяет каждый сегмент ЛВС и какой адрес внутренней сети должен использовать ваш сервер. Настройка для сервера, который находится в двух непохожих сегментах с сетевыми адресами IPX, равными `23a9c300` и `23a9c301`, и адресом внутренней сети `bdefaced` может выглядеть так:

```
ipx_internal_net add BDEFACED 1
ipx_interface add eth0 802.3 23a9c300
ipx_interface add eth1 etherii 23a9c301
```

Для управления таблицей маршрутизации используются два демона, входящие в комплект Lwared:

- ipxripd — управляет маршрутизационной информацией IPX;
- ipxsapd — управляет информацией SAP.

Для конфигурирования сервера Lwared необходимо сконфигурировать нижеприведенные файлы.

- /etc/lwpasswd — в этом файле хранится информация о пользователях сервера Lwared. Для работы с записями в этом файле используется программа lwpasswd.

Файл /etc/lwpasswd содержит текстовые строки, каждая из них идентифицирует пользователя и его пароль в зашифрованном виде. Отсутствие шифрованного пароля разрешает вход без пароля. Пользователи Lwared должны быть также зарегистрированы в операционной системе Linux.

- /etc/lwvtab — этот файл содержит таблицу томов Lwared и хранит информацию о доступных для сетевых клиентов каталогов сервера.

  Формат файла очень прост — после имени тома через пробел следует экспортируемый каталог Linux. Вы должны иметь, по крайней мере, запись для тома SYS, чтобы запустить сервер.

Для запуска сервера Lwared достаточно выполнить следующую команду:

```
lwared
```

IPX-маршрутизатор

Маршрутизатор используется для того, чтобы пересылать информацию из одной локальной сети в другую. Для сети на базе Novell NetWare есть два вида информации, которые необходимо распространять по сети для ее нормального функционирования. Это информация о сетевых маршрутах, использующая Novell RIP, и информация о сервисах, применяющая Novell SAP. Поэтому маршрутизатор должен поддерживать оба этих протокола.

Для нормального функционирования IPX-маршрутизатора Linux необходимы программы, реализующие Novell RIP и SAP, обеспечивающие правильность построения таблицы маршрутизации IPX и ее периодическое обновление для отражения изменений в сети.

Существует, по крайней мере, два способа создать IPX-маршрутизатор:

- можно использовать демон маршрутизации ipxripd;
- в состав пакета Mars_nwe входит свой демон маршрутизации.

Для настройки системы в качестве IPX-маршрутизатора необходимо выполнить следующие условия:

- ядро должно быть скомпилировано с поддержкой IPX и Ethernet;
- необходимо установить программу ipxd;
- включить протокол IPX на каждом сетевом интерфейсе, используя команду `ipx_interface`;
- запустить программу демона ipxd.

Пример:

```
# ipx_interface add eth0 802.2 0x0100000000
# ipx_interface add eth1 etherii 0x0300000000
# ipxd
```

Для проверки работоспособности маршрутизации посмотрите файл /proc /net/ipx_route. В этом файле вы должны увидеть маршруты IPX, относящиеся к вашей конфигурации.

Настройка Linux как клиента печати сервера Novell

Пакет ncpfs содержит две программы, которые производят печать из Linux-системы на принтер, подключенный к серверу печати Novell. Команда `nprint` ставит файл в очередь печати NetWare. Команда `pqlist` выводит список доступных очередей печати на сервере NetWare.

Обе команды требуют указать имя пользователя и пароль.

Пример:

```
pqlist -S ACCT_FS01 -U guest -n
nprint -S ACCT_FS01 -q LASER -U guest -n filename.txt
```

Синтаксис команд похож на синтаксис команды `ncpmount`.

Настройка Linux как сервера печати Novell

Программа pserver, которая позволяет Linux выступать в качестве сервера печати в сети NetWare, входит в пакет ncpfs. Альтернативная поддержка включена в пакет Mars_nwe.

Когда у вас на сервере настроены принтеры и установлена утилита pserver, необходимо добавить команды ее запуска в RC-файл.

Простейший вариант приведен далее:

```
pserver -S ACCT_01 -U LASER -P secret -q LASERJET
```

Эта команда предписывает утилите pserver войти на файловый сервер `ACCT_01` с именем пользователя `LASER` и паролем `secret` и брать задания из очереди печати `LASERJET`. Когда входящее задание печати будет переслано, то начнет действовать команда печати по умолчанию `lpr` для переноса задания печати на демон печати Linux. Очередь печати должна быть уже определена на файловом сервере, и пользователь должен иметь привилегии оператора для этой очереди.

Пользовательские и административные команды ncpfs

В пакет ncpfs входит набор пользовательских и административных команд, выполняющих различные функции: копирование файлов, печать, просмотр Bindery, отсылка сообщений по сети и т. п.

Команды пользователя

В качестве пользовательских применяются следующие команды:

- `ncopy` (Network Copy) — позволяет копировать файлы, используя функцию копирования NetWare вместо копирования по сети;
- `nprint` (Network Print) — позволяет послать файл в очередь печати на сервере NetWare;
- `nsend` (Network Send) — позволяет послать сообщение другим пользователям на сервере NetWare;
- `nwbols` (List Bindery Objects) — позволяет вам увидеть содержимое Bindery на сервере NetWare;
- `nwboprops` (List Properties of a Bindery Object) — позволяет просмотреть свойства объекта Bindery NetWare;
- `nwbpset` (Set Bindery Property) — позволяет установить свойства объекта Bindery NetWare;
- `nwbpvalues` (Print NetWare Bindery Objects Property Contents) — позволяет напечатать содержимое свойства Bindery NetWare;
- `nwfsinfo` (Fileserver Information) — печатает общую информацию о сервере NetWare;
- `nwpasswd` (Netware Password) — позволяет сменить пароль пользователя NetWare;
- `nwrights` (Netware Rights) — показывает список прав, ассоциированных с отдельным файлом или каталогом;
- `nwuserlist` (Userlist) — перечисляет пользователей, подключенных к файловому серверу NetWare;
- `pqlist` (Print Queue List) — показывает содержимое очереди печати NetWare;
- `slist` (Server List) — показывает список известных серверов NetWare.

Утилиты администрирования

В качестве утилит администрирования используются следующие команды:

- `nwbocreate` (Create a Bindery Object) — позволяет создать объект Bindery NetWare;
- `nwborm` (Remove Bindery Object) — позволяет удалить объект Bindery NetWare;
- `nwbpadd` (Add Bindery Property) — позволяет установить значение существующего свойства объекта Bindery NetWare;

- `nwbpcreate` (Create Bindery Property) — позволяет создать новое свойство для существующего объекта Bindery NetWare;
- `nwbprm` (Remove Bindery Property) — позволяет удалить свойство из объекта Bindery NetWare;
- `nwgrant` (Grant Trustee Rights) — позволяет установить попечительские права на каталог на файловом сервере NetWare;
- `nwrevoke` (Revoke Trustee Rights) — позволяет удалить попечительские права с каталога на файловом сервере NetWare.

Туннелирование IPX через IP

В том случае, если у вас две локальных сети Novell, между которыми есть только IP-сеть, и вам необходимо каким-либо образом соединить две эти сети — воспользуйтесь пакетом ipxtunnel.

Ipxtunnel позволяет пакетам IPX быть включенными в пакеты TCP/IP так, что они могут без потерь информации переноситься посредством соединения TCP/IP. Для нормального функционирования необходимо сконфигурировать и запустить пакет ipxtunnel на обоих концах туннеля.

Настройка

Настроить ipxtunnel не составляет труда. Пусть один конец туннеля (компьютер) называется q.odessa.ua, а второй — w.odessa.ua. Для конфигурации `ipxtunnel` используется файл /etc/ipxtunnel.conf. Этот файл позволяет указать порт UDP по умолчанию для использования в соединении TCP/IP, куда посылать инкапсулированные данные, на каком локальном интерфейсе должен слушать ipxtunnel и на какой отправлять пакеты IPX.

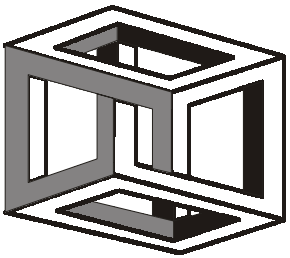
Пример конфигурационного файла:

```
#
# /etc/ipxtunnel.conf для q.odessa.ua
#
# Порт UDP для использования:       (по умолчанию 7666)
port 7777
#
# Удаленная машина, на которую отправлять пакеты IPX
remote w.odessa.ua
#
# Локальные интерфейсы, на которых искать пакеты IPX: (по умолчанию eth0)
interface eth0
interface eth1
```

Другой компьютер должен иметь похожий конфигурационный файл.

Литература и ссылки

- IPX HOWTO — настройка IPX-протокола.
- MARS HOWTO.
- **www.compu-art.de/mars_nwe/** — домашняя страница Mars_new.
- **www.osp.ru/pcworld/1998/05/44.htm** — Суханов А., Хименко В. Linux и Windows 95: эффективность совместной работы. — Мир ПК № 5/98.



ЧАСТЬ IV

На службе

Предыдущие части книги были необходимы в качестве базиса для *части IV*. Она предназначена для опытных пользователей и администраторов. Таким образом, книга будет служить вам верой и правдой в качестве справочного пособия долгое время, и вы периодически будете возвращаться к ней для решения специфических задач, возникающих в вашей работе. Здесь вы найдете описание основных приложений, используемых для организации НОРМАЛЬНОГО функционирования сети фирмы, подключенной к Интернету. В этой части рассмотрена защита сети от нежелательного воздействия, организация виртуальных частных сетей, учета сетевого трафика, настройка сетевых принтеров, изготовление бездисковых компьютеров и организация шлюза в Интернет.

Глава 16. Firewall

Глава 17. Сетевые принтеры

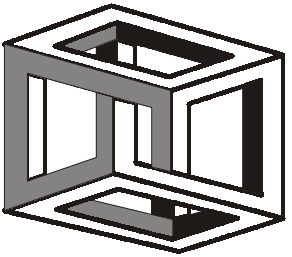
Глава 18. Организация шлюза в Интернет для локальной сети

Глава 19. Учет сетевого трафика

Глава 20. Виртуальные частные сети

Глава 21. Бездисковые компьютеры

ГЛАВА 16



Firewall

Эта глава посвящена защите сети от вторжения как извне, так и изнутри. Для защиты локальной сети используется комплекс программного обеспечения, в литературе известный как Firewall (брандмауэр), или межсетевой экран. Брандмауэр позволяет "отгородить" систему (или сеть) от внешнего мира. Он используется для предотвращения получения посторонними данных (или ресурсов) защищаемой сети, а также для контроля за внешними ресурсами, к которым имеют доступ пользователи вашей сети.

Чаще всего брандмауэр — это набор программ маршрутизации и фильтрации сетевых пакетов. Такие программы позволяют определить, можно ли пропустить данный пакет и если можно, то отправить его точно по назначению. Для того чтобы брандмауэр мог сделать это, ему необходимо определить набор правил фильтрации. Главная цель брандмауэра — контроль удаленного доступа извне или изнутри защищаемой сети или компьютера.

Брандмауэр позволяет лишь частично решить проблемы, связанные с обеспечением безопасного функционирования вашей сети. Как бы хорошо он ни был настроен, если вы вовремя не обновили программный пакет, в котором была найдена уязвимость, или кто-то узнал ваши логин и пароль — ждите больших неприятностей. Основная задача брандмауэра — разрешать функционирование только тем службам, которым было явно разрешено работать в вашей сети или защищаемом компьютере. В результате мы получаем маленькую дверцу, через которую в уютный внутренний мирок могут попасть только те гости, которых пропустила ваша охрана, а список этих гостей рекомендуется сузить до минимального.

Основными компонентами брандмауэра являются:

- ❒ политика безопасности сети;
- ❒ механизм аутентификации;
- ❒ механизм фильтрации пакетов.

О практической реализации этих компонентов мы поговорим несколько позже, а пока разберемся, какие бывают брандмауэры.

Типы брандмауэров

При построении брандмауэра обычно используется компьютер (компьютеры), непосредственно подключенный к Интернету и содержащий базовый набор средств, реализующих брандмауэр. Такой компьютер иногда называют *бастионом*.

Термин "брандмауэр" может приобретать различные значения в зависимости от принципа, положенного в основу работы средств защиты, сетевой архитектуры и схемы маршрутизации. Брандмауэры обычно подразделяют на три типа:

- брандмауэр с фильтрацией пакетов;
- прикладной шлюз;
- универсальный Proxy-сервер.

Брандмауэр с фильтрацией пакетов, как правило, действует на сетевом и транспортном уровнях и реализуется в составе операционной системы. Источником информации для фильтрации является содержимое заголовков IP-пакетов, на основе которого брандмауэр принимает решение, по какому маршруту следует направить пакет.

Прикладной шлюз реализуется посредством выбора сетевой архитектуры и конфигурации системы. Сетевой трафик никогда не проходит через компьютер, на котором выполняется прикладной шлюз. Чтобы получить доступ в Интернет, локальный пользователь должен зарегистрироваться на прикладном шлюзе. Компьютер, содержащий прикладной шлюз, может быть защищен брандмауэрами с фильтрацией пакетов как извне, так и из локальной сети.

Proxy-сервер обычно реализуется в виде независимого приложения, управляющего доступом к различным типам сетевых служб. Для клиентов Proxy-сервер выполняет роль сервера, предоставляющего информацию. Вместо того чтобы непосредственно обращаться к удаленным серверам, клиентские программы обращаются к Proxy-серверу. Получив обращение клиента, Proxy-сервер устанавливает связь с удаленным узлом от своего имени, при этом он заменяет в пакете адрес клиента своим адресом. Подобный сервер может контролировать целостность данных, осуществлять проверку на наличие вирусов и обеспечивать выполнение правил системной политики, определяющих обмен высокоуровневыми данными.

Помимо этого, брандмауэры можно разделить по типу построения защиты:

- пороговый и его разновидность — бастионного типа;
- организующий так называемую демилитаризованную зону.

Брандмауэр порогового типа призван защитить локальную сеть от атак извне, а при соответствующей настройке и от атак изнутри. Такого типа брандмауэры обычно используются для защиты небольшой сети или даже одного компьютера. Как правило, сетевые службы, предоставляющие услуги вне локальной сети (HTTP, FTP и т. п.), размещаются на том же компьютере, что и брандмауэр.

Организация демилитаризованной зоны оправдана тогда, когда в сети выделено несколько специальных компьютеров для интернет-сервисов, предоставляемых большому миру, а также при отсутствии уверенности в благонадежности собственных сотрудников. Для организации демилитаризованной зоны используются, по меньшей мере, два брандмауэра — один для защиты демилитаризованной зоны от проникновения извне, а второй — от проникновения из вашей собственной локальной сети. Организация демилитаризованной зоны сложнее, чем организация брандмауэра бастионного типа, но взамен вы получаете большую защиту ваших данных.

Брандмауэр с фильтрацией пакетов

Брандмауэр с фильтрацией пакетов представляет собой "сито" для проходящих через него входящих и исходящих пакетов. В операционной системе Linux реализован брандмауэр, позволяющий контролировать ICMP-, UDP- и TCP-пакеты. Брандмауэр с фильтрацией пакетов организован как механизм, реализующий набор разрешающих и запрещающих правил для входящих и исходящих пакетов. Этот набор правил определяет, какие пакеты могут проходить через конкретный сетевой интерфейс.

Брандмауэр с фильтрацией пакетов может производить с проходящим пакетом всего три действия:

- переслать пакет в узел назначения;
- удалить пакет без уведомления посылающей пакет стороны;
- вернуть передающему компьютеру сообщение об ошибке.

Несмотря на простоту таких действий, в большинстве случаев их достаточно для организации эффективной защиты. Как правило, брандмауэр устанавливается для того, чтобы контролировать данные, которыми компьютеры обмениваются с Интернетом. В результате работы фильтрующего брандмауэра отсеиваются недопустимые обращения к узлам внутренней сети и запрещается передача из внутренней сети в Интернет для пакетов, определенных правилами фильтрации.

В целях получения более гибкой системы правила фильтрации пакетов составляются для каждого сетевого интерфейса, в них учитываются IP-адреса источника и получателя, номера портов TCP и UDP, флаги TCP-соединений и ICMP-сообщений. Причем правила для входящих и исходящих

пакетов различаются. Это значит, что при настройке фильтрующего брандмауэра правила для конкретного сетевого интерфейса представляются как отдельные правила для входящей и исходящей информации, поскольку входящие и исходящие пакеты обрабатываются брандмауэром независимо друг от друга. Списки правил, которые управляют фильтрацией сетевых пакетов, поступающих извне в локальную сеть и отправляемых из локальной сети в Интернет, принято называть *цепочками* (chains). Термин "цепочка" используется потому, что при проверке пакета правила применяются последовательно одно за другим, пока не обнаружится подходящее правило для сетевого пакета или список правил не будет исчерпан.

Описанный механизм фильтрующего брандмауэра достаточно эффективен, однако он не обеспечивает полной безопасности локальной сети. Брандмауэр — это всего лишь один из элементов общей схемы защиты. Анализ заголовков сетевых пакетов — операция слишком низкого уровня, для того чтобы реально выполнять аутентификацию и контролировать доступ. В процессе фильтрации пакетов практически невозможно распознать отправителя сообщения и проанализировать смысл передаваемой информации. Из всего набора данных, пригодных для аутентификации, на рассматриваемом уровне доступен только IP-адрес отправителя, однако этот адрес очень легко подделать, на чем и базируется множество способов сетевых атак. Несмотря на то, что средства фильтрации пакетов позволяют эффективно контролировать обращение к портам, использование протоколов обмена и содержимое пакетов, проверку данных необходимо продолжить на более высоком уровне.

Политика организации брандмауэра

При построении брандмауэров используются два основных подхода:

- запрещается прохождение всех пакетов, пропускаются лишь те, которые удовлетворяют явно определенным правилам;
- разрешается прохождение всех пакетов, за исключением пакетов, удовлетворяющих определенным правилам.

Или перефразируя, запрещено все, что не разрешено, и разрешено все, что не запрещено.

С практической точки зрения лучше использовать подход, при котором поступающий пакет по умолчанию отвергается (запрещено все, что не разрешено). В этом случае организация безопасности сети достигается достаточно просто, но с другой стороны, приходится предусматривать возможность обращения к каждой сетевой службе и использование каждого конкретного протокола. Это означает, что администратор сети, занимающийся настройкой брандмауэра, должен точно знать, какие протоколы применяются в его

локальной сети. При использовании подхода, предусматривающего запрет по умолчанию, приходится предпринимать специальные меры всякий раз, когда необходимо разрешить доступ к какому-то ресурсу, однако эта модель с нашей точки зрения более надежна, чем противоположный вариант.

Политика разрешения по умолчанию позволяет добиться функционирования системы малыми усилиями, но при этом необходимо предусмотреть каждый конкретный случай, при котором требуется запретить доступ. Может случиться так, что необходимость внесения запретов станет ясна лишь тогда, когда в результате несанкционированного доступа сети будет нанесен значительный ущерб.

В обоих случаях для конфигурации брандмауэра используются цепочки правил. Каждая цепочка представляет собой набор правил, заданных явным образом, и политику по умолчанию. Пакет проверяется на соответствие каждому из правил, а правила выбираются из списка последовательно до тех пор, пока не будет обнаружено соответствие сетевого пакета одному из них. Если пакет не удовлетворяет ни одному из заданных правил, с сетевым пакетом производятся действия, определенные политикой по умолчанию.

В процессе работы брандмауэр может *пропустить* сетевой пакет (ACCEPT), *запретить* прохождение сетевого пакета (DENY) либо отказать сетевому пакету в прохождении, т. е. *отклонить* его (REJECT). С прохождением сетевого пакета все ясно, а чем же отличаются запрет и отклонение сетевого пакета? При отклонении сетевого пакета (REJECT) сам пакет удаляется, а его отправителю возвращается ICMP-сообщение об ошибке. При запрете прохождения сетевого пакета (DENY) сам пакет удаляется, но отправитель не оповещается об удалении сетевого пакета.

В большинстве случаев запрет сетевого пакета считается лучшим решением, чем отказ в прохождении сетевого пакета. Во-первых, отправка сообщения об ошибке увеличивает сетевой трафик, а во-вторых, сообщения об ошибке могут быть использованы для организации атаки с целью вывода из строя сервера. Помимо этого, любое ответное действие на "неправильные" пакеты предоставляет взломщику дополнительную информацию о конфигурации вашей системы.

Фильтрация сетевых пакетов

В этой части главы мы рассмотрим, на основании каких данных можно производить фильтрацию входящих и исходящих сетевых пакетов, а также каким образом определять "неправильные" сетевые пакеты.

Фильтрация входящих пакетов

Рассмотрение построения брандмауэра логично начать со входящих пакетов. Поскольку именно извне обычно происходит проникновение в сеть.

Фальсификация исходящего адреса и недопустимые адреса

Рассмотрим признаки, по которым можно однозначно судить о поддельности сетевого пакета, поступающего из Интернета, или о проблемах прикладного программного обеспечения. На основании этих признаков нужно будет задать соответствующие правила фильтрации, чтобы ваш брандмауэр, обнаружив такой "неправильный" исходящий адрес в пакете, мог запретить прохождение сетевого пакета.

1. Если в заголовке сетевого пакета в качестве исходного адреса указан адрес вашего компьютера. В процессе сетевого обмена невозможна ситуация, при которой сетевой пакет, отправленный с вашего компьютера, вернулся бы через внешний интерфейс. Следовательно, такой сетевой пакет — поддельный.
2. Если в качестве исходящего IP-адреса указан адрес, попадающий в зарезервированный диапазон адресов, предназначенных для внутреннего применения. Согласно правилам распределения IP-адресов в каждом из классов IP-адресов A, B и C существуют группы IP-адресов, выделенных для организации внутренних локальных сетей. В Интернете эти адреса не используются. При правильной конфигурации программного обеспечения через внешний порт не может прийти пакет с адресом источника, попадающий в один из перечисленных далее диапазонов:
   - класс A — в диапазоне от 10.0.0.0 до 10.255.255.255;
   - класс B — в диапазоне от 172.16.0.0 до 172.31.255.255;
   - класс C — в диапазоне от 192.168.0.0 до 192.168.255.255.
3. Если в качестве исходящего IP-адреса указан IP-адрес класса D, предназначенный для группового вещания. Адреса класса D, специально выделенные для организации группового вещания, находятся в диапазоне адресов от 224.0.0.0 до 239.255.255.255 и ни при каких обстоятельствах не могут выступать в качестве адреса источника.
4. Если в качестве исходящего IP-адреса использован зарезервированный IP-адрес класса E. Класс E зарезервирован для будущего использования, ему принадлежат адреса от 240.0.0.0 до 247.255.255.255. Если брандмауэр встретит пакет с исходным адресом класса E, он должен предпринять меры, необходимые для того, чтобы такой пакет не попал в локальную сеть.
5. Если в качестве исходящего IP-адреса использован адрес, принадлежащий интерфейсу обратной петли. Интерфейс обратной петли предназначен для локального использования сетевыми службами. Как правило, для обращения к интерфейсу обратной петли используется адрес 127.0.0.1, а вообще за интерфейсом локальной сети зарезервирована целая подсеть 127.*x.x.x.* Адрес интерфейса обратной петли не может присутствовать в заголовке пакета, полученного через внешний сетевой интерфейс.

6. Если в качестве исходящего IP-адреса использован некорректный широковещательный адрес. Широковещательный адрес — это специальный тип адреса, определяющий передачу сетевого пакета на все компьютеры в сети. В качестве исходного адреса при широковещательной передаче может выступать обычный IP-адрес или адрес 0.0.0.0.

Фильтрация на основе адреса источника

При фильтрации пакетов единственный способ идентификации отправителя сетевого пакета — проверка IP-адреса источника в заголовке пакета. Одним из самых распространенных приемов при организации сетевых атак является фальсификация сетевых пакетов, при которой отправитель заменяет свой IP-адрес в заголовке сетевого пакета другим значением. Для подмены может быть выбран несуществующий или реальный IP-адрес, принадлежащий другому узлу.

Блокирование ненадежных узлов

Еще одна схема фильтрации, основанная на анализе IP-адресов источников, — это блокирование доступа с компьютеров, IP-адреса которых попадают в определенный диапазон. Как правило, таким образом отсекаются "подозрительные" компьютеры и целые сети, в частности, обычно это происходит с сетями различных учебных заведений или разнообразных интернет-клубов, поскольку именно там молодежь любит "пошалить" в сети.

Работа с ограниченным набором удаленных узлов

В том случае, если вы организуете корпоративную сеть, не исключено, что вам потребуется таким образом настроить брандмауэр, чтобы некоторые типы пакетов принимались только в том случае, если они были отправлены с компьютеров с определенными адресами. Например, для организации системы передачи приватной информации.

Фильтрация на основе адреса назначения

В большинстве случаев фильтрация на основе адреса назначения выполняется автоматически. Сетевой интерфейс игнорирует пакеты, не адресованные непосредственно ему. Исключением являются широковещательные пакеты, адресованные всем узлам сети.

Фильтрация на основе порта источника

Номер порта источника, содержащийся в заголовке пакета, предназначен для идентификации программы-отправителя сетевого пакета, выполняющейся на удаленном узле. В запросах удаленных клиентов к вашему серверу содержатся различные номера портов, а в ответах сервера клиентам — один и тот же порт.

Фильтрация на основе порта назначения

Порт назначения определяет программу на вашем компьютере, которой предназначен пакет. В запросах удаленных клиентов, передаваемых на сервер, содержится один и тот же порт назначения, а в ответах сервера клиентам — различные номера портов.

Фильтрация на основе информации о состоянии TCP-соединения

В правилах обработки сетевых пакетов могут использоваться флаги, определяющие состояние TCP-соединения, поскольку любое сетевое соединение проходит через определенные состояния. Состояния клиента и сервера различаются между собой.

В первом пакете, отправленном удаленным клиентом, установлен флаг `SYN`, а флаг `ACK` сброшен. Передача такого пакета является началом в установлении TCP-соединения. Во всех последующих сетевых пакетах, передаваемых клиентом, установлен флаг `ACK`, а флаг `SYN` сброшен.

Пакеты, передаваемые удаленными серверами, всегда являются ответами на предыдущие обращения клиентов. В каждом пакете, поступившем от удаленного сервера, должен быть установлен флаг `ACK`, поскольку TCP-соединение никогда не устанавливается по инициативе сервера.

На основе анализа флагов можно отсеивать "неправильные" сетевые пакеты, которые могут являться признаком сетевой атаки.

Фильтрация исходящих пакетов

Фильтрация исходящих сетевых пакетов позволит исключить попадание в Интернет сетевых пакетов, передаваемых по локальной сети, а также избежать нежелательных обращений к серверам с узлов локальной сети. Источником таких обращений могут быть неверно сконфигурированные или вредоносные программы, запускаемые пользователями на их компьютерах.

Фильтрация на основе адреса источника

При этом типе фильтрации необходимо сформировать правила фильтрации таким образом, чтобы пакет, в котором указан адрес источника, не совпадающий ни с одним из адресов компьютеров вашей локальной сети, не был пропущен брандмауэром. Это может вызвать некоторые затруднения, если в вашей организации разветвленная локальная сеть или IP-адреса выдаются динамически. Однако эти проблемы решаемы.

Фильтрация на основе адреса назначения

Как уже упоминалось ранее, возможна ситуация, при которой вам потребуется ограничить передачу сетевых пакетов за пределы локальной сети адресами отдельных сетей или отдельных компьютеров. Эти адреса или диапазоны адресов могут быть указаны в правилах, задаваемых брандмауэру.

Фильтрация на основе порта источника

Проверка портов, указанных в заголовках сетевых пакетов, может выполняться как для клиентов, запущенных в локальной сети, так и для серверов. Такая проверка позволяет убедиться в том, что программы работают корректно и защищают Интернет от попадания в него внутреннего трафика локальной сети.

Пакеты, передаваемые сервером, обязательно должны содержать в заголовке порт источника, совпадающий с номером порта, выделенным для службы данного типа. Проверка номера порта представляет собой проверку конфигурации сетевых протоколов.

Фильтрация на основе порта назначения

Поскольку локальные клиенты могут обращаться к удаленным серверам лишь по конкретным номерам портов, фильтрация исходящих пакетов является одновременно средством контроля за использованием протоколов. Запрет прохождения сетевых пакетов на основе порта назначения не дает возможности пользователям локальной сети проводить сканирование портов удаленных компьютеров, ведь обычно сканирование портов — предвестник сетевой атаки.

Защита локальных служб

Как правило, локальные сервисы используются только внутри вашей сети, и предоставление доступа к этим службам извне нецелесообразно, а зачастую и вредно. Поэтому самый простой способ уберечься от проникновения в систему через один из сервисов — запретить доступ к сервису извне. Однако существуют службы, которые могут вызвать большое количество проблем при организации запрета доступа, например ICQ.

Один из способов защитить службы, предназначенные для внутреннего использования, — отказаться от размещения соответствующих серверов на компьютерах, доступных из глобальной сети. Однако в небольших сетях обычно существует один-единственный сервер, зачастую выполняющий роль брандмауэра, поэтому в некоторых случаях компромиссы неизбежны.

Для защиты сервера от обращений из Интернета можно применить брандмауэр, выполняющий фильтрацию пакетов по порту назначения. Наличие такого брандмауэра позволяет запускать в локальной сети большое количество служб, не подвергая серьезной опасности сетевые ресурсы.

Программа ipchains

Как уже упоминалось ранее, брандмауэр — это набор программных средств для организации защиты вашей сети. Большая часть функциональности брандмауэра интегрирована в ядре операционной системы Linux, но для

создания и управления цепочками правил используются внешние программы. Для этих целей и предназначена программа ipchains. В последнее время активно внедряется программа iptables, о которой вы узнаете немного позже. Ipchains использовалась в ядрах версий 2.0—2.2. В ядрах версий 2.4 можно применять как ipchains, так и iptables.

Правила фильтрации пакетов, составляющих цепочки input, output и forward (входящие, исходящие и переадресация), содержатся во внутренних таблицах ядра операционной системы Linux. Каждое правило может быть включено в начало цепочки или добавлено в ее конец. Для определенности будем считать, что все правила, определяемые в данной главе, добавляются в конец цепочки. Порядок, в котором задаются правила, определяет порядок, в котором они будут включены в цепочку, и последовательность их применения к каждому пакету.

При поступлении информационного пакета на сетевой интерфейс извне содержимое заголовка этого пакета проверяется. Правила, принадлежащие цепочке input данного сетевого интерфейса, применяются последовательно одно за другим до тех пор, пока не будет найдено такое, которому удовлетворяет данный сетевой пакет. Соответственно, каждый сетевой пакет, отправляемый вовне, проверяется на соответствие правилам, содержащимся в цепочке output сетевого интерфейса. При обнаружении первого соответствия правилу проверка прекращается и к пакету применяется действие, указанное в составе правила: ACCEPT, REJECT или DENY. Если пакет не удовлетворяет ни одному из правил, содержащихся в цепочке, вступает в действие политика по умолчанию. Таким образом, при работе брандмауэра пакет обрабатывается по первому из правил, которому он удовлетворяет.

Программе ipchains параметры передаются при вызове в командной строке. Формат командной строки приведен далее:

```
ipchains   -A|I [<цепочка>] [-i <интерфейс>] [-p <протокол>] [  [!]  -y]
           [-s <адрес> [<порт> [: <порт>]]]
           [-d <адрес> [<порт> [: <порт>]]] — j <действие> [l]
```

В правилах, управляющих работой брандмауэра, предусматривается проверка адреса источника и адреса назначения. Для сравнения могут использоваться IP-адрес узла, диапазон IP-адресов, символьное имя узла и имя домена.

Программа ipchains позволяет задавать после IP-адреса дескриптор маски. *Дескриптор маски* — это целое число, которое может принимать значения от 0 до 32, и определяет число битов в маске. Дескриптор маски указывает, сколько старших битов адреса узла должны в точности совпадать с адресом, заданным в составе правила. Дескриптор маски, равный 32, означает, что адрес узла должен полностью совпадать с адресом, указанным в правиле. Если дескриптор маски отсутствует, считается, что он равен 32. Так адрес 192.168.0.45 означает то же самое, что и выражение 192.168.0.45/32.

Опции ipchains

В табл. 16.1 приведены наиболее часто применяемые опции программы ipchains.

Таблица 16.1. *Опции программы ipchains*

Опция	Описание
`-A [<цепочка>]`	Добавляет правило к концу цепочки. Используются встроенные цепочки input, output и forward. Если цепочка не указана, правило добавляется ко всем цепочкам
`-I [<цепочка>]`	Включает правило в начало цепочки. Используются встроенные цепочки input, output и forward. Если цепочка не указана, правило включается во все цепочки
`-i <интерфейс>`	Определяет сетевой интерфейс, к которому должно применяться данное правило. Если сетевой интерфейс не указан, правило применяется ко всем сетевым интерфейсам
`-p <протокол>`	Определяет протокол семейства TCP/IP, к которому должно применяться правило. Если опция `-p` не указана, правило применяется ко всем протоколам. В качестве имен протоколов могут быть заданы `tcp`, `udp`, `icmp` и `all`. Разрешается использование имен и числовых значений, указанных в файле /etc/protocols
`-y`	В пакете, содержащем запрос на установление TCP-соединения, флаг `SYN` должен быть установлен, а флаг `ACK` — сброшен. Если данная опция не указана, проверка состояния флагов `SYN` и `ACK` не производится
`! -y`	В пакете, который передается в ответ на запрос на установление TCP-соединения, а также во всех последующих пакетах флаг `ACK` должен быть установлен. Если опция `! -y` не указана, состояние флага `ACK` не проверяется
`-s <адрес> [<порт>]`	Определяет исходящий адрес пакета. Если исходящий адрес не указан, обрабатываются пакеты, переданные с любого узла. Если указан порт или диапазон портов, правило применяется только к пакетам, содержащим заданный номер порта. Если порт не указан, правило применяется ко всем пакетам, независимо от номера порта источника. При указании диапазона портов задаются начальный и конечный номера, разделенные двоеточием (например, 1024:65535). Если в опции `-s` задается порт, адрес также должен быть указан

Таблица 16.1 *(окончание)*

Опция	Описание
-d <адрес> [<порт>]	Определяет адрес назначения пакета. Если адрес назначения не указан, обрабатываются все пакеты, передаваемые любому узлу. Если указан порт или диапазон портов, правило применяется только к пакетам, содержащим заданный номер порта. Если порт не указан, правило применяется ко всем пакетам, независимо от номера порта назначения. При указании диапазона портов задаются начальный и конечный номера, разделенные двоеточием (например, 1024:65535). Если в опции -d задается порт, адрес также должен быть указан
-j <действие>	Определяет действие, которое должно быть выполнено над пакетом (ACCEPT, REJECT или DENY). Для цепочки forward данный параметр также может принимать значение MASQ (masquerade — маскировка)
-1	Если пакет удовлетворяет правилу, в файл протоколов (по умолчанию /var/log/messages) должно быть записано информационное сообщение ядра

Символьные константы

Для улучшения удобочитаемости правил фильтрации и для удобства сопровождения в сценариях брандмауэра рекомендуется использовать символьные имена. В табл. 16.2 приведены некоторые символьные константы, используемые при конфигурировании брандмауэра.

Таблица 16.2. *Символьные константы, используемые при описании правил фильтрации*

Константа	Описание
EXTERNAL_INTERFACE = "eth0"	Сетевой интерфейс, подключенный к Интернету
Internal_ INTERFACE = "eth1"	Сетевой интерфейс, подключенный к локальной сети (для брандмауэра бастионного типа)
LAN_1="192.168.1.0/24"	Диапазон адресов внутренней сети
LAN_IPADDR_1="192.168.1.1"	Адрес внутреннего интерфейса
LOOPBACK_INTERFACE = "lo"	Интерфейс локальной петли
IPADDR = "ipaddress"	Адрес вашего компьютера

Таблица 16.2 *(окончание)*

Константа	Описание
ANYWHERE ="any/0"	Произвольный адрес
MY_ISP = " ip range"	Диапазон адресов провайдера
LOOPBACK="127.0.0.0/8"	Диапазон адресов обратной петли
CLASS_A ="10.0.0.0/8"	Адреса класса A для локальных сетей
CLASS_B ="172.16.0.0/12"	Адреса класса B для локальных сетей
CLASS_C ="192.168.0.0/16"	Адреса класса C для локальных сетей
CLASS_D_MULTICAST ="224.0.0.0/4"	Адреса класса D для группового вещания
Class_E_Reserved_Net ="240.0.0.0/5"	Адреса класса E. Зарезервировано
BROADCAST_SRC ="0.0.0.0"	Исходящий широковещательный адрес
BROADCAST_DEST ="255.255.255.255"	Широковещательный адрес
NAMESERVER = "mydns"	Адрес DNS-сервера
SMTP_GATEWAY="isp.server"	Адрес почтового шлюза провайдера
POP_SERVER="isp.server"	Адрес POP-сервера провайдера
NEWS_SERVER="isp.server"	Адрес NEWS-сервера провайдера
IMAP_SERVER="isp.server"	Адрес IMAP-сервера провайдера
PRIVPPORTS="0:1023"	Номера привилегированных портов
UNPRIVPORTS="1024:65535"	Номера непривилегированных портов
SSH_PORTS="1000:1023"	Номера привилегированных портов для протокола SSH — ограничиваем 24 одновременно возможными соединениями

Создание правил фильтрации

В этом разделе мы рассмотрим создание правил фильтрации для нашего брандмауэра. Эти правила ориентированы для среднестатистического брандмауэра, который полностью удовлетворяет потребностям одного компьютера или малой локальной сети. Для более серьезных случаев вы сможете оформить свой набор правил, используя приведенные далее как базис.

Удаление существующих правил

Начиная создание своих правил фильтрации, обязательно удалите уже существующие правила. Это необходимо с точки зрения элементарной

предосторожности — вдруг в системе уже определены некоторые правила фильтрации, идущие вразрез с вашей политикой безопасности.

Удаление правил фильтрации называется *сбросом цепочки*. Для сброса встроенных цепочек не обязательно обращаться к ним явно. Все три цепочки — input, output и forward — можно сбросить с помощью одной команды:

```
ipchains -F
```

Определение политики по умолчанию

После удаления правил фильтрации автоматически устанавливается политика фильтрации сетевых пакетов по умолчанию, согласно которой разрешается прохождение всех сетевых пакетов. Таким образом, до тех пор пока вы не внесете изменения в политику фильтрации сетевых пакетов, непосредственно фильтрация производиться не будет.

Для обеспечения безопасности вашей операционной системы политика по умолчанию должна быть выбрана так, чтобы входящие сетевые пакеты удалялись без передачи сообщений на хосты, посылающие сетевые пакеты. Исходящим сетевым пакетам должно быть отказано в прохождении, а компьютеры, с которых эти сетевые пакеты были отправлены, должны получать ICMP-сообщения об ошибке. Обратившись к компьютеру вашей сети, программа, выполняющаяся на одном из компьютеров, размещенных в Интернете, не получит никакой информации о том, существует ли сервер, указанный в запросе. Такая же программа на локальном компьютере сразу получит сообщение о том, что операция, которую она собиралась выполнить, недопустима.

В следующих трех строках мы задаем уничтожение сетевых пакетов для входящей цепочки и их отклонение для исходящей цепочки и цепочки маршрутизации, компьютеры, посылающие сетевые пакеты, получат уведомление об ошибке.

```
ipchains -P input   DENY
ipchains -P output  REJECT
ipchains -P forward REJECT
```

После ввода в действие приведенных ранее правил весь сетевой трафик оказывается блокированным, в том числе и весь трафик, проходящий через интерфейс обратной петли.

Разрешение прохождения пакетов через интерфейс обратной петли

Поскольку в предыдущем разделе мы заблокировали весь сетевой трафик, автоматически возникнут проблемы с рядом программ, исполняемых на вашем компьютере, т. к. многие из них для нормального функционирова-

ния используют интерфейс обратной петли. Поэтому мы должны обеспечить прохождение всего сетевого трафика через интерфейс обратной петли. Поскольку этот интерфейс недоступен из-за пределов системы, подобные установки не могут повлечь за собой нежелательных последствий.

Правила, разрешающие прохождение сетевых пакетов без ограничений, очень просты. В данном случае надо нейтрализовать влияние политики по умолчанию на интерфейс обратной петли. Для этого введем следующие правила:

```
ipchains — A  input    -i $LOOPBACK_INTERFACE -j ACCEPT
ipchains — A  output  -i $LOOPBACK_INTERFACE -j ACCEPT
```

Таким простым способом прохождение трафика через интерфейс обратной петли будет восстановлено.

Запрет прохождения пакетов с фальсифицированными адресами

Как уже упоминалось ранее, фальсификация адресов — один из признаков сетевых атак, поэтому следует бороться с сетевыми пакетами, имеющими сфальсифицированный адрес. Первое и очевидное правило — запретить прием сетевых пакетов, якобы отправленных с внешнего интерфейса вашего узла:

```
ipchains -A input  -i  $EXTERNAL_INTERFACE  -s  $IPADDR  -j  DENY  -1
```

Это правило отсекает входящие сетевые пакеты, содержащие в качестве адреса источника сетевой адрес вашего внешнего интерфейса. В том случае, если вы посылаете сетевой пакет на свой компьютер, он пройдет не через внешний сетевой интерфейс, а через интерфейс обратной петли. Если система настроена нормально, сетевой пакет, направленный на локальный компьютер, никогда не попадет на внешний интерфейс. В противном случае — либо у вас проблемы с сетевыми настройками, либо кто-то пытается пробраться к вам в систему, поскольку весь сетевой трафик, идущий через интерфейс обратной петли, проходит внутри системы, и любой сетевой пакет, содержащий такой адрес, является поддельным.

Следующие два правила запрещают пакеты, содержащие в качестве исходящего адреса интерфейс обратной петли:

```
ipchains -A input    -i $EXTERNAL_INTERFACE  -s  $LOOPBACK  -j DENY
ipchains -A output  -i  $EXTERNAL_INTERFACE  -s  $LOOPBACK  -j DENY  -l
```

Далее необходимо отсечь сетевые пакеты, исходящие адреса которых попадают в диапазон IP-адресов, выделенных для использования во внутренних сетях. Маршрутизаторы не должны обрабатывать пакеты с исходящими адресами, принадлежащими внутренним сетям.

Тем же самым ограничениям должны подвергаться и исходящие сетевые пакеты, у которых адреса назначения попадают в диапазон IP-адресов, выделенных для использования во внутренних сетях, поскольку в Интернете не могут существовать адреса, предназначенные для применения исключительно в локальных сетях.

Приведенные далее наборы правил запрещают прохождение входящих и исходящих сетевых пакетов в случае, если адрес источника или адрес назначения принадлежит диапазонам сетевых адресов классов A, B и C, выделенных для использования в локальных сетях.

```
# Запретить прохождение сетевых пакетов,
# которые содержат адрес источника,
# принадлежащий диапазону адресов класса A,
# предназначенных для внутреннего использования.
ipchains -A input    -i  $EXTERNAL_INTERFACE  -s  $CLASS_A  -j DENY
ipchains -A input    -i  $EXTERNAL_INTERFACE  -d  $CLASS_A  -j DENY
ipchains -A output  -i  $EXTERNAL_INTERFACE  -s  $CLASS_A  -j DENY  -l
ipchains -A output  -i  $EXTERNAL_INTERFACE  -d  $CLASS_A  -j DENY  -l

# Запретить прохождение сетевых пакетов,
# которые содержат адрес источника,
# принадлежащий диапазону адресов класса B,
# предназначенных для внутреннего использования.
ipchains -A input    -i  $EXTERNAL_INTERFACE  -s  $CLASS_B  -j DENY
ipchains -A input    -i  $EXTERNAL_INTERFACE  -d  $CLASS_B  -j DENY
ipchains -A output  -i  $EXTERNAL_INTERFACE  -s  $CLASS_B  -j DENY  -l
ipchains -A output  -i  $EXTERNAL_INTERFACE  -d  $CLASS_B  -j DENY  -l

# Запретить прохождение сетевых пакетов,
# которые содержат адрес источника,
# принадлежащий диапазону адресов класса C,
# предназначенных для внутреннего использования.
ipchains -A input    -i  $EXTERNAL_INTERFACE  -s  $CLASS_C  -j DENY
ipchains -A input    -i  $EXTERNAL_INTERFACE  -d  $CLASS_C  -j DENY
ipchains -A output  -i  $EXTERNAL_INTERFACE  -s  $CLASS_C  -j DENY  -l
ipchains -A output  -i  $EXTERNAL_INTERFACE  -d  $CLASS_C  -j DENY  -l
```

Следующее правило, которое мы должны создать, необходимо для блокирования сетевых пакетов, содержащих недопустимые широковещательные адреса.

```
ipchains -A input  -i $EXTERNAL_INTERFACE -s  $BROADCAST_DEST  -j DENY -l
ipchains -A input  -i $EXTERNAL_INTERFACE -d  $BROADCAST_SRC  -j DENY  -l
```

Первое из приведенных правил запрещает пакеты с исходящим адресом 255.255.255.255. Второе правило — с адресом назначения 0.0.0.0. Подобные

пакеты появляются не в результате ошибки, а являются признаком попытки атаки на вашу сеть.

Адреса группового вещания могут применяться лишь в качестве адреса назначения. Следующие правила выявляют фальсифицированные сетевые пакеты и фиксируют случаи их появления в файлах протоколов.

```
# Запретить пакеты, содержащие адреса класса D.
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_D_MULTICAST -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_D_MULTICAST -j REJECT -l
```

Групповое вещание производится с использованием протокола UDP. В сетевых пакетах, передаваемых посредством группового вещания, адрес назначения отличается от пакетов, которые передаются в процессе обычного обмена между двумя узлами. Следующее правило запрещает передачу сетевых пакетов группового вещания с локального узла:

```
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_D_MULTICAST -j REJECT -l
```

Фильтрация ICMP-сообщений

Сообщения ICMP передаются при возникновении различных ситуаций, в том числе ошибочных. Они генерируются программами, анализирующими состояние сети, например ping или traceroute. В табл. 16.3 приведены ICMP-сообщения, которые могут представлять интерес для администратора сети.

Таблица 16.3. *Часто встречающиеся типы ICMP-сообщений*

Тип сообщения	Символьное имя	Описание
0	Echo Reply	Отклик программы ping
3	Destination Unreachable	Сообщение об ошибке: один из маршрутизаторов по пути следования сетевого пакета не может доставить данные на следующий узел
4	Source Quench	Сообщение, предназначенное для управления потоком между двумя маршрутизаторами или между маршрутизатором и обычным узлом
5	Redirect	Если маршрутизатор знает о наличии более короткого пути, данное сообщение возвращается узлу, с которого был передан сетевой пакет
8	Echo Request	Запрос программы ping

Таблица 16.3 *(окончание)*

Тип сообщения	Символьное имя	Описание
11	Time Exceeded	Данное сообщение передается, когда количество узлов, через которые прошел сетевой пакет, превышает максимально допустимое
12	Parameter Problem	В заголовке сетевого пакета была обнаружена недопустимая запись

Сообщения об ошибках и управляющие сообщения

При настройке брандмауэра необходимо обеспечить прохождение четырех типов сообщений:

- Source Quench — подавление источника;
- Parameter Problem — некорректный параметр;
- Destination Unreachable (подтип Fragmentation Needed) — узел назначения недоступен (для входящих сообщений);
- Destination Unreachable (подтип Fragmentation Needed) — узел назначения недоступен (для исходящих сообщений).

Еще четыре типа ICMP-сообщений также могут быть разрешены для прохождения через брандмауэр. Это Echo Request (эхо-запрос), Echo Reply (эхо-ответ), различные подтипы исходящих сообщений Destination Unreachable, а также сообщение Time Exceeded (превышение времени). Остальные сообщения желательно игнорировать, чтобы они были удалены в соответствии с политикой по умолчанию.

Из всех типов сообщений, которые следует игнорировать, в таблице приведено только сообщение Redirect (перенаправление). Данное сообщение может быть использовано для организации атаки с целью вывода из строя сервисных средств. Остальные типы сообщений в основном предназначены для организации взаимодействия маршрутизаторов.

В последующих разделах описываются типы сообщений, поддержка которых необходима для обеспечения работы хостов локальной сети.

Управляющее сообщение *Source Quench*

ICMP-сообщение типа Source Quench (подавление источника) передается в тех случаях, когда маршрутизатор передает пакеты быстрее, чем принимающий узел может их обработать. Source Quench — одно из простейших средств контроля обмена данными на сетевом уровне. Обычно такими со-

общениями обмениваются компьютеры, непосредственно связанные между собой.

Следующие правила разрешают прохождение входящих и исходящих ICMP-сообщений `Source Quench`:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 4 -d
$IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 4 -d
$IPADDR -j ACCEPT
```

Если компьютер, которому предназначены пакеты, возвращает сообщение `Source Quench`, передающий узел должен снизить скорость обмена. Со временем он начинает повышать скорость передачи данных и делает это до тех пор, пока не получает следующее сообщение `Source Quench`.

Сообщение *Parameter Problem*

ICMP-сообщение типа `Parameter Problem` (некорректный параметр) возвращается в том случае, когда в заголовке сетевого пакета содержится недопустимая запись, либо если контрольная сумма заголовка сетевого пакета не соответствует контрольной сумме, указанной передающим хостом.

Следующие правила разрешают прохождение входящих и исходящих ICMP-сообщений `Parameter Problem`:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 12 -d
$IPADDR -j ACCEPT
pchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 12 -d
$IPADDR -j ACCEPT
```

Сообщение об ошибке *Destination Unreachable*

ICMP-сообщение типа `Destination Unreachable` (узел назначения недоступен) представляет собой сообщение об ошибке. В заголовке пакета данного типа содержится код, обозначающий ошибку, которая имела место.

Следующие правила разрешают прохождение входящих и исходящих ICMP-сообщений `Destination Unreachable`:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 3 -d
$IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 3 -d
$IPADDR -j ACCEPT
```

С точки зрения безопасности это достаточно неоднозначный пакет, поскольку можно использовать такого типа сообщения для сбора информации об адресах узлов и портах. Кроме того, сообщения `Destination Unreachable` могут быть использованы для организации атаки с целью вывода узла из строя.

Тем не менее, подтип `Fragmentation Needed` сообщения `Destination Unreachable` необходим для нормальной работы сетевых средств. С его помощью взаимодействующие узлы договариваются об особенностях разбиения передаваемых сетевых пакетов на фрагменты.

Также, если требуется, чтобы компьютеры вашей локальной сети отвечали на входящие запросы программы traceroute, необходимо разрешить передачу исходящих пакетов, содержащих подтип `Port Unreachable` сообщения `Destination Unreachable`.

Сообщение *Time Exceeded*

ICMP-сообщение типа `Time Exceeded` (превышение времени) передает сведения о том, что число узлов, через которые проходил сетевой пакет по пути его следования, превысило максимально допустимое значение. В настоящее время сообщение `Time Exceeded` обычно передается в ответ на UDP-запрос программы traceroute.

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 11 -d
$IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $IPADDR 11 -d
$MY_ISP -j ACCEPT
```

Если требуется, чтобы ваша система отвечала на входящие запросы traceroute, необходимо разрешить передачу исходящих ICMP-сообщений `Time Exceeded`. Приведенные ранее правила допускают обращения traceroute лишь с компьютера провайдера. Если вы хотите использовать traceroute на локальном узле, вы должны разрешить входящие сообщения `Time Exceeded`. Так как описываемая конфигурация брандмауэра не является маршрутизатором общего назначения, сообщения `Time Exceeded` используются только с описанной ранее целью.

Программа ping: сообщения *Echo Request* и *Echo Reply*

Программа ping использует два типа ICMP-сообщений: `Echo Request` (эхо-запрос) и `Echo Reply` (эхо-ответ). Программа ping применяется для проверки связи с конкретными узлами сети.

Следующие два правила дают возможность передавать пакеты ping по любому адресу:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $IPADDR 11 -d
$MY_ISP -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $ANYWHERE 11-d
$IPADDR -j ACCEPT
```

Приведенные ниже правила позволяют принимать пакеты ping только с определенных узлов, а конкретно — из сети вашего провайдера:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -s $MY_ISP 8 -d
$IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -s $IPADDR 0 -d
$MY_ISP -j ACCEPT
```

В данном примере набор внешних узлов, которым разрешена передача вашей системе пакетов ping, ограничен компьютерами провайдера. Сделано это для того, чтобы администратор провайдера мог в любой момент проверить, как происходит обмен данными с внешним интерфейсом вашего компьютера. Прием ping с остальных хостов запрещен.

Противодействие smurf-атакам

При организации атаки типа smurf пакеты ping, содержащие сообщения `Echo Request`, передаются в широковещательном режиме. Исходный IP-адрес в составе пакета подменяется IP-адресом "жертвы" — IP-адресом того узла, против которого направлена атака. В результате все узлы сети, получившие сообщения `Echo Request`, передают ответы по адресу "жертвы", загружая линии связи ICMP-пакетами. В результате, если у вас наружный канал не очень широкий — вы лишаетесь доступа в Интернет.

Приведенные далее правила предназначены для протоколирования попыток smurf-атаки. Поскольку прохождение широковещательных ICMP-пакетов явно не разрешено ни одним из правил, эти пакеты будут удалены по умолчанию. Обратите внимание, что в правилах указаны не только `Echo Request`, но и другие типы сообщений. Дело в том, что возможности для атаки не ограничиваются сетевыми пакетами ping.

```
# Противодействие smurf-атаке
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -d $BROADCAST_DEST -j
DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -d $BROADCAST_DEST -j
REJECT -l

# Маска сети
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp -d $NETMASK -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -d $NETMASK -j REJECT -l

# Адрес сети
ipchains -A input -i $EXTERNAL_INTERFACE  -p icmp -d $NETWORK -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp -d $NETWORK -j REJECT -l
```

Разрешение функционирования служб

Ранее мы определили правила, позволяющие отклонять сетевые пакеты с сомнительными адресами, а также разрешили локальному компьютеру работать через интерфейс обратной петли. В результате мы получили нормально функционирующий локальный компьютер с полностью отсутствующим дос-

тупом в Интернет. Наша дальнейшая задача — обеспечить нормальное функционирование локального компьютера (сети) в Интернете. Для того чтобы ваш компьютер мог принимать и отправлять почту, работать по FTP, HTTP и т. п., необходимо разрешить прохождение сетевых пакетов с определенными портами. На первый взгляд — задача объемная, впрочем, необходимо обеспечить прохождение пакетов всего от десятка служб, что не так уж и много.

Служба DNS

Служба DNS использует в работе порт 53 и протоколы UDP и TCP. Соединение может устанавливаться как между клиентом и сервером, так и между двумя серверами. Для разрешения взаимодействия между клиентом и сервером необходимо добавить следующее правило:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p udp -s $IPADDR
$UNPRIVPORTS -d $NAMESERVER 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp -s $IPADDR
$UNPRIVPORTS -d $NAMESERVER 53 -j ACCEPT
```

В том случае, если ответ сервера не помещается в одной UDP-датаграмме, между клиентом и сервером устанавливается TCP-соединение. Обычно это происходит при передаче данных зоны между первичным и вторичным DNS-серверами. Для этого случая необходимо в цепочку правил добавить следующее правило:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNPRIVPORTS -d $NAMESERVER 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNPRIVPORTS -d $NAMESERVER 53 -j ACCEPT
```

В том случае, если у вас есть локальный DNS-сервер, и вы предоставляете его услуги каким-либо клиентам (например, компьютерам вашей локальной сети), желательно ограничить конкретным списком компьютеров доступ к вашему локальному DNS-серверу. Для этого воспользуйтесь следующими правилами:

```
# Разрешение обмена между клиентом и DNS-сервером
ipchains -A input -i $EXTERNAL_INTERFACE -p udp -s <clients.addr>
$UNPRIVPORTS -d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp -s <clients.addr>
$UNPRIVPORTS -d $IPADDR 53 -j ACCEPT

# Разрешение обмена между DNS-серверами
ipchains -A input -i $EXTERNAL_INTERFACE -p udp -s <clients.addr> 53 -d
$IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp -s <clients.addr> 53 -d
$IPADDR -j ACCEPT
```

Следующие правила применяются тогда, когда необходима передача с использованием протокола TCP:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s <dns.sec>
$UNIPRIVPORTS -d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 53 -d
<dns.sec> $UNIPRIVPORTS -j ACCEPT
```

Email

Для приема и пересылки электронной почты используются следующие протоколы:

- SMTP порт 25 TCP;
- POP3 порт 110 TCP;
- IMAP порт 143 TCP.

При создании правил, разрешающих функционирование SMTP-протокола, будем считать, что наша почта отправляется через провайдера.

Для передачи почты по SMTP-протоколу на почтовый сервер провайдера необходимо добавить следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $SMTP_GATEWAY 25 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$SMTP_GATEWAY 25 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

В том случае, если у вас в локальной сети присутствует свой собственный SMTP-сервер, правила фильтрации несколько изменяются и принимают такой вид:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $ANYWHERE 25 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$ANYWHERE 25 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

Для получения электронной почты используется протокол POP3 или протокол IMAP. Для нормального функционирования POP3-протокола необходимо для нашего брандмауэра добавить следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $POP_SERVER 110 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$POP_SERVER 110 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

Если вы хотите предоставить некоторым внешним хостам доступ к вашему POP3-серверу — используйте следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s <pop.clients>
$UNIPRIVPORTS -d $IPADDR 110 -j ACCEPT
```

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 110 -d
<pop.clients> $UNIPRIVPORTS -j ACCEPT
```

В том случае, если у вас используется IMAP-протокол, добавьте следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $IMAP_SERVER 143 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$IMAP_SERVER 143 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

Если вы хотите предоставить некоторым внешним хостам доступ к вашему IMAP-серверу — используйте следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s <pop.clients>
$UNIPRIVPORTS -d $IPADDR 143 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 143 -d
<pop.clients> $UNIPRIVPORTS -j ACCEPT
```

NNTP

Сервер новостей использует порт 119 и протокол TCP. Для обеспечения нормального функционирования сервера новостей необходимо добавить три набора правил.

Если вы используете сервер новостей вашего провайдера, то для получения и отправки статей в группы новостей необходимо добавить следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s
$IPADDR $UNIPRIVPORTS -d $NEWS_SERVER 119 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$NEWS_SERVER 119 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

В том случае, если у вас в локальной сети есть свой собственный сервер новостей, и вы хотите разрешить извне доступ определенным хостам — воспользуйтесь следующими правилами:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s <ip.clients>
$UNIPRIVPORTS -d $NEWS_SERVER 119 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$NEWS_SERVER 119 -d <ip.clients> $UNIPRIVPORTS -j ACCEPT
```

А поскольку вашему локальному серверу новостей необходимо получать и передавать статьи от сервера новостей провайдера — воспользуйтесь следующими правилами:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $NEWS_SERVER 119 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$NEWS_SERVER 119 -d $IPADDR $UNIPRIVPORTS -j ACCEPT
```

Telnet

Telnet использует порт 23 TCP. Применение протокола удаленного доступа Telnet было очень популярно еще три-четыре года назад, однако из-за того, что этот протокол абсолютно не защищен, а также вследствие появления альтернативы в виде протокола SSH — категорически не рекомендуется разрешать доступ извне по данному протоколу.

SSH

Протокол SSH использует порт 22 и TCP. Защищенная замена Telnet и r-командам. При функционировании использует привилегированные порты 513—1023.

Чтобы применять протокол SSH для доступа из локальной сети к Интернету, SSH-серверам необходимо ввести следующие правила:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNIPRIVPORTS -d $ANYWHERE 22 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s $ANYWHERE 22 -d
$IPADDR $UNIPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $ANYWHERE
$SSH_PORTS -d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$IPADDR 22 -d $ANYWHERE $SSH_PORTS -j ACCEPT
```

Следующие правила разрешают доступ удаленным клиентам к вашим локальным SSH-серверам:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $ANYWERE
$UNIPRIVPORTS -d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 22 -d
$ANYWHERE $UNIPRIVPORTS -j ACCEPT
ipchains -A -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR $SSH_PORTS -d
$ANYWERE 22 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s $ANYWHERE 22 -d
$IPADDR $SSH_PORTS -j ACCEPT
```

FTP

Пожалуй, один из сложных протоколов, поскольку использует несколько портов (TCP 21, 20). Протокол предусматривает два режима передачи — активный и пассивный канал передачи, что несколько осложняет конфигурацию брандмауэра.

Следующие правила разрешают доступ к удаленным FTP-серверам:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNPRIVPORTS -d $ANYWERE 21 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s $ANYWHERE 21 -d
$IPADDR $UNPRIVPORTS -j ACCEPT
```

Следующие правила разрешают устанавливать соединения в режиме активного канала передачи данных:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s $ANYWHERE 20 -d
$IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNPRIVPORTS -d $ANYWERE 20 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR
$UNPRIVPORTS -d $ANYWERE $UNPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$ANYWHERE $UNPRIVPORTS -d $IPADDR $UNPRIVPORTS -j ACCEPT
```

Для того чтобы к вашему локальному FTP-серверу был разрешен доступ извне, необходимо ввести следующие правила:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $ ANYWERE
$UNPRIVPORTS -d $IPADDR 21 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 21 -d
$ANYWERE $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR 20 -d
$ANYWERE $UNPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y -s
$ANYWHERE $UNPRIVPORTS -d $IPADDR 20 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $ANYWHERE
$UNPRIVPORTS -d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR
$UNPRIVPORTS -d $ANYWHERE $UNPRIVPORTS -j ACCEPT
```

HTTP

Протокол HTTP использует порт 80 TCP. Для того чтобы локальные клиенты могли получить доступ к Web-серверам Интернета, необходимо ввести следующие правила:

```
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp -s $IPADDR $UNPRIVPORT -
d $ANYWHERE 80 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp !-y -s $ANYWHERE 80 -d
$IPADDR $UNIPRIVPORTS -j ACCEPT
```

В том случае, если у вас в локальной сети есть свой собственный Web-сервер, и вы хотите разрешить извне доступ, воспользуйтесь следующими правилами:

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp -s $ANYWHERE
$UNIPRIVPORTS -d $IPADDR 80 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y -s $IPADDR 80 -d
$ANYWHERE $UNIPRIVPORTS -j ACCEPT
```

Помимо перечисленных сервисов у вас могут применяться и другие, однако, зная протокол, используемые порты и опираясь на ранее приведенные пра-

вила, не составит труда добавить соответствующие правила для нормального функционирования ваших сервисов.

Запрет доступа с "неблагонадежных" узлов

Если вы обнаружите попытки сканирования портов или другие сомнительные действия, которые периодически предпринимаются с одного и того же хоста, желательно вовсе запретить обращение к системе с этого адреса.

Пример запрещающего правила приведен далее:

```
ipchains -I input  -i $EXTERNAL_INTERFACE  -s <адрес/маска> -j DENY
```

Согласно этому правилу удаляется любой пакет, независимо от протокола и номера исходного порта или порта назначения.

Поддержка обмена в локальной сети

Для поддержки локальной сети, стоящей за брандмауэром, следует добавить некоторые правила. Эти правила необходимы, для того чтобы разрешить доступ к внутреннему сетевому интерфейсу брандмауэра и направить трафик в глобальную сеть. Как только на компьютере-брандмауэре будет реализована поддержка двух или более интерфейсов, он превратится в бастион.

Разрешение доступа к внутреннему сетевому интерфейсу брандмауэра

При работе с небольшими сетями вряд ли есть основание ограничивать доступ к брандмауэру из локальной сети. Следующие правила разрешают все виды взаимодействия между брандмауэром и локальной сетью:

```
ipchains -A input -i $INTERNAL_INTERFACE -s LAN -j ACCEPT
ipchains -A output -i $Internal_INTERFACE -s LAN_1 -j ACCEPT
```

Обратите внимание, что данные правила разрешают обмен лишь с брандмауэром. Доступ к Интернету отсутствует, поскольку по умолчанию компьютер, выполняющий роль брандмауэра, не производит динамической маршрутизации пакетов и не поддерживает статических маршрутов. Чтобы обеспечить маршрутизацию пакетов через брандмауэр, надо задать дополнительные правила.

Выбор конфигурации для пользующейся доверием локальной сети

Пакеты, передаваемые компьютерами локальной сети, можно условно разбить на две категории. Первая категория представляет собой данные, которыми локальные узлы обмениваются с брандмауэром. Вторая категория —

это данные, направляемые через внешний интерфейс брандмауэра в Интернет.

При работе небольшой сети вряд ли может возникнуть потребность в фильтрации пакетов, проходящих через внутренний интерфейс брандмауэра, однако некоторая обработка все-таки необходима. Речь идет о маскировке.

Если компьютер, выполняющий роль брандмауэра, имеет реальный IP-адрес, а всем остальным машинам, подключенным к локальной сети, присвоены адреса, предназначенные для внутреннего использования, то для обеспечения доступа локальных машин в Интернет брандмауэр должен взять на себя функции Proxy-сервера.

По сути, компьютер, выполняющий маскировку пакетов, представляет собой низкоуровневый Proxy-сервер. Такой сервер обслуживает клиентов, устанавливая соединения с удаленными узлами от своего имени. Поскольку исходный адрес в пакете, передаваемом в Интернет, заменяется исходным адресом компьютера, выполняющего маскировку, то с точки зрения удаленного узла обмен данными с ним проводит Proxy-сервер. В пакетах, передаваемых удаленным узлом, адрес назначения заменяется на адрес локального компьютера.

Организация доступа из локальной сети к брандмауэру бастионного типа

Если вы занимаетесь администрированием небольшой сети, то, скорее всего, вы не захотите ограничивать доступ локальных компьютеров к брандмауэру бастионного типа. Следующее правило организует неограниченный доступ к внутреннему сетевому интерфейсу брандмауэра:

```
ipchains -A input -i $INTERNAL_INTERFACE -S LAN_1 -j ACCEPT ipchains -A
output -i $INTERNAL_INTERFACE -d LAN_1 -j ACCEPT
```

Перенаправление трафика

Если несколько локальных сетей должны обмениваться информацией, вам необходимо разрешить передачу пакетов между соответствующими интерфейсами. Конечно, делать это надо лишь в том случае, если маршрутизация не выполняется какими-либо другими средствами.

Чтобы брандмауэр можно было использовать в качестве маршрутизатора, объединяющего две локальные сети, необходимо добавить следующие правила:

```
#Приведенные правила разрешают доступ к брандмауэру
ipchains -A input -i $LAN_INTERFACE_1 -s LAN_1 -j ACCEPT ipchains -A
output -i $LAN_INTERFACE_1 -d LAN_1 -j ACCEPT
```

```
ipchains -A input -i $LAN_INTERFACE_2 -s LAN_2 -j ACCEPT ipchains -A
output -i $LAN_INTERFACE_2 -d LAN_2 -j ACCEPT
```

Следующие правила обеспечивают передачу трафика между локальными сетями в двух направлениях без выполнения маскировки.

```
ipchains -A forward -i $LAN_INTERFACE_2 -s LAN_1 -d LAN_2 -j ACCEPT
ipchains -A forward -i $LAN_INTERFACE_1 -s LAN_2 -d LAN_1 -j ACCEPT
```

Разрешение доступа в Интернет из локальной сети: IP-перенаправление и маскировка

На данном этапе на обмен данными между машинами локальной сети и внутренним интерфейсом брандмауэра не накладывается никаких ограничений. Однако локальные компьютеры не имеют доступа к Интернету. Для обеспечения такого доступа необходимо реализовать перенаправление и маскировку пакетов.

Механизм перенаправления реализуется на уровне ядра системы и позволяет компьютеру под управлением Linux выступать в роли маршрутизатора, перенаправляя трафик из одной сети в другую. Однако, даже если IP-перенаправление будет реализовано путем выбора конфигурации сети, пакеты не будут передаваться между интерфейсами до тех пор, пока не будут созданы правила, разрешающие такую передачу.

Перенаправления пакетов, адресованных различным узлам глобальной сети, не всегда достаточно для нормального взаимодействия локальных компьютеров с Интернетом. Если компьютерам локальной сети присвоены IP-адреса классов A, B и C, предназначенные для внутреннего использования, необходимо выполнить их маскировку, т. е. заменить исходящий адрес локального узла в пакете IP-адресом внешнего интерфейса брандмауэра. Эта возможность также реализована на уровне ядра операционной системы. Но даже если компьютеры локальной сети имеют обычные IP-адреса, допустимые для использования в Интернете, маскировка остается одним из самых эффективных средств защиты внутренней сети.

Несмотря на то, что перенаправление и маскировка — это совершенно различные механизмы, на уровне программы ipchains они представлены как одна процедура. Пакеты, поступившие на внутренний интерфейс брандмауэра, передаются на его внешний интерфейс. Перед тем как пакет будет помещен в очередь внешнего интерфейса, средства маскировки заменяют адрес источника IP-адресом внешнего интерфейса брандмауэра. Наличие средств перенаправления и маскировки превращает брандмауэр в Proxy-сервер с возможностями фильтрации.

Приведенное далее правило позволяет перенаправить трафик с внутреннего интерфейса на внешний, попутно выполняя маскировку пакетов:

```
ipchains -A forward -I $EXTERNAL_INTERFACE -s LAN_1 -j MASQ
```

Действия ACCEPT и DENY, указанные в цепочке output внешнего интерфейса, производятся после того, как перенаправление будет выполнено. Таким образом, несмотря на то, что передача от внутреннего к внешнему интерфейсу разрешена для всех пакетов, в Интернет попадут лишь те из них, для которых существуют разрешающие правила, связанные с внешним интерфейсом.

Правила маскировки позволяют задавать адреса источника и назначения, а также номера портов.

При перенаправлении трафик передается между сетевыми интерфейсами без изменений. Если компьютеры внутренней сети имеют IP-адреса, допустимые в Интернете, и брандмауэр выполняет перенаправление графика, то с точки зрения стороннего наблюдателя между локальной машиной и хостом Интернета устанавливается непосредственное соединение. При обращении удаленного компьютера к локальному узлу пакеты перенаправляются в локальную сеть.

При наличии маскировки передача трафика перестает быть симметричной. В этом случае разрешены лишь обращения из локальной сети к внешним серверам. При передаче пакета в Интернет исходный адрес, принадлежащий локальному компьютеру, заменяется IP-адресом внешнего интерфейса брандмауэра. При получении ответа от сервера производится обратное преобразование пакета: IP-адрес брандмауэра заменяется адресом локального компьютера, которому адресован пакет.

Как перенаправление, так и маскировка пакетов выполняются на уровне ядра операционной системы, поэтому оно должно быть скомпилировано с поддержкой маскировки и перенаправления пакетов.

Разрешить маскировку можно с помощью программы ipchains. Для того чтобы система выполняла маскировку всего трафика, направленного из локальной сети к удаленным узлам, необходимо задать следующее правило:

```
ipchains -A forward -i $EXTERNAL_INTERFACE \ -s LAN_1 -j MASQ
```

Независимо от того, выделены ли для компьютеров локальной сети допустимые IP-адреса или им присвоены адреса, предназначенные для внутреннего использования, при настройке брандмауэра рекомендуется отказаться от прямого перенаправления пакетов и использовать маскировку. Маскировка локальных сетей — мощное средство защиты. При использовании маскировки хосты Интернета не могут обращаться к компьютерам вашей локальной сети. Более того, локальные машины не видны извне. С точки зрения Интернета вся ваша локальная сеть состоит из одного хоста — компьютера, на котором реализован брандмауэр.

Дополнительной мерой защиты могут стать Proxy-фильтры прикладного уровня, такие как SOCKS. И в этом случае при обмене с удаленным узлом создается впечатление, что запросы генерируются брандмауэром. Преиму-

щество фильтров прикладного уровня также состоит в том, что с их помощью можно организовать специальную обработку трафика, учитывающую специфику обмена с конкретными службами.

Организация демилитаризованной зоны

Конфигурация брандмауэра, описанная в начале главы, вполне подходит для защиты одного компьютера от нежелательных воздействий извне. Брандмауэр с двумя сетевыми интерфейсами может использоваться для защиты локальной сети. Брандмауэр бастионного типа защищает локальную сеть до тех пор, пока система, на которой установлен брандмауэр, не будет взломана. Даже если в процессе фильтрации участвует не только внешний, но и внутренний интерфейс, это не спасет систему. Если злоумышленнику удастся взломать компьютер, выполняющий роль брандмауэра, то ваша локальная сеть остается беззащитной перед лицом взломщика. Поэтому брандмауэр бастионного типа представляет собой единственную линию обороны. Такой тип защиты распространен в небольших организациях.

В средних и крупных организациях применение брандмауэра бастионного типа является недостаточной мерой. В таких сетях обычно используются Proxy-серверы либо система из двух брандмауэров, между которыми располагается демилитаризованная зона, или граничная сеть. Внешний интерфейс первого брандмауэра используется для соединения с Интернетом, а внутренний принадлежит демилитаризованной зоне, как правило, использующей свою локальную сеть. Второй брандмауэр, который обычно называется *заглушкой* (choke), также имеет два интерфейса. Внешний интерфейс подключен к демилитаризованной зоне, а внутренний — к внутренней сети предприятия. Обычно в демилитаризованной зоне размещаются серверы, которые должны быть доступны из Интернета. Для реализации описанной архитектуры требуется намного больше компьютеров и большая численность обслуживающего персонала, чем в случае применения брандмауэра бастионного типа.

Защита подсетей с помощью брандмауэров

Для организации демилитаризованной зоны обычно используется один из двух способов. Первый способ предполагает применение брандмауэра бастионного типа с тремя сетевыми интерфейсами. Один сетевой интерфейс подключается к Интернету, а два остальных — к двум изолированным локальным сетям. Одна из сетей выполняет роль демилитаризованной зоны, в ней размещаются общедоступные серверы. Во второй сети располагаются службы, предназначенные для внутреннего использования, и компьютеры пользователей.

Другой способ состоит в использовании второго брандмауэра, называемого заглушкой (choke). Компьютер, на котором реализован брандмауэр-заглуш-

ка, выполняет роль шлюза между демилитаризованной зоной и локальной сетью. Внутренний сетевой интерфейс брандмауэра-заглушки подключен к локальной сети, а внешний — к демилитаризованной зоне. Данные, передаваемые компьютерами локальной сети, маскируются. Таким образом, с точки зрения брандмауэра-бастиона и машин, принадлежащих демилитаризованной зоне, вся внутренняя сеть представлена одним брандмауэром-заглушкой.

Бастион маскирует трафик внутренней сети, поэтому, на первый взгляд, брандмауэр-заглушка не должен выполнять маскировку. Однако, если вся внутренняя сеть имеет один адрес, принадлежащий брандмауэру-заглушке, набор правил бастиона упрощается.

Подобная структура реализует две линии обороны локальной сети. Локальная сеть расположена за заглушкой и полностью изолирована от бастиона и, тем более, от Интернета. При использовании описанной системы не обязательно задавать полный набор правил для внутреннего интерфейса бастиона, достаточно, если правила, осуществляющие фильтрацию пакетов, будут связаны с внешним интерфейсом заглушки.

Таким образом, система защиты внутренней сети содержит как минимум четыре набора правил — по одному для внутреннего и внешнего интерфейса каждого из брандмауэров. Правила для внешнего интерфейса бастиона практически совпадают с правилами брандмауэра, описанного ранее.

Реально описанная здесь система отличается от ранее рассмотренной наличием демилитаризованной зоны, а также новыми правилами, заданными для внутреннего интерфейса бастиона и для внешнего интерфейса заглушки. Указанные два набора правил, по сути, представляют собой зеркальное отражение друг друга.

Отладка брандмауэра

Брандмауэр установлен, настроен и активизирован, но функционирует не так, как хотелось. Даже если брандмауэр работает, рекомендуется сразу после установки и настройки произвести проверку правильности функционирования брандмауэра.

Общие рекомендации по отладке брандмауэра

Приведем рекомендации, позволяющие облегчить отладку брандмауэра.

- Перед запуском сценария убедитесь, что в первой строке находится команда удаления существующих правил, а следующая команда устанавливает политику по умолчанию.

- Производите отладку с текстовой консоли. Не производите отладку брандмауэра с удаленной машины. В случае обрыва связи или неправильного конфигурирования вы рискуете остаться без доступа в Интернет.
- По возможности производите добавление правил по одному. В этом случае гораздо проще выявить причину неисправности. Сразу после добавления правил рекомендуется проверить их работоспособность.
- Обработка сетевого пакета определяется первым правилом, которому удовлетворяет этот сетевой пакет, поэтому порядок следования правил имеет большое значение.
- Помните, что существуют как минимум две не зависящие друг от друга цепочки: input и output. Если правила, содержащиеся в одной цепочке, обрабатывают пакет корректно, причина неисправности, очевидно, находится в другой цепочке.
- Если сценарий "зависает", возможно, что правило, в котором содержится доменное имя узла, вступает в действие раньше, чем правило, разрешающее доступ к DNS. Если какое-либо правило предшествует правилам, определяющим взаимодействие с DNS, в нем должны быть указаны IP-адреса. Использование доменных имен в таких правилах недопустимо, поскольку у вас еще нет доступа к серверу DNS.
- Проверяйте синтаксис команд программы ipchains. При составлении правил легко перепутать адрес или порт источника с адресом или портом назначения либо неверно задать регистр опции.
- При наличии синтаксической ошибки выполнение сценария брандмауэра завершается, и последующие правила не устанавливаются. Чтобы определить неверно составленное правило, запускайте сценарий с опциями `-x` или `-v`. Если указана опция `-v`, строки сценария выводятся в тот момент, когда они читаются интерпретатором команд. Опция `-x` задает вывод строк по мере выполнения команд оболочкой.
- Если какой-либо из серверов не работает, включите протоколирование удаляемых пакетов, указав опцию `-l` программы ipchains. Проанализируйте записи в файле /var/log/messages.
- Если вы обмениваетесь данными с Интернетом, работая на компьютере-брандмауэре, но не можете сделать этого с узла локальной сети, проверьте установки в /etc/sysconfig/network, связанные с перенаправлением пакетов.
- Если сервер доступен в пределах локальной сети, но попытка обратиться к нему извне оканчивается неудачей, включите протоколирование пакетов, проходящих через внутренний интерфейс. Постарайтесь выполнить всю проверку как можно быстрее, в противном случае в файле /var/log /messages появятся сотни записей.

- Если одна из служб не работает, временно включите в начало сценария брандмауэра правила, разрешающие прохождение пакетов в обоих направлениях, и задайте протоколирование, указав опцию `-l`. Проверьте, доступен ли сервер. Если это так, просмотрите записи в файле /var/log/messages и определите, какие порты используются при его работе.

Отображение списка правил брандмауэра

Чтобы убедиться, что правила брандмауэра инсталлированы именно так, как вы это планировали при составлении сценария, можно вывести содержимое цепочек. Сделать это позволяет опция `-L` программы ipchains. Если опция `-L` задана, ipchains выводит содержащиеся в соответствующей таблице ядра правила в той последовательности, в которой они применяются при обработке пакета. Для вывода содержимого цепочек используются следующие команды:

```
ipchains -L input
ipchains -L output
ipchains -L forward
```

Различные опции программы ipchains позволяют выводить содержимое одной и той же цепочки с различной степенью детализации. Форматы вывода для цепочек input, output и forward совпадают.

Утилиты

В состав пакета ipchains входит утилита ipchains-save, с помощью которой вы можете получить текущую конфигурацию брандмауэра на стандартный вывод, а затем перенаправить его в файл. Так же есть утилита-близнец ipchains-restore, которая может получать информацию из стандартного ввода.

Iptables

Iptables — это логическое развитие ipchains. Как обычно, взяли все лучшее, модифицировали, развили в сторону гибкости, надежности и производительности. В современных дистрибутивах iptables является неотъемлемой частью системы и ядро скомпилировано с учетом требований iptables. В том случае, если вы предпочитаете лично пересобирать ядро операционной системы, при его конфигурации необходимо сделать следующие настройки:

- `CONFIG_PACKET` — эта опция необходима для приложений, работающих непосредственно с сетевыми устройствами, например, tcpdump или snort;
- `CONFIG_NETFILTER` — эта опция необходима, если вы собираетесь использовать компьютер в качестве сетевого экрана или шлюза;

- CONFIG_IP_NF_CONNTRACK — трассировка соединений (среди всего прочего, используется при трансляции сетевых адресов и маскарадинге (masquerading));
- CONFIG_IP_NF_FTP — трассировка FTP-соединений;
- CONFIG_IP_NF_IPTABLES — эта опция необходима для выполнения операций фильтрации, преобразования сетевых адресов (NAT) и маскарадинга;
- CONFIG_IP_NF_MATCH_LIMIT — этот модуль предоставляет возможность ограничения количества проверок для некоторого правила. Например, `-m limit --limit 3/minute` указывает, что заданное правило может пропустить не более 3-х пакетов в минуту. Таким образом, данный модуль может использоваться для защиты от нападений типа "Отказ в обслуживании";
- CONFIG_IP_NF_MATCH_MAC — этот модуль позволяет строить правила, основанные на MAC-адресации;
- CONFIG_IP_NF_MATCH_MARK — функция маркировки пакетов MARK. При использовании функции MARK можно пометить требуемые пакеты, а затем, в других таблицах, в зависимости от значения метки, принимать решение о маршрутизации помеченного пакета;
- CONFIG_IP_NF_MATCH_MULTIPORT — этот модуль позволяет строить правила с проверкой на принадлежность пакета к диапазону номеров портов источника/приемника;
- CONFIG_IP_NF_MATCH_TOS — этот модуль позволяет строить правила, отталкиваясь от состояния поля TOS в пакете. Поле TOS устанавливается для Type Of Service;
- CONFIG_IP_NF_MATCH_TCPMSS — эта опция добавляет возможность проверки поля MSS в TCP-пакетах;
- CONFIG_IP_NF_MATCH_STATE — это самое серьезное усовершенствование по сравнению с ipchains. Данный модуль предоставляет возможность управления TCP-пакетами, основываясь на их состоянии (state);
- CONFIG_IP_NF_MATCH_UNCLEAN — этот модуль реализует возможность дополнительной проверки IP-, TCP-, UDP- и ICMP-пакетов на предмет наличия в них несоответствий и ошибок;
- CONFIG_IP_NF_MATCH_OWNER — проверка "владельца" соединения (socket). Для примера, мы можем позволить только пользователю root выходить в Интернет;
- CONFIG_IP_NF_FILTER — реализация таблицы filter, в которой в основном и осуществляется фильтрация. В данной таблице находятся цепочки input, forward и output;

- CONFIG_IP_NF_TARGET_REJECT — добавляется действие REJECT, которое производит передачу ICMP-сообщения об ошибке в ответ на входящий пакет, который отвергается заданным правилом;
- CONFIG_IP_NF_TARGET_MIRROR — возможность отправки полученного пакета обратно;
- CONFIG_IP_NF_NAT — трансляция сетевых адресов. С помощью этой опции вы сможете дать выход в Интернет всем компьютерам вашей локальной сети, имея лишь один уникальный IP-адрес;
- CONFIG_IP_NF_TARGET_MASQUERADE — маскарадинг. В отличие от NAT, маскарадинг используется в тех случаях, когда заранее неизвестен наш IP-адрес в Интернете. Маскарадинг дает несколько более высокую нагрузку на компьютер, по сравнению с NAT, однако он работает в ситуациях, когда невозможно заранее указать собственный внешний IP-адрес.
- CONFIG_IP_NF_TARGET_REDIRECT — перенаправление. Вместо того чтобы просто пропустить пакет дальше, это действие перенаправляет пакет на другой порт сетевого экрана;
- CONFIG_IP_NF_TARGET_LOG — используется для фиксации отдельных пакетов в системном журнале (syslog);
- CONFIG_IP_NF_TARGET_TCPMSS — эта опция может использоваться для преодоления ограничений, накладываемых некоторыми провайдерами, которые блокируют ICMP-пакеты Fragmentation Needed;
- CONFIG_IP_NF_COMPAT_IPCHAINS — добавляет совместимость с ipchains;
- CONFIG_IP_NF_COMPAT_IPFWADM — добавляет совместимость с ipfwadm.

Порядок движения транзитных пакетов

В табл. 16.4 приведен порядок движения транзитных пакетов, соответствующие таблицы фильтрации и описание.

Таблица 16.4. *Прохождение транзитных пакетов*

Таблица	Цепочка	Примечание
		Входящий сетевой интерфейс
mangle	prerouting	Цепочка используется для внесения изменений в заголовок пакета
nat	prerouting	Цепочка используется для трансляции сетевых адресов. Любого рода фильтрация в этой цепочке может производиться только в исключительных случаях
		В этой точке решается куда пойдет пакет: локальному приложению или на другой хост

Таблица 16.4 *(окончание)*

Таблица	Цепочка	Примечание
mangle	forward	Цепочка forward таблицы mangle должна использоваться только в исключительных случаях, когда необходимо внести некоторые изменения в заголовок пакета между двумя точками принятия решения о маршрутизации
filter	forward	В цепочку forward попадают только те пакеты, которые идут на другой хост. Вся фильтрация транзитного трафика должна выполняться здесь. Через эту цепочку проходит трафик в обоих направлениях, обязательно учитывайте это обстоятельство при написании правил фильтрации
mangle	postrouting	Эта цепочка предназначена для внесения изменений в заголовок пакета после того, как принято последнее решение о маршрутизации
nat	postrouting	Эта цепочка предназначена в первую очередь для NAT и маскарадинга
		Исходящий сетевой интерфейс

Порядок движения пакетов для локальной программы

Существует определенный порядок движения пакетов для локальной программы. Он приведен в табл. 16.5.

Таблица 16.5. *Движение пакетов для локальных программ*

Таблица	Цепочка	Примечание
		Входной сетевой интерфейс
mangle	prerouting	Используется для внесения изменений в заголовок пакета
nat	prerouting	Преобразование адресов. Фильтрация пакетов здесь допускается только в исключительных случаях
		Принятие решения о маршрутизации
mangle	input	Здесь вносятся изменения в заголовок пакета перед тем, как он будет передан локальной программе
filter	input	Производится фильтрация входящего трафика
		Локальная программа

Порядок движения пакетов от локальной программы

Существует определенный порядок движения пакетов от локальной программы. Он приведен в табл. 16.6.

Таблица 16.6. *Движение пакетов, созданных локальными программами*

Таблица	Цепочка	Примечание
		Локальный процесс
		Принятие решения о маршрутизации
mangle	output	Используется для внесения изменений в заголовок пакета
nat	output	Используется для трансляции сетевых адресов (NAT) в пакетах, исходящих от локальных процессов брандмауэра
filter	output	Фильтруется исходящий трафик
mangle	postrouting	Используется для правил, которые должны вносить изменения в заголовок пакета перед тем, как он покинет брандмауэр, но уже после принятия решения о маршрутизации. В эту цепочку попадают все пакеты, как транзитные, так и созданные локальными процессами брандмауэра
nat	postrouting	Нежелательно производить фильтрацию пакетов во избежание побочных эффектов. Однако и здесь можно останавливать пакеты, применяя политику по умолчанию DROP
		Сетевой интерфейс

Таблица mangle

Таблица mangle (от англ. mangle — *изменять*) предназначена для внесения изменений в заголовки пакетов.

В таблице допускается выполнять только следующие действия:

- TOS
- TTL
- MARK

Действие TOS выполняет установку битов поля `Type of Service` в пакете. Это поле используется для назначения сетевой политики обслуживания пакета.

Действие TTL используется для установки значения поля `TTL` (Time To Live) пакета.

Действие MARK устанавливает специальную метку на пакет, которая затем может быть проверена другими правилами в iptables или другими программами. С помощью "меток" можно управлять маршрутизацией пакетов, ограничивать трафик и т. п.

Таблица nat

Таблица используется для выполнения преобразований сетевых адресов (Network Address Translation, NAT). Для этой таблицы применяются следующие действия:

- DNAT (Destination Network Address Translation) — производит преобразование адресов назначения в заголовках пакетов (перенаправление пакетов);
- SNAT (Source Network Address Translation) — используется для изменения исходных адресов пакетов. С помощью этого действия можно скрыть структуру локальной сети, разделить единственный внешний IP-адрес между компьютерами локальной сети для выхода в Интернет;
- MASQUERADE (маскировка) — применяется в тех же целях, что и SNAT, но в отличие от последнего, MASQUERADE дает более сильную нагрузку на систему. Происходит это потому, что каждый раз, когда требуется выполнение этого действия, производится запрос IP-адреса для указанного в действии сетевого интерфейса, в то время как для SNAT IP-адрес указывается непосредственно. Однако, благодаря такому отличию, MASQUERADE может работать в случаях с динамическим IP-адресом.

Таблица filter

В этой таблице должны содержаться наборы правил для выполнения фильтрации пакетов. Пакеты могут пропускаться далее либо отвергаться (действия ACCEPT и DROP соответственно) в зависимости от их содержимого.

Построение правил для iptables

В этом разделе мы рассмотрим порядок построения правил для iptables. В целом, оно мало отличается от построения правил для ipchains.

Каждое правило — это строка, содержащая в себе правила, с помощью которых определяется, подпадает ли пакет под заданное правило, и действие, которое необходимо выполнить в случае выполнения критерия. В общем виде правила записываются так:

```
iptables [-t table] command [match] [target/jump]
```

Если в правило не включается спецификатор [-t table], то по умолчанию предполагается использование таблицы filter, если же предполагается применение другой таблицы, то это требуется указать явно.

Непосредственно за именем таблицы должна стоять команда. Если спецификатора таблицы нет, то команда всегда должна стоять первой. Команда определяет действие iptables.

Раздел `[match]` задает критерии проверки, по которым определяется, подпадает ли пакет под действие этого правила или нет. Здесь можно задать самые разные критерии: IP-адрес источника пакета или сети, IP-адрес места назначения, порт, протокол, сетевой интерфейс и т. п.

И наконец, `[target]` указывает, какое действие должно быть выполнено при условии выполнения критериев в правиле.

Команды ipchains

В табл. 16.7 приведен список команд и правила их использования. Обычно предполагается одно из двух действий: добавление нового правила в цепочку или удаление существующего правила из таблицы.

Таблица 16.7. *Команды iptables*

Команда	Пример использования	Описание
`-A, --append`	`iptables -A INPUT ...`	Добавляет новое правило в конец цепочки
`-D, --delete`	`iptables -D INPUT --dport 80 -j DROP, iptables -D INPUT 1`	Удаление правила из цепочки. Команда имеет два формата записи, первый — когда задается критерий сравнения с опцией `-D`, второй — порядковый номер правила. Если задается критерий сравнения, то удаляется правило, которое имеет в себе этот критерий, если задается номер правила, то будет удалено правило с заданным номером. Счет правил в цепочках начинается с 1
`-R, --replace`	`iptables -R INPUT 1 -s 192.168.0.1 -j DROP`	Команда производит замену одного правила другим
`-I, --insert`	`iptables -I INPUT 1 --dport 80 -j ACCEPT`	Вставляет новое правило в цепочку. Число, следующее за именем цепочки, указывает номер правила, перед которым нужно вставить новое правило

Таблица 16.7 *(окончание)*

Команда	Пример использования	Описание
`-L, --list`	`iptables -L INPUT`	Вывод списка правил в заданной цепочке. Если имя цепочки не указывается, то выводится список правил для всех цепочек
`-F, --flush`	`iptables -F INPUT`	Удаление всех правил из заданной таблицы. Если имя цепочки и таблицы не указывается, то удаляются все правила, во всех цепочках
`-Z, --zero`	`iptables -Z INPUT`	Обнуление всех счетчиков в заданной цепочке. Если имя цепочки не указывается, то подразумеваются все цепочки
`-N, --new-chain`	`iptables -N allowed`	Создается новая цепочка с заданным именем в заданной таблице. Имя цепочки должно быть уникальным и не должно совпадать с зарезервированными именами цепочек и действий
`-X, --delete-chain`	`iptables -X allowed`	Удаление заданной цепочки из заданной таблицы. Удаляемая цепочка не должна иметь правил и не должно быть ссылок из других цепочек на удаляемую цепочку
`-P, --policy`	`iptables -P INPUT DROP`	Задает политику по умолчанию для заданной цепочки. Политика определяет действие, применяемое к пакетам, не попавшим под действие ни одного из правил в цепочке
`-E, --rename-chain`	`iptables -E allowed disallowed`	Команда выполняет переименование пользовательской цепочки

Критерии проверки пакетов

Критерии проверки пакетов можно разделить на пять групп.

- Общие критерии — могут использоваться в любых правилах.
- TCP-критерии — применяются только к TCP-пакетам.
- UDP-критерии — применяются только к UDP-пакетам.

- ICMP-критерии — используются для работы с ICMP-пакетами.
- Специальные критерии: state, owner, limit и пр.

Общие критерии

Общие критерии допустимо употреблять в любых правилах, они не зависят от типа протокола и не требуют модулей расширения. Список общих критериев приведен в табл. 16.8.

Таблица 16.8. *Общие критерии проверки пакетов*

Критерий	Пример использования	Описание
`-p, --protocol`	`iptables -A INPUT -p tcp`	Этот критерий используется для указания типа протокола. Можно применять TCP, UDP и ICMP или ключевое слово `ALL`. Для логической инверсии критерия, перед именем протокола используется символ `!`
`-s, --src, --source`	`iptables -A INPUT -s 192.168.1.1`	IP-адрес источника пакета. Адрес источника может указываться так, как показано в примере, тогда подразумевается единственный IP-адрес. А можно указать адрес в виде *<address>/<mask>*, например как 192.168.0.0/255.255.255.0, или более современным способом 192.168.0.0/24. Символ `!`, установленный перед адресом, означает логическое отрицание
`-d, --dst, --destination`	`iptables -A INPUT -d 192.168.1.1`	IP-адрес получателя. Можно определять как единственный IP-адрес, так и диапазон адресов. Символ `!` используется для логической инверсии критерия
`-i, --in-interface`	`iptables -A INPUT -i eth0`	Интерфейс, с которого был получен пакет. Использование этого критерия допускается только в цепочках input, forward и prerouting. При отсутствии этого критерия предполагается любой интерфейс. Символ `!` инвертирует результат совпадения. Если имя интерфейса завершается символом `+`, то критерий задает все интерфейсы, начинающиеся с заданной строки

Таблица 16.8 *(окончание)*

Критерий	Пример использования	Описание
`-o, --out-interface`	`iptables -A FORWARD -o eth0`	Задает имя выходного интерфейса. Этот критерий допускается использовать только в цепочках output, forward и postrouting. При отсутствии этого критерия предполагается любой интерфейс, что равносильно использованию критерия `-o +`. Символ ! инвертирует результат совпадения. Если имя интерфейса завершается символом +, то критерий задает все интерфейсы, начинающиеся с заданной строки
`-f, --fragment`	`iptables -A INPUT -f`	Правило распространяется на все части фрагментированного пакета, кроме первого, сделано это потому, что нет возможности определить исходящий/входящий порт для фрагмента пакета, а для ICMP-пакетов определить их тип. Допускается использование символа ! для инверсии результата сравнения, только в данном случае символ ! должен предшествовать критерию `-f` (т. е. `! -f`)

TCP-критерии

Этот набор критериев работает только с TCP-пакетами (табл. 16.9). Чтобы использовать их, вам потребуется в правилах указывать тип протокола `--protocol tcp`.

Таблица 16.9. *TCP-критерии проверки пакетов*

Критерий	Пример использования	Описание
`--sport, --source-port`	`iptables -A INPUT -p tcp --sport 22`	Исходный порт, с которого был отправлен пакет. В качестве параметра может указываться номер порта или название сетевой службы. Номера портов могут задаваться в виде интервала из минимального и максимального номеров, например, `--source-port 22:80`. Символ ! используется для инверсии

Таблица 16.9 *(окончание)*

Критерий	Пример использования	Описание
`--dport`, `--destination-port`	`iptables -A INPUT -p tcp --dport 22`	Порт или диапазон портов, на который адресован пакет
`--tcp-flags`	`iptables -p tcp --tcp-flags SYN,FIN,ACK SYN`	Определяет маску и флаги TCP-пакета. Пакет считается удовлетворяющим критерию, если из перечисленных флагов в первом списке в единичное состояние установлены флаги из второго списка. В качестве аргументов критерия могут выступать флаги `SYN`, `ACK`, `FIN`, `RST`, `URG`, `PSH`, а также зарезервированные идентификаторы `ALL` и `NONE`. Символ `!` означает инверсию критерия. Имена флагов в каждом списке должны разделяться запятыми, пробелы служат для разделения списков
`--syn`	`iptables -p tcp --syn`	Этот критерий аналогичен критерию `--tcp-flags SYN,ACK,FIN SYN`. Такие пакеты используются для открытия соединения TCP. Заблокировав такие пакеты, вы надежно заблокируете все входящие запросы на соединение, однако этот критерий не способен заблокировать исходящие запросы на соединение
`--tcp-option`	`iptables -p tcp --tcp-option 16`	Удовлетворяющим условию данного критерия будет считаться пакет, TCP-параметр которого равен заданному числу. Допускается использование флага инверсии условия `!`

UDP-критерии

Этот набор критериев работает только с UDP-пакетами (табл. 16.10). Чтобы использовать их, вам потребуется в правилах указывать тип протокола `--protocol udp`.

Таблица 16.10. *UDP-критерии проверки пакетов*

Критерий	Пример использования	Описание
`--sport`, `--source-port`	`iptables -A INPUT -p udp --sport 53`	Исходный порт, с которого был отправлен пакет. В качестве параметра может указываться номер порта или название сетевой службы. Номера портов могут задаваться в виде интервала из минимального и максимального номеров. Символ ! используется для инверсии
`--dport`, `--destination-port`	`iptables -A INPUT -p udp --dport 53`	Порт, на который адресован пакет. Формат аргументов полностью аналогичен `--source-port`

ICMP-критерии

Этот набор критериев работает только с ICMP-пакетами (табл. 16.11). Чтобы использовать их, вам потребуется в правилах указывать тип протокола `--protocol icmp`.

Таблица 16.11. *ICMP-критерии проверки пакетов*

Критерий	Пример использования	Описание
`--icmp-type`	`iptables -A INPUT -p icmp --icmp-type 8`	Тип сообщения ICMP определяется номером или именем

Специальные критерии

Перед использованием эти расширения должны быть загружены явно, с помощью ключа `-m` или `--match`. Если мы собираемся применять критерии `state`, то мы должны явно указать это в строке правила `-m state` левее используемого критерия (табл. 16.12).

Таблица 16.12. *Специальные критерии проверки пакетов*

Критерий	Пример использования	Описание
`--limit`	`iptables -A INPUT -m limit --limit 3/hour`	Устанавливается средняя скорость "освобождения емкости" за единицу времени. В качестве аргумента указывается число пакетов и время. Допустимыми считаются следующие единицы измерения времени: `/second`, `/minute`, `/hour`, `/day`

Таблица 16.12 *(продолжение)*

Критерий	Пример использования	Описание
`--limit-burst`	`iptables -A INPUT -m limit --limit-burst 5`	Устанавливает максимальное значение числа `burst limit` для критерия `limit`. Это число увеличивается на единицу, если получен пакет, подпадающий под действие данного правила, и при этом средняя скорость поступления пакетов (задаваемая ключом `--limit`) уже достигнута. Так происходит до тех пор, пока число `burst limit` не достигнет максимального значения, устанавливаемого ключом `--limit-burst`. После этого правило начинает пропускать пакеты со скоростью, задаваемой ключом `--limit`
`--mac-source`	`iptables -A INPUT -m mac --mac-source 00:00:00:00:00:01`	MAC-адрес сетевого узла, передавшего пакет. MAC-адрес должен указываться в форме *XX*:*XX*:*XX*:*XX*:*XX*:*XX*. Этот критерий имеет смысл только в цепочках prerouting, forward и input
`--mark`	`iptables -t mangle -A INPUT -m mark --mark 1`	Критерий производит проверку пакетов, которые были предварительно "помечены". Метки устанавливаются действием MARK
`--source-port`	`iptables -A INPUT -p tcp -m multiport --source-port 22,53,80,110`	Служит для указания списка исходящих портов. С помощью данного критерия можно указать до 15 различных портов. Названия портов в списке должны отделяться друг от друга запятыми, пробелы в списке не допустимы
`--destination-port`	`iptables -A INPUT -p tcp -m multiport --destination-port 22,53,80,110`	Служит для указания списка входных портов
`--port`	`iptables -A INPUT -p tcp -m multiport --port 22,53,80,110`	Критерий проверяет как исходящий, так и входящий порт пакета
`--uid-owner`	`iptables -A OUTPUT -m owner --uid-owner 500`	Производится проверка "владельца" по User ID (UID). Подобного рода проверка может использоваться для блокировки выхода в Интернет отдельных пользователей

Таблица 16.12 *(окончание)*

Критерий	Пример использования	Описание
`--gid-owner`	`iptables -A OUTPUT -m owner --gid-owner 0`	Производится проверка "владельца" пакета по Group ID (GID)
`--pid-owner`	`iptables -A OUTPUT -m owner --pid-owner 78`	Производится проверка "владельца" пакета по Process ID (PID)
`--sid-owner`	`iptables -A OUTPUT -m owner --sid-owner 100`	Производится проверка Session ID пакета
`--state`	`iptables -A INPUT -m state --state RELATED,ESTABLISHED`	Проверяется признак состояния соединения (state). На сегодняшний день можно указывать 4 состояния: `INVALID`, `ESTABLISHED`, `NEW` и `RELATED`. `INVALID` подразумевает, что пакет связан с неизвестным потоком или соединением и, возможно содержит ошибку в данных или в заголовке. `ESTABLISHED` указывает на то, что пакет принадлежит уже установленному соединению, через которое пакеты идут в обоих направлениях. `NEW` подразумевает, что пакет открывает новое соединение или пакет принадлежит однонаправленному потоку. `RELATED` указывает на то, что пакет принадлежит уже существующему соединению, но при этом он открывает новое соединение
`--tos`	`iptables -A INPUT -p tcp -m tos --tos 0x16`	Критерий предназначен для проверки установленных битов TOS. В качестве аргумента критерию может быть передано десятичное или шестнадцатеричное число
`--ttl`	`iptables -A OUTPUT -m ttl --ttl 60`	Производит проверку поля `TTL` на равенство заданному значению

Действия и переходы

Действия и переходы сообщают правилу, что необходимо выполнить, если пакет соответствует заданному критерию.

Действие — это предопределенная команда, описывающая действие, которое необходимо выполнить, если пакет совпал с заданным критерием.

Описание переходов в правилах выглядит точно так же, как и описание действий, т. е. ставится ключ `-j` и указывается название цепочки правил, на которую выполняется переход. На переходы есть ограничения: цепочка, на которую выполняется переход, должна находиться в той же таблице, что и цепочка, из которой этот переход выполняется. Также цепочка, являющаяся целью перехода, должна быть создана до того, как на нее будут выполняться переходы.

При выполнении перехода iptables продолжит движение пакета по цепочке, в которую был произведен переход. Если пакет достиг конца цепочки, то он будет возвращен в вызывающую цепочку и движение пакета продолжится с правила, следующего за правилом, вызвавшем переход. Если к пакету во вложенной цепочке будет применено действие ACCEPT, то пакет будет считаться принятым и в вызывающей цепочке и уже не будет продолжать движение по вызывающим цепочкам.

Действие ACCEPT

Если над пакетом выполняется действие ACCEPT, то пакет прекращает движение по цепочке и считается принятым. Однако пакет продолжит движение по цепочкам в других таблицах и может быть отвергнут там. Действие задается с помощью ключа `-j ACCEPT`.

Действие DNAT

DNAT (Destination Network Address Translation) используется для преобразования адреса места назначения в IP-заголовке пакета. Если пакет подпадает под критерий правила, выполняющего DNAT, то пакет и все последующие пакеты из этого же потока будут подвергнуты преобразованию адреса назначения и переданы на требуемый адрес. Действие DNAT может выполняться только в цепочках prerouting и output таблицы nat и во вложенных цепочках.

Действие DROP

Данное действие удаляет пакет. Пакеты прекращают свое движение полностью — они не передаются в другие таблицы, как это происходит в случае с действием ACCEPT.

Действие LOG

LOG-действие служит для журналирования отдельных пакетов и событий. В журнал могут заноситься заголовки IP-пакетов и другая интересующая вас информация. Информация из журнала может быть затем прочитана с помощью dmesg или syslogd.

Действие MARK

Используется для установки меток для определенных пакетов. Это действие может выполняться только в пределах таблицы mangle. Метка пакета существует только в период времени, пока пакет не покинул брандмауэр, таким образом, метка не передается по сети.

Действие MASQUERADE

Маскарадинг подразумевает получение IP-адреса от заданного сетевого интерфейса, вместо прямого его указания. Действие MASQUERADE имеет хорошее свойство "забывать" соединения при остановке сетевого интерфейса.

Действие MASQUERADE допускается указывать только в цепочке postrouting таблицы nat.

Действие MIRROR

MIRROR производит замену в пакете поля `source` на `destination` и `destination` на `source`. Данное действие допускается использовать только в цепочках input, forward и prerouting, а также в цепочках, вызываемых из этих трех.

Действие QUEUE

QUEUE ставит пакет в очередь на обработку пользовательскому процессу. Оно может быть использовано для нужд учета или дополнительной фильтрации пакетов.

Действие REDIRECT

REDIRECT выполняет перенаправление пакетов и потоков на другой порт того же самого хоста. Действие REDIRECT очень удобно для выполнения "прозрачного" проксирования (transparent proxying), когда машины в локальной сети даже не подозревают о существовании Proxy.

Действие REJECT

REJECT используется в тех же самых ситуациях, что и DROP, но, в отличие от DROP, REJECT выдает сообщение об ошибке на хост, передавший пакет.

Действие RETURN

RETURN прекращает движение пакета по текущей цепочке правил и производит возврат в вызывающую цепочку, если текущая цепочка была вло-

женной, или если текущая цепочка лежит на самом верхнем уровне (например, input), то к пакету будет применена политика по умолчанию.

Действие SNAT

SNAT используется для преобразования сетевых адресов, т. е. изменение исходящего IP-адреса в IP-заголовке пакета. SNAT допускается только в таблице nat, в цепочке postrouting. Если первый пакет в соединении подвергся преобразованию исходящего адреса, то все последующие пакеты, из этого же соединения, будут преобразованы автоматически и не пойдут через эту цепочку правил.

Действие TOS

Команда TOS используется для установки битов в поле `Type of Service` IP-заголовка пакета.

Действие TTL

Действие TTL используется для изменения содержимого поля `Time To Live` в IP-заголовке пакета. Действие TTL можно указывать только в таблице mangle и нигде больше.

Действие ULOG

Действие ULOG предоставляет возможность журналирования пакетов в пользовательское пространство. Оно заменяет традиционное действие LOG, базирующееся на системном журнале. При использовании этого действия пакет передается специальному демону, который может выполнять очень детальное журналирование в различных форматах (обычный текстовый файл, база данных MySQL и т. д.), и поддерживает возможность добавления надстроек для формирования различных выходных форматов и обработки сетевых протоколов.

Утилиты iptables

В этом разделе приведены некоторые утилиты, используемые для управления iptables.

Iptables-save

Утилита iptables-save предназначена для сохранения текущего набора правил в файл, который затем может быть использован утилитой iptables-restore. Эта команда очень проста в применении и имеет всего два аргумента.

```
iptables-save [-c] [-t table]
```

Первый аргумент `-c` (или `--counters`) заставляет iptables-save сохранить значения счетчиков байтов и пакетов.

С помощью ключа `-t` (или `--table`) можно указать имя таблицы для сохранения. Если ключ `-t` не задан, то сохраняются все таблицы.

Iptables-restore

Утилита iptables-restore используется для восстановления набора правил, которые ранее были сохранены с помощью iptables-save. Iptables-restore получает набор правил со стандартного ввода и не может загружать его из файла напрямую. Команда имеет следующий синтаксис:

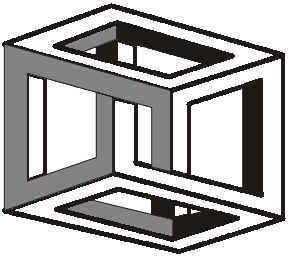
```
iptables-restore [-c] [-n]
```

Ключ `-c` (или `--counters`) позволяет восстановить значения счетчиков.

Ключ `-n` (или `--noflush`) сообщает iptables-restore о том, что правила должны быть добавлены к имеющимся. По умолчанию iptables-restore очищает содержимое таблиц и цепочек перед загрузкой нового набора правил.

Литература и ссылки

- Ipchains HOWTO.
- **bog.pp.ru/work/ipchains.html** — Bog BOS: ipchains: фильтрация пакетов в Linux: принципы работы, установка и настройка.
- **gazette.linux.ru.net/rus/articles/iptables-tutorial.html** — Andreasson O. Iptables Tutorial 1.1.19. Перевод: Киселев А.



ГЛАВА 17

Сетевые принтеры

Из самого словосочетания "сетевые принтеры" понятно, что это принтеры, которые каким-то образом подключены к локальной сети, и компьютеры, подключенные к ней же, которые могут распечатывать документы на данном принтере. Принтеры бывают разные: матричные, струйные, лазерные, сублимационные и т. д. Они могут использовать разные интерфейсы для подключения: последовательный, параллельный, USB, Ethernet, SCSI, FireWire и еще более экзотичный. Производители принтеров продолжают увеличивать набор проблем — то протокол свой придумают, то с целью удешевления создадут Win-принтер, для которого драйвер не достать. И во всем этом приходится разбираться.

Принтер может стать сетевым, по меньшей мере, тремя путями:

- в принтер встроена Ethernet-карта и специальное программное обеспечение для работы в сети;
- принтер подключен к специальному устройству — принт-серверу, который представляет собой специализированный компьютер, с одной стороны подключенный в локальную сеть, а с другой стороны к нему подключаются принтеры;
- принтер подключен к компьютеру, на котором установлено и настроено программное обеспечение, позволяющее компьютерам из сети производить печать на этом принтере.

Далее мы рассмотрим настройку и использование сетевого принтера в обратном порядке: сначала принтер подключен к компьютеру, затем — к специализированному принт-серверу, и наконец, обсудим Ethernet-принтер.

Способы вывода на принтер

Как и для большинства задач, существует несколько способов добиться вывода данных на принтер. Конечно, в конце концов они замыкаются на про-

стой вывод последовательности байтов в порт, к которому подключен принтер, однако с данными по пути следования от документа до распечатки могут производиться различные манипуляции.

Самый простой путь — прямой вывод информации без всякой предварительной обработки на порт принтера. Для этого достаточно выполнить всего лишь следующую команду:

```
cat mytext.txt > /dev/lp
```

Как обычно, это простота кажущаяся. Во-первых, для того чтобы таким образом что-то отправить на печать, необходимо быть пользователем root — для остальных пользователей невозможно напрямую работать с файлами устройств. Во-вторых, зачастую вы получите на распечатке сплошную кашу из символов. Такое произойдет потому, что любой принтер имеет свой специальный язык управления, причем этих языков более десятка разновидностей. Так что выход для данного случая — использовать специальные утилиты, на вход которых подаются текстовые файлы, а на выходе получают преобразованный с учетом языка управления принтера текст. Однако это крайне неудобно. Поэтому применяют специальные программные пакеты, предназначенные для управления печатью. Именно об этих программных пакетах и пойдет далее речь.

Система печати CUPS

CUPS (Common UNIX Printing System, общая система печати для UNIX) интересна своими богатыми возможностями. В ней реализован протокол печати, сходный с протоколом HTTP, заменяющий морально устаревший протокол LPD.

Данная система поддерживает форматы Adobe PostScript, PDF, HP-GL/2, TIFF, JPEG, PNG, PBM, PGM, PPM, GIF, SGI, RGB, Sun Raster, Kodak Photo CDTM. Интересным моментом для администратора являются следующие особенности системы:

- правила управления доступом;
- наличие системы квот;
- авторизация пользователя;
- ведение LOG-журналов.

Программный пакет LPD

LPD (Line Printer Daemon, демон линейной печати) — пожалуй, старейший программный пакет для печати в мире UNIX. Идеология стандартна для UNIX — существуют программы-утилиты для управления процессом печати и программа-демон, обеспечивающая печать на несколько принтеров. Бла-

годаря такому построению программного пакета вы имеете возможность одновременно работать с несколькими принтерами и настроить сетевую печать. В пакет входят следующие программы:

- lpd — демон системы печати;
- lpr — пользовательская утилита печати. Lpr выдает новое задание печати в очередь печати lpd. Синтаксис команды `lpr` очень прост:

  `lpr [ <опции> ] [ <имя_файла> ... ]`

  Если *<имя_файла>* не задано, lpr ожидает ввод данных со стандартного устройства ввода. Это позволяет пользователям перенаправлять вывод команд в очередь печати;
- lpq — утилита для просмотра очереди печати. Команда `lpq`, запущенная без аргументов, возвращает содержимое очереди печати принтера по умолчанию;
- lpc — утилита контроля lpd. С ее помощью можно производить любые манипуляции с очередью печати: добавлять и удалять задания, останавливать печать, переупорядочивать задания в очереди печати и т. д. Lpc чаще всего используется в системах, где несколько принтеров установлено на один компьютер.

  Команда `lpc` обычно используется в интерактивном режиме, однако никто вам не мешает запускать на выполнение эту команду с опциями, некоторые из них приведены далее:

  - `disable <option>` — запрещает добавление любых новых заданий печати;
  - `down <option>` — запрещает все задания на принтере;
  - `enable <option>` — разрешает ввод новых заданий в очередь печати;
  - `quit` (или `exit`) — покинуть lpc;
  - `restart <option>` — перезагрузить lpd для данного принтера;
  - `status <option>` — статус печати принтера;
  - `up <option>` — разрешить все и запустить новый демон lpd;
- lprm — утилита для удаления задания из очереди печати. Команда `lprm` удаляет из очереди все задания печати, владельцем которых является пользователь, выполнивший эту команду. Для того чтобы отменить одиночное задание печати, надо сначала получить номер задания с помощью команды `lpq`, а затем сообщить полученный номер команде `lprm`.

Функционирует система следующим образом. При запуске операционной системы стартует демон lpd. Используя файл /etc/printcap, он узнает, какие принтеры он будет обслуживать. При запуске lpr (пользователь что-то выводит на печать), lpr взаимодействует с lpd через именованный сокет /dev

/printer и передает LPD-файл для печати и некоторую информацию о том, кто печатает и как печатать файл. Затем lpd печатает файл на соответствующем принтере в порядке очереди.

Настройка LPD

Начнем с простого: настроим струйный принтер DeskJet 400 фирмы Hewlett-Packard. Будем считать, что пакет LPD уже установлен в вашей операционной системе, поскольку он входит во множество дистрибутивов как стандартная система печати.

Для добавления очереди печати к lpd вы должны добавить запись в файл /etc/printcap и создать новый буферный каталог в каталоге /var/spool/lpd. Запись в файле /etc/printcap выглядит следующим образом:

```
# ЛОКАЛЬНЫЙ deskjet400
lp|dj|deskjet:\
        :sd=/var/spool/lpd/dj:\
        :mx#0:\
        :lp=/dev/lp0:\
        :sh:
```

Приведенная ранее запись определяет принтер с псевдонимами `lp`, `dj` или `deskjet`, его спул печати размещается в каталоге /var/spool/lpd/dj. Отсутствует ограничение максимального размера задания. Печать производится на устройство /dev/lp0 и не сопровождается выводом титульной страницы задания с именем человека, распечатывающего документ, которая добавляется перед распечаткой файла. Как видите — все очень просто. Но существует извечная проблема текстовых файлов UNIX и Windows — для UNIX в конце текстовой строки достаточно символа перевода строки, для Windows — необходимо наличие символов возврата каретки и перевода строки. Большинство современных принтеров рассчитаны для использования совместно с Windows, и поэтому для нормальной печати текста им также необходимо в конце текстовой строки наличие символов возврата каретки и перевода строки. Если не учесть эту особенность, при распечатке текста на принтере получится приблизительно следующее:

```
Строка номер один
                Строка номер два
                                Строка номер три
                                                Строка номер четыре
```

Это называется лестничным эффектом, и с ним необходимо бороться. Существует много способов, самый простой — написать небольшой фильтр, через который перед печатью будет пропускаться наш текстовый файл, а результат — уходить на печать.

Поправим нашу запись в файле /etc/printcap следующим образом:

```
# ЛОКАЛЬНЫЙ deskjet400
lp|dj|deskjet:\
        :sd=/var/spool/lpd/dj:\
        :mx#0:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/dj/filter:\
        :sh:
```

В документации к printcap описаны атрибуты принтера: `if` — входной фильтр и `of` — выходной фильтр. Как видите, мы определили входной фильтр, расположенный в каталоге /var/spool/lpd/dj/ и носящий имя `filter`. Этот файл представляет собой несколько строчек, написанных на языке Perl:

```
#!/usr/bin/perl
while(<STDIN>){chop $_; print "$_\r\n";};
print "\f";
```

В результате мы получаем принтер, на котором корректно можно распечатать текстовые файлы, используя встроенные шрифты принтера. Для современного мира это не актуально — практически всегда применяется графическая печать. Обычно печатают документы PostScript или графические файлы. На первый взгляд — нетривиальная задача, но на самом деле все достаточно просто. Вспомните еще раз идеологию UNIX — сколь угодно сложные задачи решать применением последовательности небольших утилит.

Для решения этой проблемы опять применяется свойство файла printcap — наличие входных и выходных фильтров. Если мы будем использовать фильтр, который может воспринимать произвольные типы файлов как ввод, обрабатывать их в зависимости от формата файла и производить вывод на принтер, мы решим нашу задачу.

Такой фильтр называется *магическим фильтром* (magic-filter). Существует большое количество магических фильтров, причем наверняка несколько такого типа фильтров находится в вашем дистрибутиве операционной системы. Далее приведены некоторые магические фильтры печати:

- APSFILTER — фильтр печати для стандартного lpd;
- lprMagic — фильтр печати с неплохими возможностями. Автоматически определяет тип входного документа, есть поддержка печати через Samba.

Учет ресурсов

Обычно в больших фирмах принято хранить информацию о том, кто, когда и сколько печатал. Стандартный пакет LPD предоставляет очень небольшую помощь для учета ресурсов. Вы можете указать имя файла для учета ресур-

сов, используя атрибут af= в printcap, но, по большому счету, это не решение проблемы. Пожалуй, лучший вариант — использовать фильтр, который может писать данные в файл учета ресурсов, а вы будете обрабатывать этот файл позже каким-нибудь скриптом обработки статистики.

Настройка сетевого принтера

Одним из важных свойств пакета LPD является то, что он позволяет осуществлять печать по сети на принтер, физически подключенный к другому компьютеру, принт-серверу или просто сетевому принтеру.

Для того чтобы разрешить удаленным компьютерам печатать на ваш принтер, используя протокол LPD, вы должны перечислить эти компьютеры в файле /etc/hosts.lpd. Помимо этого, вы можете разрешить только определенным пользователям с других компьютеров печатать на ваш принтер.

Для того чтобы печатать на другой компьютер, вы должны в файле /etc/printcap сделать следующую запись:

```
# Удаленный deskjet400
lp|dj|deskjet:\
        :sd=/var/spool/lpd/dj:\
        :rm=machine.out.there.com:\
        :rp=printername:\
        :lp=/dev/null:\
        :sh:
```

Как видно из приведенного ранее текста, на нашем компьютере существует каталог очереди печати, обслуживаемой lpd. Это позволяет сохранить и распечатать позднее задание, если удаленная машина занята или отключена. Также мы определяем имя компьютера, который предоставляет нам свой принтер (machine.out.there.com), имя принтера на удаленном компьютере (printername) и показываем, что сетевой принтер не подключен ни к какому ресурсу на нашем компьютере (lp=/dev/null).

Использование принт-сервера

Сегодня уже нет необходимости для небольшой локальной сети покупать достаточно дорогой Ethernet-принтер или выделять отдельный компьютер для организации принт-сервера. Множество фирм выпускают специализированные принт-серверы, причем стоимость некоторых моделей уже менее 100 долларов США.

В качестве примера конфигурации и использования такого принт-сервера в сетях UNIX возьмем принт-сервер фирмы Surecom.

Принт-сервер поддерживает следующие сетевые протоколы:

- Novell NetWare IPX/SPX и NDS;
- TCP/IP;
- DHCP для автоматического получения IP-адреса;
- BOOTP для автоматического получения IP-адреса;
- RARP для автоматического получения IP-адреса.

Для взаимодействия с UNIX-хостами принт-сервер Surecom использует протокол LPD, описанный ранее.

Для конфигурирования принт-сервера необходимо выполнить следующие операции:

1. Включить поддержку протокола TCP/IP.
2. Назначить IP-адрес для принт-сервера.
3. Сконфигурировать удаленную LPD-печать на хостах.
4. Проверить корректность печати.

Опишем все по порядку.

Включить поддержку протокола TCP/IP и назначить IP-адрес для принт-сервера можно с помощью утилиты psetup, поставляемой с принт-сервером. После запуска программы необходимо:

- выбрать пункт **TCP/IP Configuration**;
- в пункте **TCP/IP Support** установить **ENABLE**;
- в пункте **IP Address** указать IP-адрес, присваиваемый принт-серверу, либо установить все нули, если адрес назначается динамически;
- **DHCP server** — использовать динамическое назначение IP-адреса;
- **Gateway IP** — в этом поле задается IP адрес шлюза;
- **Netmask** — сетевая маска;
- **Name server** — адрес DNS-сервера.

После того как вы ввели необходимые параметры, проверьте программой ping прохождение пакетов на ваш принт-сервер. Теперь перейдем к конфигурированию на хостах системы LPD.

На хосте создаем каталог, в котором будет находиться спул печати:

```
mkdir /var/spool/lpd/pserverd
chown daemon /var/spool/lpd/pserverd
chgrp daemon /var/spool/lpd/pserverd
chmod 775 /var/spool/lpd/pserverd
```

Затем в файл /etc/printcap добавляем следующие записи:

```
printer-name:\
:lp=:\
:rm=203.66.191.186:\
:rp=lpt1:\
:lf=/var/spool/lpd/pserverd.log:\
:sd=/var/spool/lpd/pserverd:\
:mx#0:
```

где `rm` — IP-адрес принт-сервера, `sd` — спул принтера, `rp` — имя порта на принт-сервере.

После этого можно произвести пробную печать командой:

```
lpr -P<printer-name> <file> ...
```

Печать на Ethernet-принтер

Обычно высокоскоростные принтеры, позиционируемые производителем как устройства для совместной печати, имеют встроенный сетевой интерфейс, на который вы можете печатать, используя протокол LPD. Обычно в инструкции, идущей в комплекте с принтером, описывается, каким образом необходимо настроить клиентский компьютер для печати на сетевой принтер. Например, следующая запись в файле printcap используется для работы с сетевым принтером фирмы Hewlett-Packard:

```
lj-5|remote-hplj:\
        :lp=/dev/null:sh:\
        :sd=/var/spool/lpd/lj-5:\
        :rm=printer.name.com:rp=raw:
```

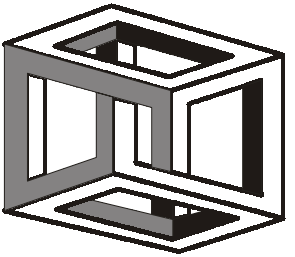
Принтеры HP LaserJet с интерфейсами Jet Direct поддерживают две встроенные очереди LPD — `raw`, которая принимает PCL (или Postscript), и `text`, которая принимает "чистые" файлы ASCII и автоматически справляется с лестничным эффектом.

Литература и ссылки

- Документация на принт-сервер Surecom.
- Linux Printing HOWTO — Komarinski M. Использование печати в Linux.
- **hpinkjet.sourceforge.net** — драйверы для принтеров Hewlett-Packard непосредственно от фирмы-производителя.

- **61.251.162.120:8080** — драйверы для принтеров Samsung от фирмы-производителя. Поддерживаются все принтеры серии ML.
- **feynman.tam.uiuc.edu/pdq/** — страница PDQ.
- **ftp://ppr-dist.trincoll.edu/pub/ppr/** — местонахождение PPR — системы буферизации печати, ориентированной на PostScript.
- **linuxcenter.ru/lib/hardware/usbprinter.phtml** — Лушня Ю. Настраиваем USB-принтер под Linux.
- **linux.yaroslavl.ru/Docum/Rus/print.html** — Толпекин В. Настройка сетевого принтера для печати русского текста.
- **metalab.unc.edu/pub/Linux/system/printing/** — lprMagic: Фильтр печати с неплохими возможностями.
- **www.astart.com/lprng/LPRng.html** — страница проекта LPRng.
- **www.citycat.ru/linux/docs/index.html** — сайт, содержащий много интересной документации по Linux на русском языке.
- **www.freebsd.org/~andreas/#apsfilter** — страница APSFILTER: Магический фильтр для печати.
- **www.linuxdoc.org/** — сайт, содержащий много интересной документации по Linux на английском языке.
- **www.linuxrsp.ru/artic/print_server.html** — Лушня Ю. Печать в Linux с железными нервами.
- **www.Linux-USB.org** — сайт, посвященный USB-устройствам и их применимости с точки зрения Linux.
- **www.l0pht.com/~weld/netcat/** — страница netcat, пакета для работы с принтером.
- **www.penguincomputing.com/prtools/npadmin.html** — страница npadmin, программы для управления сетевыми принтерами. Управление осуществляется через SNMP.
- **www.redhat.com** — Red Hat Linux/x86 9 Release Notes.
- **www.redhat.com** — Red Hat Linux 9. The Official Red Hat Linux Customization Guide.
- **www.redhat.com** — Red Hat Linux 9. The Official Red Hat Linux Getting Started Guide.

ГЛАВА 18



Организация шлюза в Интернет для локальной сети

Для организации доступа в Интернет из локальной сети обычно используется либо аппаратное решение (достаточно дорогое), либо аппаратно-программное (модем, сервер, программы — менее дорогое и, соответственно, менее надежное). Возникает вопрос: "Раз аппаратно-программное решение менее надежно, то почему же его в десятки раз чаще используют для организации шлюзов?" Как обычно, все упирается в деньги — первое решение стоит порядка 2—5 тысяч американских долларов, второе — обойдется в 300—600 американских долларов. Вот такая экономика. Надежность можно разделить на две составляющие: надежность программной части и надежность аппаратной части. С программной частью просто — правильная установка, конфигурирование, своевременная установка обновлений программ, а также грамотная политика резервирования данных позволят вам почти на сто процентов быть уверенным в надежности и безопасности программной части. Что же касается аппаратной части — тут тоже не все так плохо. По статистике самые ненадежные части компьютера — вентиляторы, блоки питания и различные приводы. Соответственно для блока питания можно установить ИБП, чтобы избавиться от вентиляторов можно взять платформу от VIA EDEN, которая не требует активного охлаждения, а с накопителями все легко — для простого маршрутизатора достаточно однодискетного дистрибутива, либо можно воспользоваться Flash-носителем.

Обычно для организации шлюза в качестве физического соединения используется выделенная линия, по концам которой установлены модемы, подключаемые к последовательному либо к синхронному порту. В последнее время все чаще для организации подключения по выделенной линии используются технологии xDSL, при которых специальные модемы подключаются по интерфейсу Ethernet напрямую к сетевой карте, причем зачастую модемы сами уже и являются маршрутизаторами. Для определенности будем считать, что у нас есть модем, подключенный к последовательному порту, а xDSL-технологии мы обсудим в самом конце.

Обычно в небольших локальных сетях используется один компьютер, выполняющий роль маршрутизатора между локальной сетью и Интернетом, а также — счетчика трафика, брандмауэра, ограничителя скорости Web-сервера и т. п. Почти все, что необходимо для создания такой многопрофильной системы, мы уже рассматривали ранее, поэтому на особенностях протоколов, реализаций и программного обеспечения особо останавливаться не будем.

Начальные установки

Как правило, во всех современных дистрибутивах Linux ядро собрано так, что работает как маршрутизатор пакетов между разными сетями и поддерживает механизм защиты маршрутизируемых пакетов и подсчет статистики.

Однако не будет лишним убедиться перед началом настройки системы, что в ядре вашей операционной системы присутствуют следующие необходимые для построения маршрутизатора элементы (функции):

- Networking support (поддержка сетевых свойств);
- TCP/IP networking (поддержка TCP/IP);
- IP forwarding/gatewaying (поддержка IP-маршрутизации);
- IP multicasting (поддержка специфических свойств IP-протокола);
- IP firewalling (поддержка firewall);
- IP accounting (поддержка управления IP);
- Network device support (поддержка сетевых устройств).

Помимо этого, ядро операционной системы должно уметь работать с сетевыми картами, установленными на вашем компьютере, и поддерживать протокол PPP (Point-to-Point Protocol).

Само собой, следует правильно настроить сетевое оборудование, IP-адреса и т. п.

Связь с провайдером

Для подключения локальной сети к Интернету при помощи модема обычно используют два варианта. Первый из них предназначен для тех, кто оплачивает проведенное в Интернете время, а второй — кто платит за трафик.

В первом случае выход в Интернет осуществляется по коммутируемому телефонному соединению, при помощи стандартного для Linux набора программ: pppd, chat и, возможно, еще нескольких скриптов. Происходит это следующим образом: вначале маршрутизатор дозванивается до провайдера и устанавливает с ним связь по протоколу PPP или по протоколу SLIP. После установления соединения полученным каналом может пользоваться любой

компьютер в вашей локальной сети (при соответствующей настройке). Канал удерживается до тех пор, пока не выключится ваш маршрутизатор или администратор явным образом не разорвет соединение.

Модификация предыдущего варианта (более универсальная), в англоязычной литературе носит название dial on demand (звонок по требованию). Для его организации дополнительно используется программа diald, с помощью которой можно организовать работу таким образом, что если в течение заранее обусловленного времени не происходит обмена данными между локальной сетью и Интернетом, то diald разрывает соединение. При первой же попытке пользователя подключиться к Интернету diald снова дозванивается и устанавливает связь.

Поскольку второй вариант более сложный, будем рассматривать его как основной для организации нашего маршрутизатора.

Схема организации подключения локальной сети

Далее приведены требования, которым должно удовлетворять подключение локальной сети к Интернету.

- Возможность доступа в Интернет — модем, телефонный номер и подключение к провайдеру.
- Набор программ для организации связи — pppd, chat и diald.
- Средство для управления брандмауэром — утилита ipchains или iptables.
- Средство для ограничения трафика (если необходимо).
- Программное обеспечение для организации Proxy-сервера.
- Программное обеспечение для учета и просмотра статистики.

Теперь можно приступать к настройке программ.

Организация связи по коммутируемому соединению

Это старейший вариант соединения с провайдером, и, к сожалению, наиболее распространенный в нашей стране. По сравнению с организацией связи по выделенному каналу представляет собой схему более сложную, поэтому рассмотрим ее первой.

Настройка программ

Будем считать, что на компьютере, который должен выходить в Интернет, правильно настроены сетевые параметры, и вы убедились в работоспособ-

ности локальной сети. Следующий шаг — добиться устойчивой связи с провайдером на вашем компьютере-маршрутизаторе.

Настройка связи с провайдером

Настроим подсистему дозвона и соединения с провайдером. Для удобства разобьем работу на два этапа:

1. Настройку PPP-соединения.
2. Установку и конфигурирование демона дозвона по требованию (diald).

Настройку модемного соединения мы здесь рассматривать не будем, поскольку это достаточно простая задача, и она очень подробно рассмотрена в работе одного из отечественных патриархов Linux — В. Водолазкого "Установка PPP-соединения в Linux".

Почему мы используем протокол PPP? Основные преимущества протокола PPP по сравнению с протоколом SLIP состоят в следующем:

- назначение IP-адресов в PPP реализуется с помощью демона pppd, что значительно упрощает процесс конфигурирования при использовании динамических IP-адресов;
- коррекция ошибок, возникающих при передаче данных, осуществляется между компьютером провайдера и клиента, а не между удаленным компьютером, откуда берутся данные, и потребителем, как в протоколе SLIP.

Для организации связи между провайдером и клиентом необходимо получить данные, представленные в табл. 18.1.

Таблица 18.1. *Необходимые данные для настройки модемного соединения*

Необходимые данные	Значения в примере
Имя пользователя (login)	`myname`
Пароль пользователя (password)	`Passpass22`
IP-адрес пользователя (если есть)	`192.168.0.10`
IP-адрес сервера DNS	`192.168.10.1`

Процесс установления связи между вами и провайдером состоит из следующих этапов:

- соединения с компьютером провайдера с помощью модема;
- регистрации пользователя в удаленной системе;
- установки PPP-соединения.

Для решения этих задач в Linux используется небольшой набор скриптов, каждый из которых выполняет какую-то небольшую функцию. А поскольку

это набор скриптов — никто не мешает на их базе определить именно те действия, которые необходимы вам при установке или обрыве PPP-соединения.

Размещение скриптов зависит от настройки и предпочтений вашего дистрибутива. В современных версиях дистрибутива Red Hat используется два места — каталоги /etc/ppp и /etc/sysconfig/network-scripts. Наименования скриптов так же могут быть произвольными и очень часто зависят от предпочтений сборщика дистрибутива или системного администратора.

Для нашего случая будем считать, что у нас есть следующие файлы:

- /etc/ppp/chap-secrets — этот файл используется для аутентификации пользователя провайдером по протоколу CHAP. Обычно содержит имя и пароль пользователя для входа к провайдеру. В нашем случае это будет выглядеть следующим образом:

  ```
  myname * Passpass22
  ```

- /etc/ppp/pap-secrets — данный файл используется для аутентификации пользователя провайдером по протоколу PAP. Обычно содержит имя и пароль пользователя для входа к провайдеру. В нашем случае это будет выглядеть следующим образом:

  ```
  myname * Passpass22
  ```

- /etc/ppp/ip-up — данный скрипт используется для соединения с провайдером. Зачастую этот файл содержит только следующую строку:

  ```
  /usr/sbin/pppd
  ```

  Здесь можно настроить установку модемом соединения с провайдером либо вызвать необходимый скрипт или программу;

- /etc/ppp/ip-down — этот файл используется для разрыва соединения с провайдером;

- /etc/ppp/options — это, пожалуй, самый сложный и ответственный файл. Он определяет параметры нашего модема, скорость передачи по последовательному интерфейсу данных, настройки программы pppd и некоторые другие параметры. Обычно файл /etc/ppp/options оставляют неизменным, а для конфигурирования параметров соединения создают копию файла с именем /etc/ppp/options.ttyS*X*, где ttyS*X* — имя последовательного порта, к которому подключен наш модем. Пусть для определенности модем подключен к ttyS0 (COM1).

  ```
  # Устройство
  /dev/ttyS0
  # Скорость
  115200
  mru 1500
  ```

```
# наш интерфейс : удаленный интерфейс
192.168.0.10:192.168.0.11
# маска подсети
netmask 255.255.255.0
bsdcomp 0
chap-interval 15
debug
crtscts
defaultroute
```

Первые две строки определяют последовательный порт, к которому подключен наш модем, и скорость, на которой будет происходить обмен между модемом и последовательным портом. Далее обратите внимание на строку со следующим содержимым:

```
192.168.0.10:192.168.0.11
```

Эта строка определяет IP-адреса нашего последовательного интерфейса и провайдера. Такую строку необходимо добавить, если провайдер выдал нам постоянный IP-адрес. Как правило, в современном мире с коммутируемыми соединениями такого не происходит. Для статического IP-адреса также нужно задать маску подсети.

Поскольку наш компьютер является маршрутизатором для локальной сети, необходимо настроить маршрутизацию. Для этого воспользуйтесь программой route и идущей с ней документацией. В том случае (а мы предположили, что точка подключения к провайдеру у нас одна) если у вас одно подключение к провайдеру, то можно в конец файла вписать команду `defaultroute`, что позволит вам добавить маршрут в системную таблицу маршрутизации, используя удаленную сторону как шлюз.

Настройка diald

Обычно программа diald входит в стандартный дистрибутив, и установка ее с помощью менеджера пакетов rpm занимает совсем немного времени. После инсталляции необходимо привести стандартную конфигурацию программы diald в соответствие с нашими требованиями.

Чтобы лучше понять то, что мы будем делать дальше, расскажем немного о принципе работы программы diald. Программа создает соединение на псевдотерминале и устанавливает маршрутизацию на получившийся интерфейс. После этого она начинает отслеживать пакеты, проходящие по виртуальному каналу. Если кто-то пытается выйти в Интернет, diald перехватывает данные, анализирует их и на основе правил, определяемых администратором, присваивает им определенные тайм-ауты. Далее пакеты отправляются по назначению, а тайм-ауты заносятся в так называемый набор соединения.

Как только в наборе появляется первый тайм-аут, diald начинает дозваниваться до провайдера и пытается установить соединение. Организовав сеанс связи, демон переустанавливает маршрутизацию на реальный канал. Таким образом, связь с внешним миром оказывается установленной.

На протяжении всего времени соединения продолжает обновляться набор соединения. Истекшие тайм-ауты удаляются, новые поступают. И так продолжается до тех пор, пока по какой-либо причине трафик не прекратится. Тайм-аутов в наборе становится все меньше и меньше, и когда последний из них оканчивается, diald разрывает связь.

Теперь перейдем к конфигурированию. Этот процесс состоит из трех частей:

- создание скрипта соединения — файл /etc/diald/connect;
- настройка основной конфигурации — файл /etc/diald.conf;
- настройка правил тайм-аутов — файл /etc/diald/standard.filter.

Создание скрипта соединения: /etc/diald/connect

Для организации сеанса связи необходимо выполнить несколько действий: дозвониться по телефону до поставщика услуг, пройти процедуру авторизации и запустить PPP-соединение. Поскольку у разных провайдеров этот процесс может коренным образом отличаться, то не имеет смысла встраивать эту процедуру в программу. Вместо этого используется внешний скрипт. Для чего достаточно подправить тот скрипт, который входит в стандартную поставку diald.

В листинге 18.1 приведен вариант файла /etc/diald/connect.

Листинг 18.1. Файл /etc/diald/connect

```
#!/bin/sh

NIT="ATZ"    # Строка инициализации модема
PHONE="2223322"    # Телефон провайдера
ACCOUNT="myname"    # логин
PASSWORD=" Passpass22"    # пароль
# Определяем функцию для посылки
# сообщений в системный журнал
# и в FIFO-канал diald
function message ()
 {
   [ $FIFO ] && echo "message $*" >$FIFO
  logger -p local2.info -t connect "$*"
 }
```

```
# Начинаем процедуру связи
# Инициализируем модем
message "*** Initializing Modem ***"
chat "" $INIT OK ""
if [ $? != 0 ]
   then
      message "!!! Failed to initialize modem !!!"
      exit 1
fi
# Пытаемся дозвониться
message "*** Dialing system  ***"
chat \
      ABORT "NO CARRIER" \
      ABORT BUSY \
      ABORT "NO DIALTONE" \
      ABORT ERROR \
      "" ATDT$PHONE \
      CONNECT ""
case $? in
      0) message "*** Connected ***";;
      1) message "!!! Chat Error !!!"; exit 1;;
      2) message "!!! Chat Script Error !!!"; exit 1;;
      3) message "!!! Chat Timeout !!!"; exit 1;;
      4) message "!!! No Carrier !!!"; exit 1;;
      5) message "!!! Busy !!!"; exit 1;;
      6) message "!!! No DialTone !!!"; exit 1;;
      7) message "!!! Modem Error !!!"; exit 1;;
      *) esac
# Проходим авторизацию
message "*** Send login and password ***"
chat \
      login: $ACCOUNT \
      password: $PASSWORD        TIMEOUT 5 ""
if [ $? != 0 ] then
      message "!!! Failed to send !!!"
      exit 1
fi
# Все прошло удачно!
message "*** Protocol started *** "
```

Приведенный ранее скрипт — просто сценарий на языке командной оболочки, который необходимо немного адаптировать для ваших параметров.

Настройка основной конфигурации: /etc/diald.conf

/etc/diald.conf — основной конфигурационный файл программы diald, в котором задаются параметры устанавливаемого соединения и определяется поведение программы. Набор команд конфигурации у diald достаточно обширен, поэтому в приведенном листинге 18.2 будут использованы только необходимые, а подробную информацию по конфигурационным командам можно посмотреть в документации на программу diald.

Листинг 18.2. Файл diald.conf

```
# Протокол для связи с провайдером
mode ppp
# Вести журнал сеансов связи diald.log
accounting-log /var/log/diald.log
# Для управления демоном из внешних программ
# организовать канал FIFO — diald.ctl.
fifo /etc/diald/diald.ctl
# Для дозвона использовать файл /etc/diald/connect
connect /etc/diald/connect
# Далее несколько команд, описывающих применяемый модем.
# Поскольку мы уже определили параметры в /etc/ppp/options,
# то нижеприведенные команды необходимо закомментировать во избежание
# конфликтов в файле /etc/ppp/options
#device /dev/modem
#speed 115200
#modem
#lock
#crtscts
# Назначаем локальный и удаленный адреса нашего
# соединения. Если при связи с провайдером IP-адрес
# для вас выделяется динамически, то здесь можно
# поставить любые свободные адреса из диапазона,
# оговоренного при настройке нашей TCP/IP-сети.
# При запуске PPP diald сам выставит корректные значения
local 192.168.0.10
remote 192.168.0.11
# Провайдер дает нам динамический IP
dynamic
# Установить маршрут по умолчанию
# на виртуальное соединение
defaultroute
# Максимальное количество неудачных попыток дозвона
dial-fail-limit 10
```

```
# Задержка между попытками дозвона
redial-timeout 5
# Время ожидания завершения скрипта connect
connect-timeout 120
# Файл с правилами для тайм-аутов
include /etc/diald/standard.filter
```

Настройка правил тайм-аутов: /etc/diald/standard.filter

Следующее, что вы должны сделать, — произвести настройку правил тайм-аутов. Это самый сложный момент настройки diald, т. к. он требует знания внутренней структуры IP-пакетов. Однако стандартный файл standard.filter имеет вполне приемлемые для большинства случаев настройки. Оставив в нем все, как есть, мы получим набор правил, рассчитанный на трехминутную паузу между окончанием активности в Интернете и разрывом связи с провайдером.

Комплексное тестирование

После проделанных манипуляций настало время проверить, правильно ли настроены наши программы. Для этого на компьютере желательно временно отключить все настройки брандмауэра (если вы, конечно, установили его). Затем необходимо запустить программу diald и попытаться выйти в "большой мир". Можно использовать браузер lynx (и зайти, например, на сайт **http://www.bhv.ru**), можно — программу ping.

Если все было настроено корректно, то после ввода предыдущей команды модем должен начать дозваниваться до провайдера. Через некоторое время связь будет установлена. Однако практически всегда lynx выдает сообщение о том, что не может соединиться с удаленным сервером! В данном случае — это нормальное явление. Дело в том, что при PPP-соединении с динамическими IP-адресами в силу определенных особенностей первый пакет обычно бывает утерян и не доходит до адресата. В результате мы ждем ответа от сервера, а он об этом и не подозревает. Достаточно повторить введенную ранее команду, чтобы все заработало.

Далее нам необходимо убедиться, что модем аккуратно разорвет соединение по прошествии трех минут. Дождавшись конца загрузки Web-страницы, засечем время. Примерно через три минуты diald должен дать команду на разрыв соединения.

Если у вас все прошло именно таким образом, значит, система работает как надо. В противном случае проанализируйте последние строки системного журнала (/var/log/messages).

Указанными действиями мы проверили корректную работу только с нашего компьютера-маршрутизатора. Однако нам надо сделать то же самое и с лю-

бого компьютера в локальной сети, поэтому попробуем повторить описанную процедуру на любом компьютере. Реакция diald должна быть аналогичной. Если что-то пошло не так, проверьте корректность настройки протокола TCP/IP на этой машине, в частности — настройки сетевого шлюза, которые должны указывать на наш компьютер-маршрутизатор.

Организация связи по выделенному каналу

В отличие от настройки связи по коммутируемому соединению, организация соединения по выделенному каналу намного более простая задача для асинхронного соединения и совсем простая для соединения при помощи xDSL-модемов.

Настройка связи с провайдером

Как и в предыдущем случае, нам необходимо правильно настроить программу pppd. Поскольку параметры программы pppd мы уже рассматривали, просто приведем файл options и прокомментируем его содержание.

```
# Устройство
/dev/ttyS0
# Скорость
115200
mru 1500
noauth
# наш интерфейс : удаленный интерфейс
192.168.0.10:192.168.0.11
# маска подсети
netmask 255.255.255.0
bsdcomp 0
chap-interval 15
debug
crtscts
-detach
defaultroute
```

Первые две строки определяют последовательный порт, к которому подключен наш модем, и скорость, на которой будет происходить обмен между модемом и последовательным портом. Далее обратите внимание на строку со следующим содержимым:

```
192.168.0.10:192.168.0.11
```

Эта строка определяет IP-адреса нашего последовательного интерфейса и провайдера. Такую строку необходимо добавить, если провайдер выдал нам постоянный IP-адрес. Для статического IP-адреса также необходимо задать маску подсети.

В том случае, если у вас одно подключение к провайдеру, то можно в конец файла вписать команду `defaultroute`, что позволит вам добавить маршрут в системную таблицу маршрутизации, используя удаленную сторону как шлюз.

Вот и все, что требовалось для конфигурации программы pppd для соединения по выделенному каналу. Правда, намного проще, чем с коммутируемым соединением?

Осталось только отредактировать файл inittab, для того чтобы наш pppd автоматически стартовал. Туда необходимо добавить следующую строчку:

```
7 : 2345 : respawn: /usr/sbin/pppd file /etc/ppp/options.ttyS0 >
/var/log/pppS0.log
```

Комплексное тестирование

Теперь настало время проверить, правильно ли настроено наше соединение по выделенному каналу. Для этого перезагрузите компьютер-шлюз для вступления в силу внесенных в файл inittab изменений и временно отключите все настройки брандмауэра (если вы, конечно, установили его). Затем необходимо попытаться выйти в Интернет. Быстрее всего — использовать программу ping:

```
ping www.bhv.ru
```

Если все было настроено корректно, то вы увидите отклик от сайта **www.bhv.ru**.

Если у вас все прошло именно таким образом, значит, система работает как надо. В противном случае проанализируйте последние строки системного журнала (/var/log/messages).

Этим действием мы проверили корректную работу только с нашего компьютера-маршрутизатора. Однако нам надо сделать то же самое и с любого компьютера в локальной сети. Если что-то пошло не так, проверьте корректность настройки протокола TCP/IP на этой машине, в частности — настройки сетевого шлюза, которые должны указывать на наш компьютер-маршрутизатор.

Для xDSL-модемов все намного проще. На вашем компьютере задается IP-адрес xDSL-модема в качестве шлюза сети, а сам модем конфигурируется с помощью программы telnet по последовательному порту.

Итак, вы получили вполне работоспособный шлюз в Интернет для вашей локальной сети. Однако это далеко не все. Система наша открыта для лю-

бого постороннего вмешательства, а шлюз должен обеспечить незащищенную локальную сеть защитой извне и изнутри, вести учет потребленного трафика (и причем зачастую — покомпьютерно), ограничить нас от информации нежелательной или сомнительной (например, баннеров), обработать статистику и красиво ее подать — лучше всего графически. Как видите — задач много, и мы будем их решать постепенно.

Защита локальной сети

Защита локальной сети — понятие комплексное и многогранное. В данном случае мы имеем в виду правильную настройку брандмауэра на нашем компьютере-шлюзе. Процедура настройки брандмауэра была описана в *гл. 16*.

Установка Proxy-сервера

Следующее, что мы должны решить для нашей локальной сети — каким образом минимизировать расходы на потребляемый Интернетом трафик и как увеличить скорость получения информации. Для решения этой проблемы используется стандартный рецепт — Proxy-сервер.

Наиболее часто используемой программой Proxy является программа Squid — высокопроизводительный кэширующий Proxy-сервер, поддерживающий протоколы FTP, Gopher и HTTP. Squid сохраняет часто запрашиваемые данные в оперативной памяти компьютера, что позволяет резко увеличить производительность Proxy-сервера, кэширует DNS-запросы (это свойство интересно тем, кто не имеет своего DNS-сервера). Помимо перечисленных ранее возможностей, поддерживает SSL, расширенный контроль доступа и полную регистрацию запросов.

Программа Squid описана в *гл. 11*, однако мы позволим себе напомнить некоторые интересные моменты по ее использованию.

Transparent Proxy

Transparent Proxy — таким образом настроенный Proxy-сервер, что его использование прозрачно для пользователей. Это имеет как хорошую, так и плохую стороны. С одной стороны, пользователям не придется настраивать соединение через Proxy-сервер в своей системе, а трафик гарантированно проходит через Proxy-сервер. С другой стороны, теряется свобода выбора пользователя — пользоваться или нет Proxy-сервером. Кроме того, некоторые сайты некорректно обрабатываются Proxy.

Для организации Transparent Proxy необходимо таким образом настроить маршрутизатор (брандмауэр), чтобы транзитные пакеты, предназначенные для порта 80, попадали на вход Proxy-сервера. Соответствующие настройки Transparent Proxy приведены в *гл. 11*.

Борьба с баннерами

Наверняка вам встречались Web-страницы, на которых рекламных баннеров было больше, чем нужной информации. В этом случае можно настроить локальный сервер Squid таким образом, чтобы не происходила закачка баннеров. Борьбу с баннерами можно производить разными методами:

- настроить отдельный Proxy-сервер с ограничением баннеров: хочешь — используй, не хочешь — не используй;
- совместить ограничение баннеров с Transparent Proxy;
- организовать Proxy на локальной системе для ограничения баннеров.

Соответствующие настройки программы Squid для борьбы с баннерами приведены в *гл. 11.*

Разделение внешнего канала (ограничение трафика)

Часто бывает так, что у вас есть внешний канал, скажем, 128 Кбит, и есть несколько групп пользователей с определенным приоритетом. Требуется, чтобы группа 1 имела одну фиксированную ширину наружного канала (скажем, 64 Кбит), а группа 2 и 3 — ширину наружного канала по 32 Кбит. Для решения этой задачи мы также можем воспользоваться Squid. Соответствующие настройки программы Squid для разделения внешнего канала приведены в *гл. 11.*

Помимо Squid для этого можно воспользоваться специализированными программами, называемыми traffic shaper. Существует несколько программ такого типа с различной функциональностью. В частности, есть traffic shaper, которая позволяет ограничить канал не по пропускной способности, а по полученным мегабайтам. Принцип действия ее прост. Допустим, у вас выделенный канал, причем в арендную плату входит один гигабайт входящего трафика. В программе traffic shaper выставляется ограничение один гигабайт в месяц. Далее происходит следующее. Вначале информация качается с той скоростью, с какой способен передавать информацию выделенный канал, но при приближении к заветной цифре — пропускная способность канала, ограниченного traffic shaper, все уменьшается и уменьшается, не позволяя вам выйти за рамки ограничения в один гигабайт в месяц. В результате, в последние дни месяца скорость канала может упасть до десятков байтов в секунду.

В качестве стабильной и хорошо конфигурируемой программы типа traffic shaper можно порекомендовать пакет CBQ. Ограничивать трафик можно и с помощью утилиты tc, входящей в пакет iproute2.

Мониторинг загрузки каналов

Для анализа загрузки интернет-канала необходимо использовать дополнительный пакет, поскольку разбираться в LOG-файлах системы нудно и долго. Чтобы обеспечить требуемую наглядность, такой пакет должен выдавать информацию в графической форме, причем, желательно, с помощью Web-интерфейса. Все эти условия реализованы в программах MRTG (Multi Router Traffic Grapher) и RRDtool (Round Robin Database).

Программа MRTG

MRTG создает HTML-страницу с отображением загрузки канала за сутки, неделю, месяц и год. Для этого используется написанный на языке Perl скрипт, который опрашивает маршрутизатор через SNMP, а программа, написанная на языке C, обрабатывает получившийся результат и создает изображения в формате GIF/PNG, встроенные в HTML-страницу. Помимо самостоятельно собранной информации пакет MRTG может обрабатывать информацию и из других источников (cpuinfo, df, squid и т. п.), а также строить графики по полученной информации.

Большим преимуществом данной программы является постоянный размер журналов, в которых более старая информация хранится с меньшими подробностями.

Внешний вид получаемых графиков приведен на рис. 18.1.

Конфигурирование MRTG

Для конфигурирования программы MRTG используется файл mrtg.cfg, параметры которого и будут рассматриваться в данном разделе. Мы опишем только ключевые параметры, с полным списком можно ознакомиться в документации, прилагаемой к этому программному пакету.

Правила записи параметров в конфигурационном файле:

- ключевое слово — в начале строки;
- двоеточие — разделитель, идущий сразу за ключевым словом;
- строка продолжения начинается с пробела;
- строки комментария начинаются с символа `#`.

Итак, файл mrtg.cfg может содержать следующие команды:

- `Include:` *<имя_файла>* — подключаемый файл;
- `WorkDir:` *<имя_каталога>* — задает размещение журнала, рабочих файлов и генерируемых страниц, имеет приоритет над `HtmlDir`, `ImageDir` и `LogDir`;

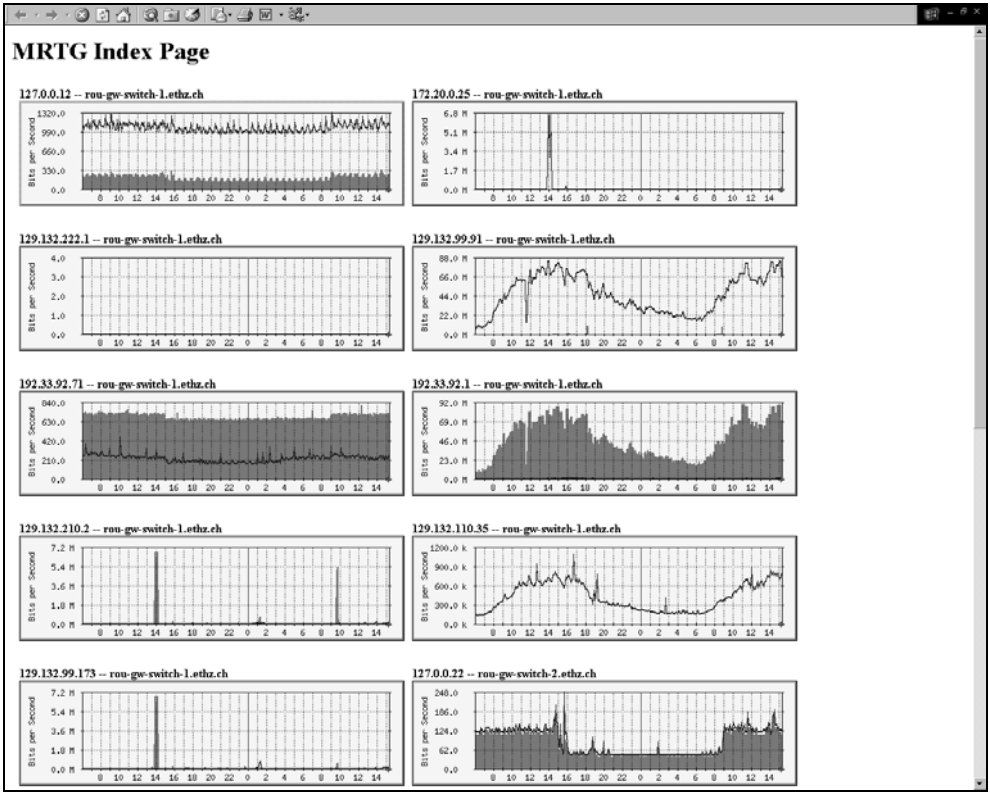


Рис. 18.1. Результат работы программы MRTG

- `HtmlDir: <имя_каталога>` — задает размещение генерируемых страниц;
- `ImageDir: <имя_каталога>` — задает размещение генерируемых изображений; обязательно находится под `HtmlDir` — страницы генерируются в этом предположении;
- `LogDir: <имя_каталога>` — задает размещение журнала;
- `Refresh: <секунды>` — частота перерисовки графиков в браузере;
- `RunAsDaemon: no | yes` — запуск MRTG в режиме демона;
- `Interval: <секунды>` — предполагаемый интервал запуска MRTG;
- `IconDir: <имя_каталога>` — каталог, где хранятся значки;
- `Forks: <число>` — определяет, сколько запускать параллельных процессов опроса;
- `WriteExpire: no | yes` — создавать META-файлы для Apache;
- `NoMib2: no | yes` — не запрашивать `sysUptime`, `sysName`;

- `Language: <язык_отчетов>` — определяет язык отчетов, есть поддержка русского;
- `LogFormat: rrdtool` — формат журналов для rrdtool — динамическое создание отчетов;
- `LibAdd: <адрес-библиотеки>-rrdtool RRDs.pm`;
- `PathAdd: <адрес>-rrdtool` — путь к `rrdtool`.

Для каждого контролируемого устройства (обозначается как target, буквы преобразуются к строчным) создается отдельная секция. При работе MRTG каждый target порождает файлы журнала (target.log и target.old), картинки с графиками (target-day.gif, target-week.gif, target-month.gif, target-year.gif) и HTML-страницу (target.html).

- `Target[target]: <порт>:community@<маршрутизатор>`
  `[:port[:timeout[:retries[:backoff[:2]]]]]`

  где:

  - `<порт>` — номер интерфейса на маршрутизаторе;
  - `community` — пароль на чтение;
  - `<маршрутизатор>` — имя или IP-адрес;
  - `port` — по умолчанию стандартный порт SNMP;
  - `timeout` — время ожидания;
  - `retries` — количество попыток;
  - `backoff` — во сколько раз увеличивать timeout при каждом повторе;
  - `2` — означает использование 64-битовых счетчиков;
- `Target[target]: <внешняя-программа-с-параметрами-в-обратных-кавычках>`

  Программа должна возвращать на стандартный вывод 4 строки:

  - значение счетчика входных байтов;
  - значение счетчика выходных байтов;
  - текстовая строка, содержащая информацию о времени работы объекта после включения;
  - строка, указывающая имя объекта;
- `RouterUptime[target]: community@<маршрутизатор>` — откуда брать информацию об имени маршрутизатора и его времени работы для составных `target`;
- `MaxBytes[target]:<число>` — значения переменных, которые больше этого числа, игнорируются;
- `Title[target]:` — заголовок для HTML-страницы;

- `PageTop[target]:` — текст, выдаваемый в верхней части HTML-страницы;
- `PageFoot[target]:` — текст, выдаваемый в нижней части HTML-страницы;
- `AddHead[target]:` — HTML-текст, вставляемый после `TITLE` внутри `HEAD`;
- `MaxAbs[target]:` *<число>* — если используется сжатие, то возвращаемое значение может превосходить `MaxByte`.;
- `Unscaled[target]:` `[d][w][m][y]` — подавить масштабирование по вертикали для соответствующего графика (`d` — день, `w` — неделя, `m` — месяц, `y` — год);
- `WithPeak[target]:` `[w][m][y]` — показывать в недельном, месячном и годовом графиках не только средние, но и пиковые значения (`w` — неделя, `m` — месяц, `y` — год);
- `Supress[target]:` `[d][w][m][y]` — подавить генерацию части графиков (`d` — день, `w` — неделя, `m` — месяц, `y` — год);
- `Directory[target]:` *<имя-каталога>* — размещать в данном каталоге все файлы, относящиеся к указанному `target`;
- `XSize[target]:` *<число>* — число пикселов в графике по горизонтали;
- `YSize[target]:` *<число>* — число пикселов в графике по вертикали;
- `YTics[target]:` *<число-вертикальных-делений>*;
- `Step[target]:` *<секунд>* — определяет шаг отображения в секундах;
- `Options[target]:` *<список-опций-через-запятую>*:
  - `growright` — время движется вправо;
  - `bits` — все числа умножать на 8 (измерять в битах);
  - `perminute` — все числа умножать на 60 (измерять в единицах за минуту);
  - `perhour` — все числа умножаются на 3600 (измерять в единицах за час);
  - `transparent` — генерировать прозрачный фон картинки;
  - `gauge` — интерпретировать полученные значения как абсолютные. Полезно для отображения таких параметров, как загрузка процессора, дискового пространства;
  - `unknaszero` — трактовать неверные значения как 0, а не как повторение предыдущего значения;
- `kilo[target]:` *<число>* — что понимается под "кило". По умолчанию — 1000, можно установить 1024;

- `kMG[target]: <список-префиксов-множителей>` — какими буквами обозначать "кило", "мега" и т. п. По умолчанию: "K, M, G, T, P";
- `Colours[target]: Colour1#RRGGBB,Colour2#RRGGBB,Colour3#RRGGBB,Colour4#RRGGBB` — определение цветовой схемы, где `ColoursX` — текстовое имя цвета, помещаемое в легенду графика, `RRGGBB` — шестнадцатеричные значения, определяющие RGB-цвет;
- `Background[target]: #RRGGBB` — задает цвет фона;
- `YLegend[target]: <текстовая-строка>` — по умолчанию: "Bits per second";
- `ShortLegend[target]: <текстовая-строка>` — по умолчанию: "b/s".

Помимо MRTG существует еще один пакет аналогичного назначения — RRDtool.

Программа RRDtool

Программный пакет RRDtool (Round Robin Database) обеспечивает хранение и отображение данных мониторинга: загрузку каналов, температуру и любую другую зависящую от времени последовательность данных. Задумывалась как повторная, но более правильная реализация MRTG. Объем хранимых данных не увеличивается со временем — ячейки хранения используются циклически. В отличие от MRTG, программа не упаковывает старые данные самостоятельно, сбор информации и генерация HTML-кода также производится с помощью внешних средств. Параметры передаются в командной строке или через утилиту stdin.

Подсчет трафика

Иногда необходимо подсчитать трафик по клиентам, особенно когда организуется подключение домашней локальной сети или несколько небольших фирм совместно покупают выделенную линию для подключения к провайдеру. К сожалению, стопроцентного совпадения подсчитанного трафика с данными провайдера получить вряд ли удастся, поскольку приведенные далее способы подсчета трафика дают *разные* результаты. Правда, погрешность подсчета обычно не превышает 5%.

Есть несколько вариантов подсчета трафика:

- данные, взятые по SNMP (OutOctets на интерфейсе);
- по данным, взятым из Cisco;
- по данным, взятым из /proc/tty/driver/serial;
- по данным, взятым из radacct (radius-accounting/OutOctets);

- по ipchains;
- с помощью nacctd.

Далее приведен достаточно простой способ подсчета трафика с использованием ipchains.

Смысл метода такой — ставим разрешительную цепочку для обсчитываемого IP-адреса, например:

```
ipchains -A output -d AA.BB.CC.DD -j ACCEPT
```

Теперь можно посчитать байты:

```
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 mark outsize source
destination ports   4  308 ACCEPT  all  -- 0xFF 0x00  any ↵
                  anywhere AA.BB.CC.DD  n/a
```

Из примера видно, что клиенту ушло 308 байт. Со временем во втором столбике будет накапливаться статистика по байтам. Далее необходимо как-то обрабатывать эти данные и выводить себе и клиенту. Для этого можно воспользоваться программой, написанной на языке Perl и расположенной по адресу **linux.uatel.net.ua/ipcount.perl**.

Существует также пакет, предназначенный для подсчета IP-трафика через протокол SNMP. Он так и называется — "Универсальный счетчик IP-трафика через SNMP". Адрес пакета приведен в списке литературы и ссылок.

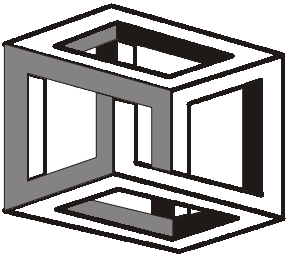
Помимо этих двух простых способов есть большое количество программ для подсчета трафика, в частности IpTraf, useripacct, netacct, ipacct и др.

Литература и ссылки

- ISP-Hookup-HOWTO.
- FIREWALLING_AND_PROXY_SERVER_HOWTO.
- THE_LINUX_KERNEL_HOWTO.
- **lin-omts.airport.sakhalin.ru/departs/ccito/guide1.htm** — как установить, настроить и запустить Web-узел UNIX не тратя лишних денег, сил и здоровья.
- **linux.perm.ru/doc/net/mrtg.html** — Богомолов С. Мониторинг загрузки каналов (и не только) MRTG.
- **linux.uatel.net.ua/ipcount.phtml** — как оперативно подсчитать IP-трафик клиента.
- **people.ee.ethz.ch/~oetiker/webtools/mrtg/mrtg.html** — описание MRTG.

- **rrdtool.eu.org** — официальный сайт пакета rrdtool.
- **ftp://ftp.kiev.farlep.net/pub/os/linux/soft/trafficcounter-snmp** — универсальный счетчик IP-трафика через SNMP.
- **www.bog.pp.ru/work/rrdtool.html** — Богомолов С. RRDtool — хранение и отображение данных мониторинга.
- **www.geocities.com/SiliconValley/Pines/7895/PPP.DOC** — Водолазкий В. Установка PPP-соединения в Linux.
- **www.linux.org.ru/books/gateway/** — Костарев А. Ф. ОС Linux как мост между локальной сетью и Internet.
- **www.mrtg.org** — официальный сайт пакета MRTG.

ГЛАВА 19



Учет сетевого трафика

Одна из забот сетевого администратора — учитывать сетевой трафик, проходящий через шлюз локальной сети. Причем задача учета трафика со временем начинает усложняться: сначала необходимо просто предоставить информацию о мегабайтах входящего и исходящего трафика, затем — статистика по месяцу, дальше — с раскладкой по часам и пользователям, потом нужно будет подключить тарификацию и т. д. На рынке существуют десятки программ биллинга (billing), но практически все они платные и зачастую стоят не одну сотню долларов. Для небольшой сети или небольшого интернет-провайдера такие затраты не оправданы, и систему учета трафика приходится разрабатывать самому.

В этой главе будет рассмотрено несколько способов организации учета трафика. Но практически все они опираются на такие предположения:

- расширенные свойства ядра Linux по управлению сетевым трафиком;
- в качестве маршрутизатора используется компьютер с ОС Linux;
- для обработки результатов используются скриптовые языки типа Perl либо специализированные программы и базы данных.

Простой учет трафика

Как обычно, начнем с простых вещей и постепенно перейдем к более сложным.

Итак, система учета трафика строится с использованием, по крайней мере, трех вещей:

- специально созданных правил для брандмауэра, установленного на нашем маршрутизаторе. Использовать будем iptables как более новую систему, но ничто не мешает воспользоваться и ipchains;

- программы снятия статистики;
- программы отображения статистики.

Наша простая система учета трафика обладает следующими возможностями:

- учитывает весь трафик, который проходит через маршрутизатор, с возможностью анализа трафика по хостам;
- изменение конфигурации без внесения изменений в программную часть;
- данные о трафике хранятся в базе данных MySQL, поскольку в дальнейшем это сильно упростит модернизацию и развитие системы.

Начнем описание системы со структуры базы данных.

Пусть наша база данных будет называться `traf`, а таблица, в которой находится вся накопленная информация по трафику — `traffic`. Структура таблицы будет иметь такой вид:

```
CREATE TABLE traffic (
  id int(11) NOT NULL auto_increment,
  date datetime NOT NULL,
  ip varchar(20) NOT NULL default '',
  in int(11) NOT NULL default '0',
  out int(11) NOT NULL default '0',
  KEY id (id)
) TYPE=MyISAM;
```

В таблице есть следующие поля:

- `id` — индекс записи в базе данных, автоматически увеличивается на единицу при записи новой строки в базу данных;
- `date` — содержит дату и время сохранения записи в базе данных;
- `ip` — IP-адрес, к которому относится информация по входящему и исходящему трафику;
- `in` — содержит информацию по входящему трафику;
- `out` — содержит информацию по исходящему трафику.

Далее необходимо заняться составлением правил для iptables, которые позволят учитывать трафик. Поскольку мы хотим добиться достаточно гибко настраиваемой системы учета трафика, то необходимо создать конфигурационный файл, в котором будут заданы IP-адреса хостов, по которым производится учет трафика. Также необходима возможность задания диапазона IP-адресов. В связи с этим, правила для iptables вынесем в отдельный скрипт, который будет вызываться при старте операционной системы.

Конфигурационный файл называется traffic.conf и имеет нижеприведенную структуру.

```
# Рабочие места
WS1="192.168.0.1"
WS3="192.168.0.3"
WS7="192.168.0.7"
# Сеть банка
BANKNET="192.168.2.0/24"
# Полный список адресов
ALLNETS="$WS1 $WS3 $WS7 $BANKNET"
```

Реально скрипт непосредственно использует параметр `ALLNETS`, но лучше расписать адреса по отдельности, а не записывать их все вместе.

Далее рассмотрим правила учета для iptables, которые находятся в файле rc.traffic. После отладки вызов этого файла можно добавить в rc.local.

```
#!/bin/bash
# Подключаем конфигурационный файл
. /etc/traffic.conf
# Функция для создания правила учета
addrule(){
/sbin/iptables -N ACCT_IN_$1  # Создаем правило для учета входящего трафика
/sbin/iptables -F ACCT_IN_$1  # Обнуляем счетчик для этого правила
/sbin/iptables -A INPUT   -j ACCT_IN_$1 # Включаем учет для этого правила
/sbin/iptables -A ACCT_IN_$1  -s $2 # Считаем входящим тот трафик,
                                    # у которого источник с адресом $2
/sbin/iptables -N ACCT_OUT_$1 # Создаем правило для учета исходящего
                              # трафика
/sbin/iptables -F ACCT_OUT_$1 # Обнуляем счетчик для этого правила
/sbin/iptables -A OUTPUT  -j ACCT_OUT_$1 # Включаем учет для этого
                                         # правила
/sbin/iptables -A ACCT_OUT_$1 -d $2 # Считаем исходящим тот трафик,
                                    # у которого получатель с адресом $2
/sbin/iptables -A FORWARD -j ACCT_IN_$1
/sbin/iptables -A FORWARD -j ACCT_OUT_$1
}
# Создаем правила для учета трафика
for NET in $ALLNETS; do
# Для всех сетей в списке $ALLNET создать правила учета трафика
addrule $NET $NET
done
```

Теперь можно протестировать наши правила. Выполним приведенный ранее скрипт и после этого для проверки наберем следующее:

```
iptables -L
```

Вы должны будете увидеть что-то похожее:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCT_IN_192.168.0.1  all  --  anywhere             anywhere
ACCT_IN_192.168.0.3  all  --  anywhere             anywhere
ACCT_IN_192.168.0.7  all  --  anywhere             anywhere
ACCT_IN_192.168.2.0/24  all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCT_IN_192.168.0.1  all  --  anywhere             anywhere
ACCT_OUT_192.168.0.1  all  --  anywhere             anywhere
ACCT_IN_192.168.0.3  all  --  anywhere             anywhere
ACCT_OUT_192.168.0.3  all  --  anywhere             anywhere
ACCT_IN_192.168.0.7  all  --  anywhere             anywhere
ACCT_OUT_192.168.0.7  all  --  anywhere             anywhere
ACCT_IN_192.168.2.0/24  all  --  anywhere             anywhere
ACCT_OUT_192.168.2.0/24  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCT_OUT_192.168.0.1  all  --  anywhere             anywhere
ACCT_OUT_192.168.0.3  all  --  anywhere             anywhere
ACCT_OUT_192.168.0.7  all  --  anywhere             anywhere
ACCT_OUT_192.168.2.0/24  all  --  anywhere             anywhere
```

После этого необходимо написать программу, которая будет снимать статистику и сохранять ее в базе данных. В нашем случае это будет простейшая программа, которую необходимо добавить в ctontab и вызывать с периодичностью, скажем, в 10 минут.

```
#!/usr/bin/perl
# Функция, занимающаяся сбором и внесением данных в БД
sub account{
$name=$_[0]; # Имя правила
$IP_IN=0;    # Инициализация счетчиков
$IP_OUT=0;
# Командная строка MySQL для внесения данных в таблицу
$mysqlcommand="/usr/bin/mysql -h localhost traf -e";
# Снимаем данные со счетчика входящего трафика и обнуляем
$ipstuff=`/sbin/iptables -L -Z ACCT_IN_$name -v -x`;
# Выделяем из вывода предыдущей команды значение счетчика
@IPTBMASS=split(/\n/,$ipstuff);
chomp $IPTBMASS[2];
```

```
$string=$IPTBMASS[2];
$string=~ s/\s{1,}/ /g;
@INFOMASS=split(/ /,$string);
$IP_IN=$INFOMASS[2];
# Снимаем данные со счетчика исходящего трафика и обнуляем
$ipstuff=`/sbin/iptables -L -Z ACCT_OUT_$name -v -x`;
# Выделяем из вывода предыдущей команды значение счетчика
@IPTBMASS=split(/\n/,$ipstuff);
$string=$IPTBMASS[2];
$string=~ s/\s{1,}/ /g;
@INFOMASS2=split(/ /,$string);
$IP_OUT=$INFOMASS2[2];
# Получаем текущее время
($min, $hours, $day, $mounth,$year) = (localtime)[1,2,3,4,5];
$time=$hours.":".$min.":00";
$mounth=$mounth+1;
$year=$year+1900;
$date=$year."-".$mounth."-".$day;
# Формируем SQL-запрос
$sql="insert into traffic values('','".$date."
".$time."','".$name."','".$IP_IN."','".$IP_OUT."');";
# Выполняем его
'$mysqlcommand "$sql"';
}

$config='./lconfreader.sh'; # Прочитаем конфигурационный файл.

# Далее приводится текст скрипта lconfreader.sh
# #!/bin/bash
# . ./lbiling.conf # Включить конфигурационный файл
# echo $ALLNETS    # Вывести список всех сетей, по которым ведется учет
chomp $config;
@NETMASS=split(/ /,$config);
foreach $nets(@NETMASS)
 {
  # Для каждого элемента списка выполнить функцию account
  account $nets;
 }
```

Все остальное в ваших руках, хотите — дописывайте свои модули визуализации, генераторы отчетов и тому подобное, хотите — используйте стандартные программы.

Учет трафика при помощи net-acct

Пакет net-acct является небольшой, но гибкой системой учета трафика, которая позволяет выполнять следующее:

- выдавать подробный отчет по трафику, время, адреса мест обмена информацией (откуда и куда), порты, сетевой интерфейс;
- работать с базой данных MySQL.

Также существует возможность разделять трафик по типу: бесплатный (free), пиринговый (peer) и общий (world).

При конфигурировании и установке net-acct необходимо задать опцию `--with-mysql` с указанием пути к MySQL:

```
./configure --with-mysql=/usr/local/mysql
```

После компиляции и установки необходимо создать базу данных для net-acct с помощью следующей команды:

```
mysql -u root -p < netacct.sql.
```

Далее добавляем пользователя в MySQL и предоставляем ему права на базу `netacct`:

```
mysql>grant all privileges on netacct.* to netacct identyfied by 'netacct';
mysql>flush privileges;
```

После этого приступаем к конфигурированию. Используются два файла конфигурации, расположенные в каталоге /usr/local/etc:

- naccttab — главный файл конфигурации;
- nacctpeering — файл, содержащий список сетей, которые следует считать пиринговыми.

Naccttab

В листинге 19.1 представлен пример файла naccttab, который мы подробно разберем.

Листинг 19.1. Файл naccttab

```
database mysql
mysql_user netacct
mysql_password netacct
mysql_host localhost
mysql_database netacct
mysql_table accounting
pidfile  /var/run/nacctd.pid
```

```
flush 60
fdelay 60
dumpfile /var/log/net-acct-dump
notdev eth0
device eth0
iflimit eth0
ignorenet 127.0.0.0 255.0.0.0
debug 2
debugfile /tmp/nacctd.debug
headers lo    14     12
headers eth   14     12
hostlimit 192.168.0.1
hostlimit 192.168.0.3
hostlimit 192.168.0.7
```

Рассмотрим параметры, используемые в файле naccttab:

- `database mysql` — определяет тип хранения информации: это либо база данных (`mysql`), либо LOG-файл (`file`), при котором запись информации происходит в текстовый файл /var/log/net-acct;
- `mysql_user netacct` — определяет имя пользователя MySQL, от имени которого работает программа;
- `mysql_password netacct` — пароль пользователя;
- `mysql_host localhost` — задает хост, на котором находится MySQL;
- `mysql_database netacct` — имя базы данных;
- `mysql_table accounting` — имя таблицы в базе данных, в которой накапливается информация;
- `pidfile  /var/run/nacctd.pid` — определяет файл, в котором находится PID процесса nacctd.

Далее идут сетевые настройки. В нашем случае на маршрутизаторе находится две сетевые карты, `eth0` и `eth1`, причем `eth0` — шлюз в Интернет, `eth1` — подключение маршрутизатора к локальной сети.

- `flush 60` — задает период (в секундах), с которым данные записываются в базу.
- `fdelay 60` — задает время (в секундах), в течение которого после завершения сетевой активности данные о трафике могут быть записаны в базу.
- `dumpfile /var/log/net-acct-dump` — файл статистики.
- `notdev eth0` — если маршрутизатор имеет два интерфейса, то с помощью опции `notdev` мы указываем интерфейс, который следует исключить из диапазона отслеживаемых программой, дабы избежать дублирования записей о трафике, прошедшем через оба интерфейса.

- `device eth0` — задает интерфейс, который будет отслеживаться на предмет учета трафика.
- `iflimit eth0` — используется в том случае, когда маршрутизатор имеет несколько интерфейсов и только на одном из них необходим учет трафика.
- `ignorenet 127.0.0.0 255.0.0.0` — позволяет исключить из подсчета трафик, идущий через Loopback-интерфейс.
- `debug 1` — задает уровень отладочной информации.
- `debugfile /tmp/nacctd.debug` — задает файл, в котором записывается отладочная информация.

Опции для отладочной информации определяют, с пакетами каких интерфейсов следует работать, в нашем случае `lo` и `eth`.

- `headers lo 14 12`
- `headers eth 14 12`

Цифры после каждого из параметров указывают следующее: в первом столбце — количество байт, после которых начинают идти данные в пакете, во втором — количество байт, приходящихся на поле типа пакета.

Последние строчки листинга 19.1 определяют, для каких IP-адресов производить учет трафика.

- `hostlimit 192.168.0.1`
- `hostlimit 192.168.0.3`
- `hostlimit 192.168.0.7`

Nacctpeering

В файле nacctpeering содержится список пиринговых сетей, которые учитываются в статистике отдельно.

Например:

```
195.58.1.134
194.226.148.0/24
213.140.111.224/255.255.255.224
```

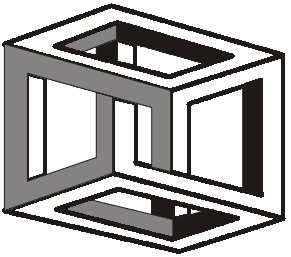
Для запуска системы учета достаточно выполнить команду `/usr/local/sbin/nacctd`. Если не было допущено ошибок при конфигурировании, то через минуту в базе уже появятся первые записи и весь трафик будет под контролем.

Для анализа информации необходимо создать программу, которая будет делать запросы к базе данных и выводить информацию в нужном виде.

Литература и ссылки

- IPTABLES HOWTO.
- **exorsus.net/projects/net-acct/** — домашняя страница проекта net-acct.
- **www.vadim.org.ua//index.php?cmd=article3** — пример простой системы учета трафика.

ГЛАВА 20



Виртуальные частные сети

Виртуальные частные сети (Virtual Private Network, VPN) — общее наименование группы протоколов, программного, а иногда и аппаратного обеспечения, позволяющих организовывать соединение независимых сетей через небезопасные (Интернет) сети общего пользования, прозрачно для пользователей сети. В связке с VPN часто звучит термин туннель, туннелирование (tunneling). Этот термин подразумевает создание между двумя точками (обычно из разных локальных сетей, которые объединяют в VPN) шифрованного канала, позволяющего безопасно пересылать данные из одного сегмента сети в другой.

Программное обеспечение может быть самым разнообразным, начиная от стандартных программ маршрутизации, PPP, SSH и заканчивая специализированными программно-аппаратными комплексами с аппаратурой шифрования и многоуровневыми протоколами проверки целостности и безопасности соединения.

В этой главе будут рассмотрены несколько вариантов организации VPN, но поскольку VPN характерно для корпоративного мира, то реализации предусматривают шифрование трафика между точками VPN и взаимодействие с операционными средами семейства Windows.

Чаще всего применяются реализации VPN, использующие следующие протоколы:

- SSH;
- IPSec (Internet Protocol Security);
- PPTP (Point-To-Point Tunneling Protocol).

Протокол SSH обычно используется для быстрой организации защищенного канала, который применяется для доступа к удаленным системам, отправки и получения файлов.

В промышленных масштабах предпочитают использовать следующие два протокола:

- IPSec реализован в проекте FreeS/WAN. Его обычно используют для объединения фрагментов корпоративных сетей с помощью общедоступных сетей. При этом данные, передаваемые из одной сети в другую, шифруются исходящим шлюзом при передаче через сети общего пользования и преобразуются в исходный вид принимающим шлюзом;
- PPTP считается устаревшим и не совсем безопасным. Тем не менее он используется Microsoft, поэтому распространен в корпоративной среде.

Организацию туннеля с помощью SSH в этой главе мы рассматривать не будем, поскольку SSH был описан ранее, а настройка маршрутизации не составляет особых проблем.

Протокол IPSec

IPSec — это расширение протокола IP, которое предоставляет не только шифрование, но и аутентификацию в транспортном слое. IPv6 поддерживает протокол IPSec изначально, т. к. IPSec является требованием спецификации IETF для протокола IPv6.

IPSec по сути является набором протоколов. Три протокола используются для инкапсуляции, шифрования и аутентификации: AH (Authentication Header, заголовок аутентификации), ESP (Encapsulating Security Payload, инкапсулированные защищенные данные) и IKE (Internet Key Exchange, обмен ключами через Интернет). IPSec прозрачен для пользователей и приложений, поэтому для его использования не требуется переделка программного обеспечения.

Протоколы AH и ESP отвечают за шифрование и аутентификацию. AH добавляется после заголовка IP, но перед данными. AH содержит аутентификационную информацию, обычно это ключи в MD5 (Message Digest, профиль сообщения[1]) или SHA (Secure Hash Algorithm, алгоритм безопасного хэша). Протокол AH используется только для аутентификации и не производит шифрования.

Протокол ESP может использоваться как для шифрования, так и для аутентификации. Он может применяться как с AH, так и без него. Ключи шифрования распределяются с помощью IKE. IKE устанавливает параметры соединения, включая инициализацию, обработку и обновление ключей шифрования. Аутентификация производится посредством общих секретных фраз или криптографических RSA-ключей, которые гарантируют идентичность

[1] Профиль сообщения — короткая цифровая строка фиксированной длины, формируемая из более длинного сообщения с использованием специального алгоритма. — *Ред.*

обеих частей. IKE основан на методе Diffie-Hellman обмена идентификационными метками (tokens). Для шифрования данных используются алгоритмы группового шифрования, такие как тройной DES (Data Encryption Standard, стандарт шифрования данных). Хэш-алгоритмы, такие как MD5 и SHA, обеспечивают аутентификацию каждого пакета. Для дополнительной безопасности соединения через малые промежутки времени происходит обновление аутентификационных ключей.

IPSec вставляет заголовок и зашифрованные полезные данные в обычный IP-пакет. Это позволяет данным IPSec проходить через любые IP-сети. IPSec реализован как в программном, так и в аппаратном виде. Хотя практически все реализации IPSec соответствуют существующим RFC, они совсем не обязательно смогут взаимодействовать. Необходимо тестировать реализации IPSec от различных производителей, чтобы убедиться, что они полностью совместимы.

VPN-сервер FreeS/WAN

Как уже упоминалось ранее, FreeS/WAN использует протокол IPSec, описанный в предыдущем разделе.

FreeS/WAN состоит из двух частей:

- патч KLIPS (KerneL IP Security) является дополнением стандартного ядра Linux;
- демон pluto обрабатывает запросы аутентификации протокола IKE и взаимодействует в ядре с компонентом KLIPS, который отвечает за инкапсуляцию и шифрование.

Программа ipsec управляет и осуществляет работу протокола IPSec. Программа ipsec используется для активизации и отключения туннелей, а также для опроса состояния каждого туннеля IPSec.

IPSec может быть использован для создания защищенных туннелей хост-хост, подсеть-подсеть, или подсеть-хост. У IPSec есть два режима: транспортный и туннельный. Транспортный режим обеспечивает защиту только дейтаграмм (datagram) между источником и адресатом. Он производит аутентификацию, инкапсуляцию и шифрование только IP-данных, но оставляет неизменными транспортные заголовки. Поэтому транспортный режим обычно применяется для создания зашифрованных туннелей хост-хост. В туннельном режиме создается новый IP-заголовок и вся оригинальная дейтаграмма инкапсулируется, скрывая информацию об оригинальном отправителе. Инкапсуляция позволяет пакетам из одной сети туннелироваться через другие сети. Одно из применений — это использование IPSec-шлюзов, соединяющих доверительные частные сети через общедоступные.

Конфигурирование FreeS/WAN осуществляется с помощью файлов ipsec.conf и ipsec.secrets.

Ipsec.conf

Файл ipsec.conf, как и все конфигурационные файлы, находится в каталоге /etc. Он является текстовым файлом и содержит одну или более секций. Строка, которая начинается с символа #, — комментарий. Пример конфигурационного файла приведен в листинге 20.1.

Листинг 20.1. Файл ipsec.conf

```
# Определяем секцию для конфигурирования сетевых настроек
config setup
# Сетевой интерфейс, используемый для организации
# VPN-соединения
interfaces="IPSEC0=eth0"
#Разрешение выдачи отладочных сообщений — в нашем случае -
# запрещаем выдачу сообщений для части ядра и для pluto
klipsdebug=none
plutodebug=none
# Автоматическая установка соединений и аутентификация при
# запуске IPSec

plutoload=%search
plutostart=%search
# Параметры соединения между локальными сетями
# Название соединения

conn Test

# Данные для 1 шлюза
# IP-адрес
left=193.1.1.2
# Описание локальной сети
leftsubnet=192.168.1.0/24
# IP ближайшего к 1 шлюзу маршрутизатора
leftnexthop=194.17.2.5
# Исходные данные для 2 шлюза
# IP-адрес
right=197.11.0.213
# Описание локальной сети
rightsubnet=192.168.1.0/24
# IP ближайшего к 2 шлюзу маршрутизатора
rightnexthop=202.22.8.24
# Количество попыток проверки ключей
# 0 — до достижения положительного результата
keyingtries=0
```

```
# Тип аутентификации (AH или ESP)
auth=ah
# Устанавливать соединение при запуске IPSec
auto=start
```

Разберем приведенный в листинге 20.1 конфигурационный файл.

- `config setup` — слово `config` задает тип секции, а `setup` — метка секции. Обычно тип секции — это `config`, определяющий системные настройки FreeS/WAN, или `conn`, задающий параметры каждого VPN-туннеля. Каждый VPN-туннель должен иметь собственную секцию `conn`.
- `interfaces="IPSEC0=eth0"` — задает интерфейсы, используемые для организации VPN-соединений.
- `klipsdebug=none` — используется для разрешения/запрета выдачи отладочных сообщений.

Примечание

Значения параметров, начинающиеся с `%`, обозначают системную переменную, загружаемую FreeS/WAN при образовании туннеля.

Следующие параметры позволяют автоматически искать и устанавливать соединения при загрузке и старте демона:

- `plutoload=%search`
- `plutostart=%search`

Pluto сканирует файл ipsec.conf в поисках соединений, которые будут загружены, и выдает доступ к VPN-туннелям. Явно можно задать установку соединения следующим образом: `plutoload="Test"`, где `Test` — имя секции, в которой описан нужный нам туннель.

Далее идет секция `Test`, описывающая параметры нашей виртуальной сети — ее левую и правую части. Тут все просто: IP-адрес шлюзов, маски подсети, адреса ближайших для данной "половинки" маршрутизаторов.

- `keyingtries=0` — говорит IKE бесконечно продолжать попытки обмена ключами при потере соединения.
- `keyexchange=ike` — устанавливает IKE как механизм по умолчанию для обмена ключами.
- `keylife=24h` — время между сменой ключей.
- `auth=ah` — тип аутентификации.
- `authby=secret` — метод аутентификации.

Ipsec.secrets

FreeS/WAN поддерживает два формата ключей, используемых демоном pluto для проверки подлинности соединений длиной до 256 бит. Каждый из

этих форматов требует некоторых дополнительных действий по созданию ключей и изменению конфигурационных файлов. В случае использования открытых зашифрованных ключей, для создания нового ключа на одном из шлюзов выполните команду:

```
ipsec ranbits 256 > /root/key
```

Теперь файл /root/key содержит новый ключ, который необходимо прописать в файлы /etc/ipsec.secrets на обоих шлюзах, для чего добавьте в них следующую строку:

```
194.17.2.5 197.11.0.213↳
"0xaf4a2a4c_f58a444f_5a55d31e_55555ac4_555a58e2_b6ea25a3_0ee661d4_daf155"
```

Она представляет собой IP-адреса левой и правой части шлюза и сгенерированный нами ранее ключ.

В случае использования RSA-ключей, необходимо выполнить следующие операции для файлов ipsec.conf и ipsec.secrets:

1. Создайте ключи RSA для каждого шлюза командой:

   ```
   ipsec rsasigkey --verbose 1024 > /root/leftey
   ```

2. В файлах /etc/ipsec.conf на обоих шлюзах для обеспечения возможности использования RSA-ключей необходимо добавить следующие строки:

   ```
   authby=rsasig
   leftrsasigkey= ""
   rightrsasigkey= ""
   ```

   где в опциях `leftrsasigkey` и `rightsasigkey` задаются открытые RSA-ключи. А закрытые ключи прописываются в файле /etc/ipsec.secrets.

Более подробную информацию по алгоритмам шифрования и их использованию в FreeS/WAN смотрите в документации к программе.

MS Windows NT VPN (PPTP)

Фирма Microsoft в своих продуктах использует протокол PPTP (Point-to-Point Tunneling Protocol) для организации VPN. Об особенностях этого протокола мы не будем рассказывать, отметим лишь, что PPTP позволяет шифровать передаваемую информацию с ключом длиной 128 бит.

Итак, у нас возникают две задачи:

1. Организация PPTP сервера на Linux для подключения Windows-клиентов.
2. Подключение Linux-клиентов к Windows NT VPN-серверу.

Linux PPTP-сервер

Устанавливаем pptpd, который входит в большинство современных дистрибутивов. Проверяем по ntsysv, что сервис pptpd запускается по умолчанию.

Конфигурационный файл /etc/pptpd.conf в бльшинстве случаев имеет следующий вид:

```
speed 115200
option /etc/ppp/options.pptpd
debug
localip 192.168.0.1
remoteip 192.168.0.100-150
```

В этом файле мы задаем скорость соединения, файл, в котором хранятся настройки PPP для PPTP, разрешаем выдачу отладочной информации, определяем локальный адрес нашей VPN и диапазон адресов, который будет назначаться VPN-клиентам.

Содержимое файла /etc/ppp/options.pptpd имеет следующий вид:

```
lock
mtu 1490
mru 1490
ipcp-accept-local
ipcp-accept-remote
lcp-echo-failure 3
lcp-echo-interval 5
deflate 0
auth
+chap
-pap
proxyarp
ms-dns 192.168.0.1
+chapms
+chapms-v2
nobsdcomp
nodeflate
nodefaultroute
+mppe-128
+mppe-stateless
```

В данном файле мы задаем размер пакета, тайм-ауты, типы авторизации и т. п. Особо следует отметить строку `+mppe-128`. Она разрешает 128-битовое шифрование.

Поле этого в файл /etc/ppp/chap-secrets заносим логины и пароли клиентов. Вот и все, можно пользоваться.

Linux PPTP-клиент

PPTP-клиент реализует протокол PPTP, предназначенный для подключения систем с Linux к VPN-сетям через MS Windows NT VPN-сервер. Для этого необходимо установить PPTP Client (обычно в дистрибутивах пакет называется pptp-linux) и программное обеспечение, реализующее протокол MPPE (Microsoft Point-To-Point Encryption) и находящееся в пакете ppp-mppe.

После установки ppp-mppe в файле /etc/modules.conf должны быть следующие строки:

```
alias char-major-108 ppp_generic
alias ppp-compress-18 mppe
```

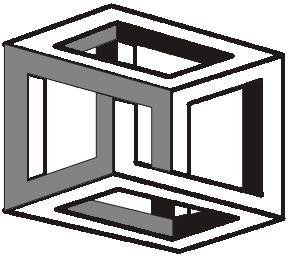
Далее запускаем файл /usr/sbin/pptp-command, выбираем в меню **Конфигурирование** и заполняем его поля (**IP-адрес**, **DNS**, **Имя системы**, **Тип авторизации** и т. п.)

Для запуска и останова VPN-соединения используется тот же файл pptp-command.

Литература и ссылки

- **securitylab.ru/34649.html** — Разумов М. По материалам Sys Admin Magazine. Администрирование IPSec VPN под Linux. Часть первая.
- **securitylab.ru/34764.html** — Разумов М. По материалам Sys Admin Magazine. Администрирование IPSec VPN под Linux. Примеры конфигураций.
- **www.bruy.info/vpn.html** — Бруй В., Карлов С. Linux-сервер: пошаговые инструкции инсталляции и настройки.
- **www.freeswan.org** — официальный сайт FreeS/WAN.
- **www.multik.ru/linux/linuxvpn/** — Калошин В. Установка VPN Linux-сервера.
- **www.opennet.ru/base/net/vpn_pptp.txt.html** — Коптев Д. Настройка VPN (PPTP) сервера под Linux.

ГЛАВА 21



Бездисковые компьютеры

Эта глава посвящена использованию бездисковых компьютеров в вашей сети. Мы рассмотрим следующие варианты:

- сервер — Linux, клиент — DOS/Windows 3.*x*;
- сервер — Linux, клиент — Linux.

Что такое бездисковый компьютер

Бездисковый компьютер представляет собой устройство, в идеале вообще лишенное всяких механических накопителей: жесткого диска, дисковода, CD-ROM, Zip Drive и т. п. Информацию такой компьютер получает из сети, туда же он информацию и отдает. Помимо этого, бездисковый компьютер в своем максимально идеализированном виде загрузку операционной системы также производит по локальной сети (или из Интернета). Конечно, существуют некоторые модификации идеи бездискового компьютера. В частности, компьютер может иметь привод CD-ROM, с которого и происходит загрузка операционной системы, или у компьютера вместо жесткого диска установлен Flash-накопитель небольшой емкости, содержащий операционную систему. Однако в дальнейшем я буду описывать сетевой компьютер в "чистом" виде, а с различными его модификациями читатель может самостоятельно ознакомиться, используя ссылки в конце главы.

Преимущества использования бездискового компьютера

Во-первых — некоторая экономия денег. Если компьютер лишить дисковода, привода CD-ROM, жесткого диска, то по сегодняшним ценам экономия на этих устройствах составит порядка 90 долларов на компьютер. 3—4 де-

сятка бездисковых компьютеров — и вы можете приобрести неплохой сервер для рабочих групп.

Во-вторых — увеличивается надежность (наработка на отказ) компьютеров. За счет того, что практически полностью устраняются подвижные части (остаются только вентиляторы охлаждения, но и этот вопрос решаем), увеличивается надежность компьютера. Не будем приводить здесь статистику отказов, уважаемый читатель может это проделать самостоятельно, в целом за гарантийный период, выход из строя жесткого диска, привода CD-ROM или дисковода составляет от 5 до 15%, в зависимости от производителей комплектующих.

В-третьих — мы экономим на системном администраторе (или освобождаем его время для других задач). Поскольку бездисковые станции загружают свою операционную систему с сервера, достаточно 1 раз настроить программное обеспечение и затем при необходимости просто вносить изменения в одном месте, не бегая между компьютерами.

В-четвертых — многие задачи можно переложить на сервер, поэтому в качестве бездисковых компьютеров могут выступать слабые по современным меркам машины класса Pentium или даже I486.

В-пятых — увеличение безопасности сети и данных. На бездисковом компьютере нет жесткого диска, дисководов, приводов CD-ROM — никто не занесет вирус, не перепишет секретные данные, не сможет установить программное обеспечение, идущее вразрез с политикой компании.

Недостатки использования бездискового компьютера

Первый и один из самых основных недостатков — несамостоятельность сетевого компьютера. Если произойдет останов сервера удаленной загрузки, то наши бездисковые компьютеры не смогут самостоятельно загрузить операционную систему и начать работать.

Второй недостаток, вытекающий частично из первого, — психология собственника: "Мне спокойнее, когда мое находится при мне". Как результат — пользователь все равно требует жесткий диск и другие накопители.

Третий — наличие физических ограничений сети. Современные операционные приложения и программы занимают десятки и сотни мегабайт, поэтому неоптимизированное программное обеспечение становится камнем преткновения при эксплуатации бездисковых компьютеров.

Четвертый — ограниченность применения для работы с большими объемами данных: графикой, расчетами, требующими обработки сотен мегабайт данных за очень короткое время. Помимо этого, архитектура операционных систем также может быть не приспособлена для использования при удаленной загрузке.

Области применения

Корпоративный мир. Зачем оператору банка мощный компьютер, если он только вводит цифры и получает информацию? Тем более что этих операторов в банке сотни. Или кассир в супермаркете, или операторы бюро ремонта в телефонных компаниях. Можно придумать достаточно много работ (те же терминалы в библиотеках), для которых не нужен полноценный компьютер.

Учебные заведения. Для наших (и не только) учебных заведений является нормой, когда операционные системы переустанавливаются чуть ли не ежедневно, после шаловливых ручек учащихся. Опять же, к сожалению, наши учебные заведения ограничены в средствах, поэтому та экономия, которую можно достичь, используя бездисковые рабочие станции, вряд ли окажется лишней.

Процесс загрузки бездискового компьютера

Для грамотной настройки бездискового компьютера необходимо знать всю технологическую цепочку его загрузки.

Как и большинство современных решений (находится в эксплуатации где-то с конца 70-х годов), процесс загрузки построен по технологии "клиент-сервер". На сервере установлено специальное программное обеспечение. А что же находится на клиентском компьютере? Ведь там ничего нет, кроме материнской платы и сетевой карты?

Если вы держали в руках сетевую карту, то наверняка видели, что на ней установлена панель под микросхему. Именно в этот разъем должна быть вставлена микросхема ПЗУ, содержащая код, отвечающий за процесс загрузки информации на бездисковую станцию.

При успешной сетевой загрузке бездисковый компьютер должен получить:

- идентификатор, однозначно определяющий этот компьютер в сети;
- образ операционной системы;
- файловую систему, с которой этот компьютер будет работать.

Давайте рассмотрим все это подробно.

Как уже говорилось в *гл. 1*, в сети TCP/IP хост однозначно определяется его IP-адресом. Однако у бездисковой станции ДО загрузки и инициализации какого-либо программного обеспечения, способного работать со стеком TCP/IP, IP-адреса нет и быть не может. Вот тут и используется MAC-адрес сетевого устройства, который в пределах сети уникален.

Получить образ операционной системы позволяет программа, находящаяся в микросхеме ПЗУ, установленной на сетевой карте. Сначала эта программа

посылает по сети MAC-адрес и запрос на получение необходимых данных для функционирования сетевого протокола (IP, IPX или NetBIOS — в зависимости от того, какая операционная система установлена и с каким сетевым протоколом она работает). Поскольку мы рассматриваем Linux и IP-протокол, то дальнейший материал касается только IP-протокола.

Протоколы, используемые для получения IP-адреса бездисковым компьютером, называются загрузочным протоколом (BOOTP) и протоколом динамической настройки компьютера (DHCP). Применение протокола DHCP шире, он используется и для динамической настройки обычных компьютеров *(см. гл. 4)*.

Поскольку первоначально бездисковый компьютер посылает по сети широковещательный запрос, то первый сервер удаленной загрузки, который откликнулся на запрос, выдаст бездисковому компьютеру IP-адрес и в дальнейшем произведет его загрузку.

После получения IP-адреса бездисковый компьютер получит образ операционной системы с сервера удаленной загрузки. Для этого используется протокол, имеющий название тривиального протокола передачи файлов (Trivial File Transfer Protocol, TFTP). Протокол TFTP можно назвать подмножеством протокола FTP, однако в нем нет подтверждения подлинности, и он использует протокол UDP, поскольку код протокола UDP легко разместить в микросхемах ПЗУ.

Передача данных происходит поблочно, после передачи каждого блока сервер удаленной загрузки ожидает подтверждения получения блока. Потерянные блоки после определенного времени ожидания передаются заново. Когда получены все блоки, микросхема ПЗУ сетевой загрузки обращается к образу операционной системы по адресу точки входа.

Последнее из требований — для нормального функционирования операционной системы компьютеру должна быть предоставлена корневая файловая система (для Linux и UNIX) или сетевые диски для других операционных систем. Linux и UNIX обычно используют сетевую файловую систему (NFS). Об этом будет рассказано далее.

Предварительные действия

Поскольку в качестве сервера удаленной загрузки мы используем в обоих случаях Linux, то сначала рассмотрим конфигурирование сервера удаленной загрузки.

Мы должны установить на сервере определенное программное обеспечение:

- Etherboot;
- TFTP;
- BOOTP или DHCP;
- NFS.

Помимо этого, если на Linux-бездисковой станции планируется использование графических сред, на сервере необходимо установить X Window.

Установка и настройка программного обеспечения на сервере

Установка пакета Etherboot тривиальна. Для Windows-клиента необходимо установить и настроить пакет Samba. Подробную информацию о пакете Samba можно найти и в *гл. 14*.

Далее следует установить следующие пакеты — сервер BOOTPD и сервер TFTP. Вместо BOOTPD можно использовать более универсальный DHCP. После установки пакетов для автоматического старта демона BOOTPD необходимо добавить следующую строчку в файл /etc/inetd.conf:

```
bootps dgram udp wait root /usr/sbin/tcpd bootpd
```

Затем надо создать BOOTP-базу, ставящую в соответствие MAC-адресам сетевых карт бездисковых компьютеров адреса IP и хранящую другую необходимую информацию (более подробную информацию следует смотреть в соответствующей man-странице). Эта база находится в файле /etc/bootptab и для нашего случая содержит следующие строки:

```
client1:hd=/tftpboot:vm=auto:ip=192.168.0.10:\
:ht=ethernet:ha=008048e2eb9c:\
:bf=bootnet
```

Рассмотрим подробнее поля базы:

- `hd` — домашний каталог, где находится загрузочный образ;
- `ht` — тип устройства;
- `ha` — аппаратный адрес хоста. Для Ethernet-карты это MAC-адрес;
- `ip` — адрес для бездискового клиента;
- `bf` — имя загрузочного образа для бездисковой станции.

Для автоматического запуска сервера TFTP необходимо добавить следующую строку в файле /etc/inetd.conf:

```
tftp dgram udp wait root /usr/sbin/tcpd in.tftpd -s /tftpboot
```

Linux-клиент

Поскольку книга о Linux, с него и начнем. Процесс условно можно разделить на три части:

1. Создание загрузочной дискеты — эмулятора ПЗУ сетевой карты, и отработка процесса первичной загрузки бездискового компьютера.

2. Компиляция ядра Linux для бездискового компьютера и настройка службы NFS.
3. Если клиент использует графическую оболочку, конфигурация X Window для работы с бездисковыми компьютерами.

Создание загрузочного ПЗУ (загрузочной дискеты)

Пакет Etherboot в своем составе имеет большое количество образов ПЗУ для разных типов сетевых карт (полный список сетевых карт, для которых имеются образы ПЗУ, следует смотреть в документации к Etherboot).

Следующее, что необходимо сделать, перед тем как программировать ПЗУ, проверить, действительно ли имеющаяся карта будет корректно работать с такой прошивкой ПЗУ. Для этого необходимо сделать специальную дискету, на которой записана загрузочная программа и образ ПЗУ для нашей карты. Затем создаем загрузочную дискету — эмулятор ПЗУ.

Замечание

Существуют стандартизованные фирмой Intel сетевые карты, поддерживающие протокол бездисковой загрузки PXE. Их выпускают фирмы Intel, 3COM, D-Link. Однако такие карты заведомо дороже.

Для создания загрузочной дискеты предусмотрена специальная маленькая программа (512 байт), которая загружает блоки с дискеты в память и начинает выполнение. Чтобы создать загрузочную дискету, надо только соединить загрузочный блок с соответствующим образом микросхемы ПЗУ. Для чего используется следующая команда:

```
cat floppyload.bin <файл_образа_ПЗУ> > /dev/fd0
```

После этого можно пробовать загрузить наш клиентский компьютер с полученной дискеты. Вы должны увидеть сообщение о старте TFTP-сервиса, о получении вашей бездисковой машиной IP-адреса и сообщение об отсутствии загрузочного образа.

Не надо пугаться этого сообщения — мы убедились, что образ ПЗУ благополучно загрузился, стартовала удаленная загрузка, но не был найден образ загрузочной дискеты. Все правильно, мы ведь ее еще не создали.

Теперь берем микросхему ПЗУ модели 2764 с ультрафиолетовым стиранием (можно, конечно, и советский аналог К273РФ6, но сейчас проще найти деталь производства Intel или других зарубежных производителей), берем программатор, файл с образом ПЗУ и программируем микросхему ПЗУ.

Наконец, у нас есть запрограммированная микросхема ПЗУ. Теперь необходимо разрешить карте работать с ПЗУ. Для этого на карте, конфигурируе-

мой перемычками, необходимо включить перемычку BOOTROM ENABLED и выставить перемычками адрес блока памяти, куда будет отображаться ПЗУ (как правило, это адрес D000, D400). Важно, чтобы этот адрес не был использован BIOS. Для сетевой карты без перемычек в комплекте с драйверами идет программа конфигурации и тестирования сетевой карты. Документация по программе конфигурации сетевой карты находится на прилагаемой к ней дискете. После успешной конфигурации сетевой карты вставьте микросхему ПЗУ в разъем.

Замечание

Некоторые сетевые карты для правильной работы удаленной загрузки требуют определить в BIOS диск A: (все равно, какого типа) либо разрешить производить загрузку компьютера с устройства, не входящего в стандартный список загрузочных устройств.

Настройка сервера

Необходимо настроить на сервере удаленной загрузки три службы: BOOTP (или DHCP), TFTP и NFS. Процессы установки и настройки указанного программного обеспечения подробно описаны в документации, входящей в каждый из пакетов.

Для нормального процесса загрузки бездискового клиента необходимо настроить разделы NFS.

Исходя из требований надежности и защищенности локальной сети, нежелательно использовать корневую файловую систему сервера в качестве файловой системы бездискового компьютера, тем более что для бездискового клиента необходимо создать свои, специфические файлы конфигурации.

В идеале, чтобы создать корневую файловую систему, вам надо знать, какие файлы требуются дистрибутиву вашей операционной системы. При загрузке необходимы файлы устройств, файлы, находящиеся в каталоге /sbin и /etc. Однако проще сделать копию существующей файловой системы и изменить в ней некоторые файлы для бездискового компьютера. В дистрибутиве Etherboot есть руководство и ссылки на скрипты, которые создают такую файловую систему на бездисковом компьютере из корневой файловой системы сервера.

Необходимо в файл /etc/exports на сервере вставить следующую строку:

```
/tftpboot/192.168.0.10 test.foo.com(rw,no_root_squash)
```

Для некоторых служб нужны права `rw`. Атрибут `no_root_squash` защищает систему NFS от отображения идентификатора суперпользователя в какой-либо другой. Если этот атрибут не будет задан, то различные демоны могут не заработать.

Теперь запустите службы NFS (rpc.portmap и rpc.mountd) и снова попробуйте бездисковую загрузку. Если все прошло удачно, то ядро сможет подмонтировать корневую файловую систему и пройти все стадии загрузки до по-

явления приглашения входа в систему. Вполне вероятно, что по ходу загрузки у вас будут выдаваться сообщения о проблемах с некоторыми службами. Так и должно быть. Дистрибутивы Linux ориентированы на операции с диском, и поэтому для бездисковой загрузки требуются небольшие изменения. Самой большой неприятностью является зависимость от файлов, находящихся в каталоге /usr во время загрузки — они в процессе загрузки поступают от сервера немного позже. Для решения этой проблемы измените пути таким образом, чтобы необходимые файлы искались в корневой файловой системе.

Конфигурация клиента

Для правильной работы клиента необходимо скомпилировать ядро операционной системы Linux с поддержкой корневой файловой системы на NFS. Кроме того, следует разрешить получение ядром IP-адреса из запроса BOOTP (DHCP). Надо также скомпилировать драйвер для вашей сетевой карты в ядро. Для уменьшения объема ядра можно отключить лишние свойства и опции.

Ядро, полученное после компиляции, необходимо преобразовать в загрузочный образ. Это делается аналогично тому, как мы создавали загрузочный образ дискеты для DOS. Для создания образа воспользуйтесь утилитой mknbi-linux. После создания загрузочного образа поместите его в каталог /tftpboot под именем, определенным в /etc/bootptab. Убедитесь, что файл доступен для чтения любому пользователю, потому что у TFTP-сервера нет специальных привилегий.

Дальнейшая проверка загрузки бездискового клиента должна подтвердить правильность наших настроек.

При желании достаточно просто из текстовой бездисковой станции сделать X терминал. X-терминал — это специализированный компьютер, который сконфигурирован таким образом, чтобы выполнять программу, называемую X-сервер. Это позволяет программам, выполняющимся на другом компьютере, использовать в качестве устройства ввода/вывода маломощный X-терминал.

В нашем случае, после загрузки ядра операционной системы мы должны запустить на выполнение соответствующим образом настроенный X-сервер.

Windows-клиенты

Для создания бездисковых DOS/Windows-клиентов необходимо определенное программное обеспечение:

- Microsoft Network Client version 3.0 for MS-DOS;
- MS-DOS 5.0 или выше;
- Windows 3.1*x*.

Установка и настройка программного обеспечения на клиенте

Компьютер, на котором будут проводиться опыты, должен удовлетворять следующим минимальным требованиям:

- процессор 386;
- оперативная память 2 Мбайт;
- винчестер 20 Мбайт;
- дисковод 3,5 дюйма;
- сетевая карта.

Наша задача — установить и отконфигурировать программы на клиентской машине, чтобы позднее перенести их на бездисковые клиентские компьютеры.

Прежде всего, для подключения к разделяемым ресурсам сервера (каталоги пользователей, общие папки) необходим DOS-клиент, поддерживающий следующие протоколы: TCP/IP и NetBIOS, а также WinSocket (для корректной работы Windows 3.1*x* с нашей сетью). Раз мы решили применять Windows 3.1*x*, то вполне логично в качестве клиента использовать Microsoft Network Client for MS-DOS version 3.0.

Теперь подготовим клиентский компьютер — на жесткий диск запишем инсталляцию сетевого клиента и Windows. Здесь есть некоторые сложности — поставить клиента на жесткий диск, а потом переписать на дискету и уже с нее запускать клиента не получается, т. к. при установке клиента некоторые пути прописываются *прямо* в исполняемый файл.

Поэтому сделаем следующее — создадим на жестком диске каталог \tmp и выполним команду (можно прописать ее в файле Autoexec.bat):

```
subst a: c:/tmp/
```

В результате в системе появится псевдодисковод A:.

После этого начнем установку клиента на псевдодисковод A:. Если в ходе инсталляции клиент зависнет, следует повторить установку. При инсталляции необходимо правильно выставить параметры сетевой карты, выбрать необходимые протоколы, определить имя пользователя и рабочую группу. Будем считать, что клиент установлен успешно.

Теперь необходимо его настроить, чтобы можно было использовать ресурсы Samba как сетевые диски. Копируем установленного клиента в любой временный каталог и наводим порядок в следующих файлах:

- Hosts;
- Lmhosts;

- Networks;
- Protocol.ini;
- System.ini.

О файлах Hosts, Lmhosts и Networks мы говорить пока не будем. Остановимся на файлах Protocol.ini и System.ini.

В файле Protocol.ini есть следующая секция:

```
[TCPIP]
NBSessions=6
SubNetMask0=255 255 0 0
IPAddress0=0 0 0 0
DisableDHCP=0
DriverName=TCPIP$
BINDINGS=MS$NE2CLONE
LANABASE=1
```

Мы должны привести значения полей `SubNetMask0`, `IPAddress0`, `DisableDHCP` в надлежащее состояние — правильно задать сетевую маску и либо определить использование DHCP (`DisableDHCP=0`), либо задать жестко IP-адрес и запретить использование DHCP.

В файле System.ini для нас интересны следующие ключи:

```
[network]
...
computername=Test
lanroot=A:\NET
username=A
workgroup=MYGROUP
```

Вот и все. Перезагружаем машину, запускаем файл net и просматриваем доступные соединения. Если вы правильно сконфигурировали сервер и клиент, то должны увидеть в списке доступных ресурсов Linux-сервер с ресурсом <*имя_пользователя*>. Именно в каталоге /home/<*имя_пользователя*> мы и будем держать нужные нам программы (в частности, Windows 3.1*x*).

Далее, с сайта фирмы Microsoft надо загрузить файлы 51.txt и 62.txt (**ftp://ftp.microsoft.com/bussys/winnt/kb/Q142/0/62.txt** и **ftp://ftp.microsoft.com/bussys/winnt/kb/Q128/7/51.txt**). В этих файлах описывается, как установить и заставить работать Windows 3.1*x* и Microsoft Network Client version 3.0 for MS-DOS.

После установки Windows 3.1*x* не забудьте в ее свойствах установить тип своп-файла (swap) как отсутствующий (в дальнейшем это поможет избежать перегрузки сети посредством пересылки туда и обратно файлов подкачки).

Создание загрузочного образа дискеты

А теперь изготовим загрузочный образ. Пойдем от простого к сложному, попробуем загрузить по сети операционную систему MS-DOS. Впрочем, тут мы имеем два варианта — загрузочную дискету можно делать, а можно использовать "виртуальную" дискету. В отличие от Nowell или Lantastic, допускается копировать в отдельный каталог на винчестере необходимые файлы и работать с ними вместо того, чтобы постоянно переписывать дискету.

Итак, делаем загрузочную дискету — берем DOS 5.0 (или выше, но помните, чем новее версия DOS, тем больше системные файлы при близкой функциональности), запишем еще, например, Volkov Commander, создадим файлы Config.sys и Autoexec.bat. Загружаемся с дискеты, проверяем работоспособность на Volkov Commander. Все работает.

Теперь необходимо специальной программой создать образ загрузочной системы. Генератор образа дискеты называется mknbi-linux для загрузки Linux и mknbi-dos для загрузки DOS. Эта утилита входит в состав пакета Netboot, который распространяется как отдельно, так и в составе Etherboot (каталог /netboot) Описание утилиты можно посмотреть по команде `man mknbi-dos`. Сама утилита mknbi-dos находится в каталоге /usr/local/bin. Если она отсутствует, необходимо ее скомпилировать.

Вводим следующую команду:

```
mknbi-dos r /dev/fd0 o bootnet
```

где:

- `/dev/fd0` — источник файлов для загрузочного образа (в данном случае — дискета);
- `bootnet` — имя файла (загрузочного образа).

Таким образом, мы получили загрузочный образ с дискеты.

Для создания загрузочного образа из файлов, находящихся на винчестере, делаем следующее — создаем каталог (например /t) и переписываем туда нужные файлы. А потом создаем образ командой:

```
mknbi-dos r /t o bootnet
```

Вот мы и получили загрузочный образ с именем bootnet. Теперь копируем (или переносим) его в заранее созданный нами каталог /tftpboot.

Загрузка бездисковой машины

Теперь пришла пора испытать нашу систему. У подопытной машины отключаем в BIOS винчестер и дисководы (для чистоты эксперимента), устанавливаем ПЗУ в сетевую карту и включаем. Если все нормально сконфигурировано, получим машину с загруженной DOS и дисководом A:, на кото-

ром находятся все файлы с ранее созданной нами загрузочной дискеты (или каталога). А теперь самое время задаться вопросами — загрузили машину по сети, запущен наш привычный Volkov Commander, а дальше? Где дисководы, где ресурсы? Тут могут быть следующие варианты: во-первых, существует возможность создать загрузочный образ таким образом, что в DOS загрузочная дискета будет видеться как диск C:. Второй вариант — это, собственно то, ради чего мы все и проделали — раздача по сети ресурсов сервера и принтера, возможность загрузки Windows 3.1*x* (установка и настройка Windows 3.1*x* описана ранее).

Оптимизация бездисковой загрузки

Бездисковую загрузку следует оптимизировать. Установленный Microsoft Network Client стандартно занимает приблизительно 1,7 Мбайт, однако после хирургического вмешательства этот объем уменьшается до 800 Кбайт.

Вот список файлов, которые необходимо оставить:

- A.PWL;
- CONNECT.DAT;
- DHCP.PRM;
- DNR.EXE;
- EMSBFR.EXE;
- HOSTS;
- IFSHLP.SYS;
- LMHOSTS;
- NDISHLP.SYS;
- NE2000.DOS;
- NEMM.DOS;
- NET.EXE;
- NET.MSG;
- NETBIND.COM;
- NETWORKS;
- NMTSR.EXE;
- PROTMAN.DOS;
- PROTMAN.EXE;
- PROTOCOL;
- PROTOCOL.INI;
- SERVICES;
- SHARES.PWL;
- SOCKETS.EXE;
- SYSTEM.INI;
- TCPDRV.DOS;
- TCPTSR.EXE;
- TCPUTILS.INI;
- TINYRFC.EXE;
- UMB.COM;
- WFWSYS.CFG;
- WSAHDAPP.EXE;

Содержимое файлов Protocol.ini, System.ini, Tcputils.ini, Config.sys и Autoexec.bat описано в листингах 21.1, 21.2, 21.3, 21.4 и 21.5 соответственно.

Листинг 21.1. Файл Protocol.ini

```
 [network.setup]
version=0x3110
```

```
netcard=ms$ne2clone,1,MS$NE2CLONE,1
transport=tcpip,TCPIP
lana0=ms$ne2clone,1,tcpip
[TCPIP]
NBSessions=6
SubNetMask0=255 255 255 0
IPAddress0=192 168 40 33
DisableDHCP=1
DriverName=TCPIP$
BINDINGS=MS$NE2CLONE
LANABASE=1
[MS$NE2CLONE]
IOBASE=0x320
INTERRUPT=5
DriverName=MS2000$
[protman]
DriverName=PROTMAN$
PRIORITY=MS$NDISHLP
[MS$NDISHLP]
DRIVERNAME=ndishlp$
BINDINGS=MS$NE2CLONE
```

Листинг 21.2. Файл System.ini

```
 [network]
sizworkbuf=1498
filesharing=no
printsharing=no
autologon=yes
computername=A
lanroot=A:\NET
username=A
workgroup=MYGROUP
reconnect=yes
dospophotkey=N
lmlogon=0
logondomain=
preferredredir=full
autostart=full
maxconnections=8
[network drivers]
netcard=ne2000.dos
transport=ndishlp.sys,tcpdrv.dos,nemm.dos
```

```
devdir=A:\NET
LoadRMDrivers=yes
[386enh]
TimerCriticalSection=5000
UniqueDosPSP=TRUE
PSPIncrement=2
[Password Lists]
*Shares=A:\NET\Shares.PWL
A=A:\NET\A.PWL
B=A:\NET\B.PWL
```

Листинг 21.3. Файл Tcputils.ini

```
 [tcpglobal]
drivername=GLOBAL$
hostname=username
[sockets]
drivername=SOCKETS$
bindings=TCPIP
numsockets=4
numthreads=32
poolsize=3200
maxsendsize=1024
 [DNR]
drivername=DNR$
bindings=TCPIP
nameserver0=192 168 40 233
[telnet]
drivername=TELNET$
bindings=TCPIP
nsessions=0
max_out_sends=0
```

Листинг 21.4. Файл Config.sys

```
FILES=100
dos=high,umb
device=C:\WINDOWS\HIMEM.SYS
device=C:\WINDOWS\EMM386.EXE ram
LASTDRIVE=Z
device=IFSHLP.SYS
STACKS=9,256
```

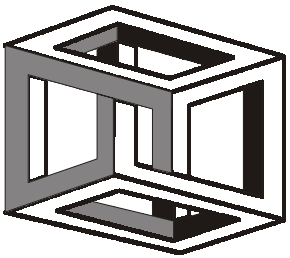
Листинг 21.5. Файл Autoexec.bat

```
set path=C:\WINDOWS;c:;c:\dos;c:\vc;c:\net
PATH=C:\IDAPI;%PATH%
SET TEMP=C:\WINDOWS\TEMP
Rem следующая строчка используется при отладке на винчестерной машине
subst a: c:\a
A:\NET\net initialize
A:\NET\netbind.com
A:\NET\umb.com
A:\NET\tcptsr.exe
A:\NET\tinyrfc.exe
A:\NET\nmtsr.exe
A:\NET\emsbfr.exe
A:\NET\dnr.exe
A:\NET\sockets
A:\NET\net start
```

Существует возможность удалить из ОЗУ загрузочный образ дискеты. Как это сделать, смотрите в документации Etherboot и в описании программы rmrd.com.

Литература и ссылки

- Страницы man для пакетов Etherboot, Netboot.
- Diskless HOWTO.
- **alst.odessa.ua** — Стахнов А. Удаленная загрузка. Сервер Linux. Клиентская часть DOS, Windows 3.1. Инструкция по установке и настройке.
- **etherboot.sourceforge.net/** — сайт в поддержку пакета Etherboot.
- **ftp.microsoft.com/bussys/clients/msclient/** — Microsoft Network Client version 3.0 for MS-DOS.
- **netstation.sf.net** — организация бездисковых станций.
- **ppg.ice.ru/ppg/xterminals** — Вагнер В. Использование бездисковых X-терминалов на базе Linux-PC.
- **pxes.sourceforge.net** — программное обеспечение для организации загрузки бездисковых клиентов с сетевыми картами, использующими PXE.
- **www.homepc.ru/offline/2002/78/22489/** — Вагнер В. Всем сестрам по терминалу.
- **www.linuxfocus.org/Russian/September1998/article63.html** — Яп К. Введение в сетевую загрузку и протокол Etherboot.
- **www.menet.umn.edu/~kaszeta/unix/xterminal/index.html** — Kaszeta R. A new use for old and outdated PCs.
- **www.nilo.org** — сетевая загрузка бездисковых компьютеров.



ЧАСТЬ V

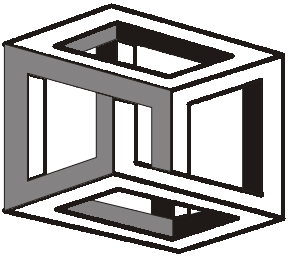
Утилиты администрирования и обеспечения безопасности

Эта часть посвящена целиком администрированию системы. Здесь рассмотрен удаленный безопасный доступ к хостам, а также утилиты для администрирования и мониторинга сети. К сожалению, данная часть может послужить вам только отправной точкой для углубления своих знаний, поскольку безопасность — тема неисчерпаемая, и в нашем изменчивом мире приходится ежедневно отслеживать новинки программного обеспечения и приемы для обеспечения безопасности сети.

Глава 22. Доступ к удаленным компьютерам

Глава 23. Обеспечение безопасности и администрирование сети

ГЛАВА 22



Доступ к удаленным компьютерам

Любая UNIX-подобная операционная система может предоставлять удаленный доступ, начиная от простейшего консольного режима и заканчивая работой системы X Window, от простого редактирования текста (сидя за сотни и тысячи километров от хоста) до полного администрирования удаленной системы. В мире UNIX в порядке вещей, когда администратор сервера месяцами не имеет физического контакта с сервером и, тем не менее, он ежедневно удаленно производит мониторинг, обновление программного обеспечения и администрирование сервера.

Для этих целей используются несколько программных пакетов и протоколов: Telnet, SSH, r-команды и некоторые другие. Будем рассматривать эти программы, так сказать, по старшинству.

Telnet

Под Telnet понимают трехкомпонентную систему, состоящую из:

- Telnet-клиента;
- Telnet-сервера;
- Telnet-протокола.

Протокол Telnet

Протокол Telnet описан в стандарте RFC854. Авторы стандарта говорят, что назначение Telnet — дать общее описание, насколько это только возможно, двунаправленного, восьмибитового взаимодействия, главной целью которого является обеспечение стандартного метода взаимодействия терминального устройства и терминал-ориентированного процесса. При этом этот протокол может быть использован и для организации взаимодействий "терминал-терминал" (связь) и "процесс-процесс" (распределенные вычисления).

Telnet является протоколом приложения и использует транспортный протокол TCP. Протокол Telnet для обеспечения функциональности основан на следующих базовых концепциях:

- сетевой виртуальный терминал (Network Virtual Terminal, NVT);
- согласование параметров взаимодействия;
- симметрия связи "терминал-процесс".

Рассмотрим эти концепции подробнее.

Сетевой виртуальный терминал позволяет абстрагироваться от реалий жизни. Это стандартное описание наиболее широко используемых возможностей реальных физических терминальных устройств. Сетевой виртуальный терминал позволяет описать и преобразовать в стандартную форму способы отображения и ввода информации. Telnet-клиент и Telnet-сервер преобразовывают характеристики физических устройств в спецификацию сетевого виртуального терминала, что позволяет унифицировать характеристики физических устройств и обеспечить принцип совместимости устройств с разными возможностями. Характеристики диалога диктуются устройством с меньшими возможностями.

В протоколе Telnet сетевой виртуальный терминал определен как "двунаправленное символьное устройство, состоящее из принтера и клавиатуры". Принтер предназначен для отображения приходящей по сети информации, а клавиатура — для ввода данных, передаваемых по сети. По умолчанию предполагается, что для обмена информацией используется 7-битовый код ASCII, каждый символ которого закодирован в 8-битовое поле.

Согласование параметров взаимодействия позволяет унифицировать возможности представления информации на терминальных устройствах. Благодаря этой концепции, можно использовать большинство возможностей современных терминалов. Обычно для этого существует специальная таблица соответствия, которая позволяет нестандартные команды терминала заменить стандартными наборами команд. Как правило, процесс согласования форм представления информации происходит в начальный момент организации Telnet-соединения. Каждый из процессов старается установить максимальные параметры сеанса. В UNIX-системах параметры терминалов содержатся в базе данных описания терминалов termcap. При инициировании Telnet-соединения обычно именно эти параметры используются в процессе согласования формы представления данных. При этом из одной системы в другую передается значение переменной окружения `TERM`. В процессе договора останутся только те функции, которые поддерживаются на обоих концах соединения.

Симметрия взаимодействия позволяет клиенту и серверу в течение одной сессии меняться ролями.

Команды Telnet

В табл. 22.1 приведены некоторые команды протокола Telnet с кратким пояснением. Как видно из таблицы, каждая команда представлена одним байтом, и чтобы различать команды и передаваемые данные, используется специальный признак команды.

Таблица 22.1. *Команды протокола Telnet*

Команда	Десятичное значение	Описание
EOF	236	Конец файла
SUSP	237	Подавить текущий процесс
ABORT	238	Прервать процесс
EOR	239	Конец записи
SE	240	Конец вспомогательной процедуры согласования
NOP	241	Нет операции
Data Mark	242	Часть потока данных для синхронизации
Break	243	Символ BRK виртуального терминала сети
Interrupt Process	244	Прервать текущий процесс
Abort Output	245	Отменить вывод без прекращения процесса
Are You There	246	Показывает, что связь не разорвана
Erase Character	247	Удалить предшествующий символ
Erase Line	248	Удалить текущую строку
Go Ahead	249	Запрос на ввод (для полудуплексных соединений)
SB	250	Начало вспомогательной процедуры согласования
WILL	251	Предложение начать выполнение факультативной команды (или подтверждение, что вы ее выполняете)
WON'T	252	Отказ выполнить (или продолжить выполнение)
DO	253	Требование к партнеру выполнить факультативную команду (или подтверждение ожидания выполнения другой стороной)

Таблица 22.1 *(окончание)*

Команда	Десятичное значение	Описание
DON'T	254	Требование к партнеру прекратить выполнение факультативной команды (или подтверждение того, что больше не ожидается выполнение операции другой стороной)
IAC	255	Интерпретировать как команду

Протокол Telnet предусматривает единый TCP-канал и для данных пользователя, и для управления. Поскольку по протоколу Telnet команды управления чередуются с данными, командам должен предшествовать специальный символ, называемый IAC (Interpret as Command, интерпретировать как команду) с кодом 255. В том случае, если необходимо передать символ данных с десятичным кодом 255, следует его передать дважды.

Команды протокола Telnet имеют размер не менее двух байтов. Первый из них всегда символ перехода в командный режим — IAC. Второй — команда протокола Telnet.

Программа-клиент telnet

Программа telnet — стандартный Telnet-клиент, входящий во все операционные системы UNIX-семейства и в практически все операционные системы Windows.

Для подключения к удаленной системе обычно используется команда вида:

telnet *<имя_хоста>*

Основные команды программы telnet приведены в табл. 22.2.

Таблица 22.2. *Команды программы telnet*

Команда	Описание
Open *<host>* [*<port>*]	Начать Telnet-сессию с машиной *<host>* по порту *<port>*. Адрес машины можно задавать как в форме IP-адреса, так и в форме доменного адреса
close	Завершить Telnet-сессию и вернуться в командный режим
Quit	Завершить работу программы telnet

Таблица 22.2 *(окончание)*

Команда	Описание
`Z`	"Заморозить" Telnet-сессию и перейти в режим интерпретатора команд локальной системы. Из этого режима можно выйти по команде `Exit`
`Mode <type>`	Если значение `<type>` равно `line`, то используется буферизованный обмен данными, если `character` — обмен не буферизованный
`? [<command>] help [<command>]`	Список команд или описание конкретной команды
`Send <argument>`	Данная команда используется для ввода команд и сигналов протокола Telnet, которые указываются в качестве аргумента

Программа-сервер telnetd

Программа telnetd — это сервер, который обслуживает протокол Telnet. Программа telnetd обслуживает TCP-порт 23, но ее можно сконфигурировать на использование другого порта.

При установке взаимодействия с удаленным клиентом telnetd обменивается командами настройки: включение режима эха, обмен двоичной информацией, тип терминала, скорость обмена, переменные окружения.

Применение Telnet и безопасность

Протокол Telnet долгие годы был единственной универсальной возможностью удаленно работать с различными консольными программами. По своей простоте, нетребовательности к ресурсам и полосе пропускания он до сих пор не имеет себе равных. Помимо этого, клиентская программа telnet позволяет устанавливать соединения и с другими сервисами (например, с почтовым сервером SMTP или POP3), что дает возможность производить различные манипуляции (например, просмотреть без почтового клиента содержимое своего почтового ящика или отправить письмо). Однако при всех его достоинствах протокол Telnet имеет один огромный недостаток — весь трафик пересылается в открытом виде. Из-за этого любому злоумышленнику не составляет труда перехватить логин и пароль пользователя, а также другую информацию. В качестве альтернативы протоколу Telnet используются программные пакеты SSH и OpenSSH. Но о них будет рассказано далее.

Семейство r-команд

Приблизительно в то же время, что и протокол Telnet, были созданы программы, предназначенные для удаленного администрирования и работы, так называемые r-команды (remote-команды).

Команда *rlogin*

Команда `rlogin` (remote login) предназначена для захода удаленным терминалом на UNIX-хост. Стандарт RFC1282 содержит спецификацию протокола Rlogin. Программа rlogin использует одно TCP-соединение между клиентом и сервером. Для нормального функционирования необходимо создать файл .rhosts, содержащий список хостов и пользователей, которым разрешено удаленно регистрироваться в системе. Каждая строка представляет собой пару "хост-пользователь", разделенную пробелом.

Команда *rsh*

Команда `rsh` (remote shell) используется для запуска командной оболочки на удаленном компьютере, после чего на нем возможно выполнять различные программы.

Команда *rcp*

Команда `rcp` (remote copy) используется для копирования файлов между компьютерами, причем эта операция может производиться между двумя удаленными компьютерами. Данная команда может копировать как один файл, так и группу файлов или каталогов.

Команда *rsync*

Команда `rsync` аналогично команде `rcp` позволяет копировать файлы между удаленными компьютерами. Однако, в отличие от `rcp`, может значительно ускорить процесс копирования часто изменяемых пользователем файлов, поскольку благодаря используемым алгоритмам передает только измененные части файлов. Также позволяет копировать ссылки (links), специальные устройства (device), владельца и группу файла, права доступа.

Команда *rdist*

Эта команда позволяет осуществить массовую автоматическую рассылку файлов с локального хоста на большую группу хостов с проверкой наличия места, рассылкой извещений о проблемах, исполнением завершающих про-

цедур и т. п. Сохраняет имя владельца, имя группы, права доступа и время модификации файла.

Применение r-команд и безопасность

Как и в ситуации с Telnet, r-команды имеют ту же проблему с безопасностью — абсолютно незащищенную передачу информации, поэтому использование r-команд категорически не рекомендуется.

SSH и OpenSSH

Протокол SSH обеспечивает возможность удаленного выполнения команд и копирования файлов с аутентификацией клиента и сервера, а также с шифрованием передаваемых данных, в том числе имени и пароля пользователя. Дополнительно обеспечивается шифрование данных X Window и перенаправление любых TCP-соединений. Существует несколько программных реализаций, в частности коммерческий SSH и бесплатный пакет с открытым исходным кодом OpenSSH.

Принцип работы SSH

SSH представляет собой протокол транспортного уровня, аутентификации и соединения, а также программные средства безопасного доступа к компьютерам по небезопасным каналам связи (Telnet, X11, RSH, FTP). Аутентификация производится с использованием асимметричного шифрования с открытым ключом (SSH1 — RSA, SSH2 — RSA/DSA). Обмен данными — симметричное шифрование. Целостность переданных данных проверяется с помощью специальных контрольных сумм. Протокол транспортного уровня работает поверх TCP и использует порт 22. В качестве ключа берется случайная строка, которую генерирует клиент, шифрует с помощью открытого ключа сервера и передает серверу. Протокол аутентификации работает поверх протокола транспортного уровня и обеспечивает аутентификацию клиента для сервера. Шифрование трафика начинается после аутентификации сервера, но до аутентификации клиента, таким образом, пароли в открытом виде не передаются. Возможно соединение произвольных портов TCP по защищенным каналам. Предусматривается возможность сжатия.

Существует две версии протокола: SSH1 и SSH2. По своей реализации это совершенно разные протоколы. Протокол SSH2 был разработан с учетом найденных в первом варианте уязвимостей. Однако не стоит уповать на абсолютную надежность и защищенность SSH2. Недавно в OpenSSH была найдена уязвимость, которую правда быстро исправили. Поэтому желательно отслеживать сообщения о найденных уязвимостях в пакетах, используемых на сервере и обновлять их по мере выхода новых версий.

OpenSSH

Это некоммерческая реализация протокола SSH с открытым кодом. Программный пакет способен работать с протоколом SSH1 и SSH2. Имеется также поддержка r-команд.

Для дистрибутива Red Hat установка пакета OpenSSH включена в стандартную инсталляцию.

Конфигурирование OpenSSH

Конфигурирование OpenSSH очень сильно зависит от вашей концепции обеспечения безопасности и необходимости поддержки старого типа протокола. Поскольку использование протокола SSH1 из-за найденных уязвимостей не рекомендуется — при конфигурировании необходимо запретить его использование. Также рекомендуется запретить использование r-команд и всего, что с ними связано. При конфигурировании OpenSSH необходимо произвести настройку сервера и клиента. Конфигурационный файл сервера называется sshd_config, а клиента — ssh_config.

Файл sshd_config

Файл sshd_config задает параметры SSH-серверу и может содержать внушительный список различных параметров. Далее приведены его основные конфигурационные параметры:

- `AllowGroups` *<список-имен-групп-через-пробел>* — вход разрешен только пользователям, чья группа входит в этот список;
- `AllowTcpForwarding yes/no` — разрешает или запрещает TCP Forwarding;
- `AllowUsers` *<список-имен-через-пробел>* — вход разрешен только перечисленным пользователям;
- `AuthorizedKeysFile` *<имя-файла-с-публичным-ключом>* — задает имя файла, содержащее публичный ключ;
- `Banner` *<сообщение-перед-аутентификацией>* — текст сообщения, выводимого сервером перед аутентификацией клиента;
- `Ciphers` — список алгоритмов симметричного шифрования для SSH2: `aes128-cbc`, `3des-cbc`, `blowfish-cbc`, `cast128-cbc`, `arcfour`;
- `ClientAliveInterval` *<секунд>* — определяет интервал в секундах, через который сервер будет производить проверку, произошло или нет отключение клиента;
- `ClientAliveCountMax` *<число>* — данный параметр определяет число неудачных проверок существования связи с пользователем до разрыва сессии;

- `DenyGroups` *<список-имен-групп-через-пробел>* — определяет список групп пользователей, которым запрещено устанавливать соединение с сервером;
- `DenyUsers` *<список-имен-через-пробел>* — определяет список пользователей, которым запрещено устанавливать соединение с сервером;
- `GatewayPorts no/yes` — данный параметр определяет, разрешать или нет удаленным хостам доступ к перенаправленным портам;
- `HostbasedAuthentication no/yes` — разрешить или запретить аутентификацию, используя имя хоста (только для SSH2);
- `HostKey` *<имя-файла-содержащего-приватный-ключ>* — с помощью данного параметра можно указать серверу, где расположен файл, содержащий секретный ключ шифрования;
- `IgnoreRhosts yes/no` — этот параметр позволяет определить, использовать или нет файлы .rhosts и .shosts для аутентификации. Для увеличения безопасности системы рекомендуется запретить использование этих файлов;
- `IgnoreUserKnownHosts no/yes` — параметр позволяет запретить использование файла ~/.ssh/known_hosts во время аутентификации rhosts+RSA;
- `KeepAlive yes/no` — параметр позволяет использовать механизм регулярных сообщений для проверки разрыва связи;
- `KerberosAuthentication yes/no` — параметр позволяет запретить использование Kerberos при аутентификации;
- `KerberosOrLocalPasswd yes/no` — в том случае, если аутентификация через Kerberos не прошла, данный параметр позволяет использовать /etc/passwd для аутентификации;
- `KeyRegenerationInterval 3600` — параметр задает интервал регенерации ключа сервера;
- `ListenAddress 0.0.0.0` — параметр определяет, к каким адресам прислушиваться; при использовании необходимо также определить параметр `Port`;
- `LoginGraceTime` *<секунд>* — данный параметр определяет, через сколько секунд произойдет разрыв соединения, если при аутентификации пользователь за это время не введет пароль;
- `LogLevel INFO` — параметр определяет, какой уровень использовать при создании сообщений в журнал системы. Можно применять следующие уровни — `QUIET`, `FATAL`, `ERROR`, `INFO`, `VERBOSE`, `DEBUG`;
- `MACs` *<алгоритмы-проверки-целостности-данных>* — этот параметр определяет, какой алгоритм будет использоваться для проверки целостности данных: `hmac-md5`, `hmac-sha1`, `hmac-ripemd160`, `hmac-sha1-96`, `hmac-md5-96`;

- `MaxStartups 10` — данный параметр задает максимально возможное количество соединений, ожидающих аутентификации;
- `PasswordAuthentication yes/no` — параметр разрешает аутентификацию по паролю;
- `PermitEmptyPasswords no/yes` — параметр разрешает использование пустых паролей;
- `PermitRootLogin yes/no/without-password/forced-commands-only` — параметр разрешает пользователю root подключаться к серверу;
- `PidFile` *<имя-файла>* — параметр задает имя файла, в котором будет храниться PID процесса сервера;
- `Port 22` — параметр определяет, какой порт слушает сервер;
- `PrintMotd yes/no` — параметр разрешает использование /etc/motd при входе пользователя в систему для выдачи сообщения;
- `Protocol 2` — параметр определяет, с какой версией протокола работает сервер;
- `PubkeyAuthentication yes/no` — параметр разрешает использовать публичный ключ при аутентификации;
- `ReverseMappingCheck no/yes` — параметр разрешает после определения адреса по имени хоста производить проверку того, что обратная зона для этого адреса указывает на тот же самый хост;
- `RhostsAuthentication no/yes` — параметр разрешает аутентификацию только на основании файлов .rhosts или /etc/hosts.equiv;
- `RhostsRSAAuthentication no/yes` — параметр разрешает аутентификацию на основе .rhosts- и RSA-аутентификации;
- `RSAAuthentication yes/no` — данный параметр используется только для протокола SSH1;
- `ServerKeyBits 768` — данный параметр определяет длину ключа;
- `StrictModes yes/no` — параметр разрешает проверять права доступа к файлам с частными паролями при запуске;
- `SyslogFacility AUTH` — параметр задает тип сообщений, передаваемых на syslog: `DAEMON`, `USER`, `AUTH`, `LOCAL0`, `LOCAL1`, `LOCAL2`, `LOCAL3`, `LOCAL4`, `LOCAL5`, `LOCAL6`, `LOCAL7`;
- `UseLogin no/yes` — параметр разрешает использовать login для интерактивных сессий;
- `X11DisplayOffset 10` — параметр определяет первый доступный номер дисплея при передаче X11.

Файлы на сервере, используемые при входе SSH

При входе SSH на сервере используются следующие файлы:

- /etc/nologin — при наличии этого файла запрещается вход пользователей, кроме root. Содержимое файла выдается в качестве сообщения о причине;
- /etc/hosts.allow — при компиляции с libwrap используется для разрешения доступа;
- /etc/hosts.deny — при компиляции с libwrap используется для запрещения доступа;
- ~/.rhosts — файл содержит пары "хост-пользователь", разделенные пробелом. Для указанного пользователя с указанного хоста разрешается заходить без указания пароля при использовании `RhostsAuthentication` и `RhostsRSAAuthentication`. Так же используется семейством r-команд;
- ~/.shosts — аналогично файлу .rhosts, но не используется семейством r-команд;
- /etc/hosts.equiv — список хостов, с которых пользователи могут заходить, не указывая паролей, под теми же самыми именами. За именем хоста можно указывать имя конкретного пользователя. Также используется семейством r-команд;
- /etc/shosts.equiv — аналогично файлу hosts.equiv, но не используется семейством r-команд;
- ~/.ssh/environment — содержит пары вида "имя-значение", которые помещаются в окружение при входе.

Файлы ключей сервера

В качестве ключей сервера используются следующие файлы:

- /usr/local/etc/ssh_host_key — приватный ключ хоста;
- /usr/local/etc/ssh_host_rsa_key — приватный ключ хоста, алгоритм шифрования RSA;
- /usr/local/etc/ssh_host_dsa_key — приватный ключ хоста, алгоритм шифрования DSA;
- /usr/local/etc/ssh_host_key.pub — публичный ключ хоста;
- /usr/local/etc/ssh_host_rsa_key.pub — публичный ключ хоста, алгоритм шифрования RSA;
- /usr/local/etc/ssh_host_dsa_key.pub — публичный ключ хоста, алгоритм шифрования DSA.

Файл ssh_config

Данный файл предназначен для конфигурации SSH-клиента и разделен на секции директивами `Host`.

Секция применяется при работе с хостом, удовлетворяющим шаблону секции:

- `Host <шаблоны>` — следующие опции применимы к хостам, подходящим под один из шаблонов; имя хоста берется из командной строки, в шаблонах используются символы `*` и `?`;
- `BatchMode no|yes` — параметр разрешает не запрашивать пароль/парольную фразу;
- `CheckHostIP yes|no` — позволяет дополнительно проверять адрес сервера в known_hosts;
- `Cipher 3des|blowfish` — определяет алгоритм шифрования данных;
- `Ciphers aes128-cbc, 3des-cbc, blowfish-cbc, cast128-cbc, arcfour, aes192-cbc, aes256-cbc` — определяют алгоритм шифрования данных;
- `ClearAllForwardings no|yes` — данный параметр позволяет сбросить все перенаправления портов;
- `Compression no|yes` — параметр разрешает производить сжатие передаваемых данных;
- `CompressionLevel <уровень-сжатия>` — параметр определяет уровень компрессии для протокола SSH1;
- `ConnectionAttempts <число-попыток-соединения>` — параметр задает число попыток установления соединения;
- `EscapeChar <символ>|<^символ>|none` — параметр позволяет определить символ для использования вместо тильды;
- `FallBackToRsh no|yes` — параметр разрешает использовать RSH в том случае, если сервер не имеет SSH-сервера;
- `ForwardAgent no|yes` — параметр определяет, передавать ли запрос к агенту аутентификации на удаленный хост;
- `GatewayPorts no|yes` — параметр разрешает удаленным хостам соединяться на перенаправленные локальные порты;
- `GlobalKnownHostsFile <имя-файла>` — параметр разрешает использовать указанный файл вместо /usr/local/etc/ssh_known_hosts;
- `HostKeyAlgorithms ssh-rsa,ssh-dss` — параметр определяет используемые алгоритмы шифрования (SSH2);
- `IdentityFile <имя-файла>` — параметр определяет файл, хранящий RSA- или DSA-приватный ключ;
- `KeepAlive yes|no` — параметр позволяет заметить разрыв связи или аварийное завершение на удаленном конце;
- `KerberosAuthentication yes|no` — параметр разрешает использовать Kerberos-аутентификацию;

- `LogLevel INFO` — параметр определяет, какой уровень использовать при создании сообщений в журнал системы. Можно использовать следующие уровни: `QUIET`, `FATAL`, `ERROR`, `INFO`, `VERBOSE`, `DEBUG`;
- `MACs hmac-md5, hmac-sha1, hmac-ripemd160, hmac-sha1-96, hmac-md5-96` — параметр определяет используемые алгоритмы для создания контрольной суммы;
- `NumberOfPasswordPrompts 3` — параметр определяет количество попыток ввода пароля пользователя;
- `PasswordAuthentication yes/no` — параметр разрешает аутентификацию по паролю;
- `Port 22` — параметр определяет, к какому порту будет подключаться клиент;
- `PreferredAuthentications publickey, password, keyboard-interactive` — параметр определяет приоритеты аутентификации (SSH2);
- `Protocol` *`<список-версий-протокола>`* — параметр задает список версий протокола в порядке предпочтительности;
- `ProxyCommand` — применение этого параметра позволяет использовать дополнительную команду для соединения с сервером;
- `PubkeyAuthentication yes|no` — параметр разрешает использовать при аутентификации публичный ключ (SSH2);
- `RhostsAuthentication yes|no` — параметр разрешает использовать при аутентификации файл .rhosts (SSH1);
- `StrictHostKeyChecking ask|no|yes` — параметр разрешает не добавлять незнакомые или изменившиеся хосты в know_hosts;
- `UsePrivilegedPort yes|no` — параметр разрешает использовать привилегированные порты для установления соединения;
- `User` *`<имя-пользователя>`* — параметр задает имя пользователя;
- `UserKnownHostsFile` *`<файл-known_hosts>`* — параметр определяет местоположение файла known_hosts;
- `UseRsh no|yes` — параметр разрешает использовать RSH в том случае, если SSH на хосте отсутствует.

Файлы ключей клиента

В качестве ключей клиента используются следующие файлы:

- ~/.ssh/identity — приватный RSA1-ключ пользователя;
- ~/.ssh/id_dsa — приватный DSA2-ключ пользователя;
- ~/.ssh/id_rsa — приватный RSA2-ключ пользователя;
- ~/.ssh/identity.pub — публичный RSA1-ключ пользователя;

- ~/.ssh/id_dsa.pub — публичный DSA2-ключ пользователя;
- ~/.ssh/id_rsa.pub — публичный RSA2-ключ пользователя.

Ключи запуска сервера SSH

Помимо конфигурационного файла, некоторые особенности функционирования сервера SSH можно задать, используя ключи запуска. Далее приведены основные ключи запуска:

- `-D` — не отсоединяться от терминала при запуске;
- `-b` *<бит>* — ключ задает число битов ключа сервера (SSH1), по умолчанию 768;
- `-d` — переводит сервер в отладочный режим, использование нескольких ключей увеличивает количество отладочной информации;
- `-e` — ключ разрешает выводить сообщения на stderr вместо syslog (т. е. не журналировать сообщения, а сразу выводить на стандартное устройство для вывода ошибок — обычно это текстовый терминал);
- `-f` *<имя-конфигурационного-файла>* — ключ определяет положение конфигурационного файла, удобно использовать при отладке;
- `-g` *<время-ожидания>* — ключ определяет время ожидания между вводом логина и пароля пользователя;
- `-h` *<файл-ключей-хоста>* — ключ определяет местоположение файла ключей;
- `-k` *<интервал>* — параметр задает интервал регенерации ключа сервера;
- `-p` *<порт>* — параметр определяет порт, который будет слушать сервер;
- `-q` — ключ запрещает выдачу информации на syslog (т. е. запрещает журналирование событий);
- `-t` — с помощью этого параметра можно произвести проверку на отсутствие ошибок в конфигурационном файле и ключах;
- `-u` *<число>* — вместо имен хостов, превышающих *<число>*, в журнале utmp будет записываться IP-адрес: `-u0` вызывает безусловную запись IP-адресов;
- `-4` — использование протокола IPv4;
- `-6` — использование протокола IPv6.

Ключи запуска клиента SSH

Как и для сервера, для изменения некоторых параметров клиента можно воспользоваться ключами запуска:

- `-a` — запретить перенаправление агента аутентификации;
- `-A` — разрешить перенаправление агента аутентификации;

- `-b <адрес>` — позволяет для хоста с несколькими интерфейсами использовать конкретный адрес;
- `-c blowfish|3des` — задает используемый алгоритм шифрования (SSH1);
- `-c <список-алгоритмов-шифрования-через-запятую>` — алгоритм в начале списка имеет наибольший приоритет; по умолчанию: `aes128-cbc`, `3des-cbc`, `blowfish-cbc`, `cast128-cbc`, `arcfour`, `aes192-cbc`, `aes256-cbc` (SSH2);
- `-D <локальный-порт>` — эмуляция SOCKS4-сервера по защищенному каналу;
- `-e <символ> | <^символ> | none` — определяет Escape-символ вместо тильды; `none` обеспечивает прозрачную передачу данных;
- `-f` — перейти в фоновый режим после запроса пароля или парольной фразы;
- `-F <имя-конфигурационного-файла>` — разрешает использовать указанный файл в качестве конфигурационного;
- `-g` — разрешать удаленному хосту подсоединяться к локальным перенаправленным портам;
- `-i <имя-файла>` — определяет файл, хранящий RSA/DSA-приватный ключ;
- `-k` — запретить перенаправление Kerberos;
- `-l <имя-пользователя>` — от имени какого пользователя устанавливается соединение;
- `-m <список-алгоритмов-обеспечения-целостности-соединения>` — определяет алгоритмы подсчета контрольной суммы;
- `-n` — направить /dev/null на stdin и перейти в фоновый режим;
- `-p <порт>` — соединиться с указанным хостом на удаленном хосте;
- `-P` — использовать непривилегированный порт для исходящего соединения, чтобы обойти ограничения сетевого экрана;
- `-R <локальный-порт>:<хост>:<удаленный-порт>` — если происходит соединение на удаленный порт, то оно перенаправляется по защищенному каналу на локальный порт;
- `-s` — запуск подсистемы на сервере — например, sftp; имя подсистемы задается последним параметром;
- `-t` — требовать выделения псевдо-tty;
- `-T` — не выделять псевдо-tty;
- `-x` — запретить перенаправление X11;
- `-X` — разрешить перенаправление X11;

- `-1` — использовать только SSH1-протокол;
- `-2` — использовать только SSH2-протокол;
- `-4` — использовать IPv4;
- `-6` — использовать IPv6.

Программы, входящие в пакет OpenSSH

Помимо клиента и сервера, в пакет OpenSSH входят программы, предназначенные для генерации ключей, аутентификации, а также программы, призванные заменить набор r-команд.

Программа ssh-keygen

Программа ssh-keygen предназначена для генерации, преобразования и управления ключами. По умолчанию генерирует RSA-ключ. При генерации запрашивается парольная фраза. Забытую парольную фразу восстановить невозможно. Число битов по умолчанию — 1024. Имя файла для хранения публичного ключа образуется из имени файла для частного ключа добавлением суффикса .pub. Ключ хоста должен иметь пустую парольную фразу.

Возможные строки запуска:

- генерирует ключ по указанному пользователем алгоритму:

```
ssh-keygen [-t rsa1|dsa|rsa] [-b <бит>] [-N <парольная-фраза>]
           [-C <комментарий>] [-f <имя-файла-записи>] [-q]
```

- изменяет комментарий:

```
ssh-keygen -c [-P <парольная-фраза>] [-C <комментарий>]
              [-f <файл-с-ключами>]
```

- читает приватный или публичный ключ в формате OpenSSH и преобразует его в формат SECSH для экспорта в другие реализации SSH:

```
ssh-keygen -e [-f <файл-с-ключами>]
```

- читает приватный или публичный ключ в формате SSH2 или SECSH и преобразует его в формат OpenSSH:

```
ssh-keygen -i [-f <файл-с-ключами>]
```

- позволяет изменить парольную фразу:

```
ssh-keygen -p [-P <старая-парольная-фраза>] [-N <новая-парольная-фраза>]
              [-f <файл-с-ключами>]
```

- читает приватный OpenSSH DSA ключ и выдает OpenSSH DSA публичный ключ:

```
ssh-keygen -y [-f <файл-с-ключами>]
```

Программа ssh-agent

Программа ssh-agent — позволяет производить RSA/DSA-аутентификацию. Она запускается в начале сессии и устанавливает переменные окружения, с помощью которых остальные программы могут использовать ее для автоматической аутентификации SSH. Параметром является имя команды и ее аргументы, выполнение ssh-agent останавливается при завершении команды. Если имя команды не указано, то ssh-agent запускается в фоновом режиме, а на stdout выдаются команды экспортирования необходимых переменных окружения.

Опции командной строки ssh-agent:

- `-c` — позволяет выдавать на stdout команды в стиле csh;
- `-s` — позволяет выдавать на stdout команды в стиле sh;
- `-k` — завершает работу агента — по переменной `SSH_AGENT_PID`.

Программа ssh-add

Эта программа используется для добавления приватных ключей. Она запрашивает парольную фразу, расшифровывает приватный ключ и посылает его ssh-agent. Если терминал недоступен, но определена переменная `DISPLAY`, то для ввода парольной фразы используется программа, определенная переменной `SSH_ASKPASS`. Таким образом, парольная фраза запрашивается только один раз за сеанс, а не при каждом вызове ssh/scp/sftp.

Опции командной строки ssh-add:

- *<имя файла>* — имя файла с приватным ключом, по умолчанию используется ~/.ssh/identity;
- `-L` — выдает публичные ключи, хранящиеся в ssh-add;
- `-d` — удаляет приватный ключ;
- `-D` — удаляет все ключи.

Программа sftp

Программа sftp (secure FTP) является клиентом для SFTP-сервера, который должен быть описан в опции `Subsystem` в конфигурационном файле sshd.

Программа sftp позволяет пересылать файлы в режиме, подобном FTP-протоколу, однако она осуществляет все операции поверх защищенного транспорта SSH. К сожалению, данный вариант FTP пока не получил широкого распространения.

Опции командной строки:

- [*<пользователь>*@]*<имя-хоста>*[:*<каталог>*/] — задает аналогично FTP имя пользователя, хост, к которому производится подключение, и каталог подключения;

- `-b <имя-файла>` — позволяет читать команды из файла вместо стандартного устройства ввода;
- `-C` — разрешает использовать сжатие пересылаемых файлов;
- `-F <имя-конфигурационного-файла-ssh>` — указывает, какой конфигурационный файл использовать;
- `-o <опция>` — опция передается SSH.

Интерактивные команды, используемые sftp, аналогичны FTP-командам:

- `bye` — разорвать соединение;
- `cd <путь>` — сменить каталог;
- `lcd <путь>` — сменить каталог;
- `chgrp gid <имя-файла>` — изменить групповой идентификатор файла на указанный в команде;
- `chmod mode <имя-файла>` — изменить атрибуты файла;
- `chown uid <имя-файла>` — изменить владельца файла;
- `exit` — выйти;
- `get [-P] <имя-удаленного-файла> [<имя-локального-файла>]` — команда для получения файла, ключ `-P` позволяет сохранить права и время создания и модификации получаемого файла;
- `help` — позволяет получить справку по командам;
- `lls [<опции-ls> [<имя-файла>]]` — получить список файлов;
- `lpwd` — пароль;
- `mkdir <имя>` — создать каталог;
- `put [-P] <имя-локального-файла> [<имя-удаленного-файла>]` — выгрузить на сервер файл, ключ `-P` позволяет сохранить права, время создания и модификации передаваемого файла;
- `pwd` — пароль;
- `quit` — выйти;
- `rename <старое-имя> <новое-имя>` — переименовать файл;
- `rmdir <имя>` — удалить каталог;
- `rm <имя-файла>` — удалить файл;
- `symlink <старое-имя> <новое-имя>` — создать символическую ссылку.

Программа scp

Программа scp является аналогом программы rcp и осуществляет копирование файлов между хостами, причем оба могут быть удаленными. Способы

аутентификации аналогичны SSH. Вызывает SSH для организации канала передачи данных. Имя файла записывается в виде:

`[[<пользователь>@]<хост>:]<файл>`

Опции командной строки:

- `-c <алгоритм-шифрования>` — опция передается SSH;
- `-i <имя-файла>` — файл с приватным ключом, передается в SSH;
- `-o <опция>` — опция передается SSH;
- `-p` — сохраняет время модификации, использования и права доступа к файлу;
- `-r` — позволяет рекурсивно копировать весь каталог;
- `-B` — пакетный режим — не запрашивать пароль или парольную фразу;
- `-C` — разрешает производить сжатие при передаче файла;
- `-F <конфигурационный-файл>` — определяет альтернативный конфигурационный файл;
- `-P <порт>` — задает порт сервера;
- `-S <программа>` — разрешает использовать указанную программу вместо SSH;
- `-4` — использовать IPv4;
- `-6` — использовать IPv6.

Программа ssh-keyscan

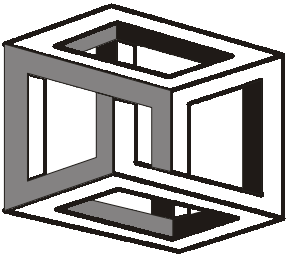
Программа ssh-keyscan позволяет собрать публичные ключи хостов, имена хостов задаются в качестве параметров или в файле. Опрос производится параллельно. Опции командной строки:

- `-t <тип-ключа>` — задает тип шифрования ключа (RSA1, RSA, DSA);
- `-T <секунд>` — определяет тайм-аут;
- `-f <имя-файла>` — определяет файл, в котором каждая строка содержит имя или адрес хоста;
- `-4` — использовать IPv4;
- `-6` — использовать IPv6;
- `-p <удаленный-порт>` — определяет порт.

Литература и ссылки

- RFC854 — описание протокола Telnet.
- **lib.ru/LABIRINT/telnet.htm** — доступ к ресурсам Интернета в режиме удаленного терминала.
- **www.bog.pp.ru/work/ssh.html** — Bog BOS: SSH и OpenSSH: принципы работы, установка и настройка.
- **www.mnet.uz/citforum/internet/services/index.shtml** — Храмцов П. Б. Администрирование сети и сервисов Internet. Учебное пособие.
- **www.openssh.com** — сайт некоммерческой реализации SSH.
- **www.ssh.com** — сайт коммерческой реализации SSH.
- **www.tigerlair.com/ssh/faq/** — SSH FAQ.

ГЛАВА 23



Обеспечение безопасности и администрирование сети

Пожалуй, одна из самых сложных и трудоемких задач системного администратора — администрирование сети. Эта задача настолько комплексная, что можно практически все, о чем писалось ранее, отнести к подготовке администрирования сети. Слишком много параметров, программ, настроек могут прямо или косвенно отражаться на функционировании сети и сетевых сервисов. В этой главе все, так или иначе, будет касаться администрирования и управления сетью, хотя некоторые вещи с первого взгляда никоим образом не относятся к сети или ее настройке.

В той части главы, где будет говориться об инструментах, предназначенных для обнаружения уязвимости системы, мы опишем несколько программных пакетов, которые с одинаковым успехом можно применить как для взлома системы, так и для ее защиты.

Расширенное управление доступом к файлам

К сожалению, стандартные средства организации прав доступа к файлам в UNIX-подобных операционных системах зачастую не удовлетворяют требованиям некоторых системных администраторов. Проблема заключается в том, что определение прав доступа к файлам сводится к установке девяти битов, с помощью которых можно задать права доступа для владельца файла, группы, к которой принадлежит владелец файла, а также для всех остальных. Часто необходимо настроить доступ к файлу достаточно сложным образом — допустим, три человека из трех разных групп имеют право делать с файлом все что угодно, десять человек из других групп могут открывать файл на чтение, а еще десять — только выполнять. Для всех других пользователей доступ к этому файлу необходимо запретить. Устроить нечто подобное стандартными средствами Linux весьма нетривиальная задача. В такой

ситуации для решения данной проблемы можно воспользоваться Linux ACL (Access Control Lists, списки контроля доступа) — версией POSIX ACL для Linux. Linux ACL — это набор патчей (patch) для ядра операционной системы и программ для работы с файловой системой и несколько утилит, дающих возможность устанавливать права доступа к файлам не только для пользователя-владельца и группы-владельца файла, но и для любого пользователя или группы. В версии ядра 2.6 ACL наконец включили в стандартную поставку.

Linux ACL использует расширенные атрибуты (extended attributes) для хранения данных о правах доступа к файлам пользователей и групп. Расширенные атрибуты — это пара "имя/значение", привязанная к определенному файлу.

Список расширенного контроля доступа существует для каждого inode и состоит из шести компонентов. Первые три являются копией стандартных прав доступа к файлу. Они содержатся в единственном экземпляре в ACL и есть у каждого файла в системе:

- ACL_USER_OBJ — режим доступа к файлу пользователя-владельца;
- ACL_GROUP_OBJ — режим доступа к файлу группы-владельца;
- ACL_OTHER — режим доступа к файлу остальных пользователей.

Следующие два компонента устанавливаются для каждого файла в отдельности и могут присутствовать в ACL в нескольких экземплярах:

- ACL_USER — содержит UID и режим доступа к файлу пользователя, которому установлены права, отличные от основных. На каждого пользователя со своими правами на данный файл хранится отдельная запись. Не может существовать более одной записи на одного и того же пользователя;
- ACL_GROUP — то же самое, что и ACL_USER, но для группы пользователей;
- ACL_MASK — маска действующих прав доступа для расширенного режима.

При установке дополнительных прав доступа присваивается значение и элементу ACL_MASK.

Каталоги также могут иметь список контроля доступа по умолчанию. В отличие от основного ACL, он действует на создаваемые внутри данного каталога файлы и каталоги. При создании файла внутри такого каталога файл получает ACL, равный ACL по умолчанию этого каталога.

Установка Linux ACL

Для использования Linux ACL необходимо получить на сайте разработчиков собственно пакет Linux ACL, а также патчи для ядра операционной системы

Linux и некоторых утилит. Само собой, после наложения патчей придется перекомпилировать ядро операционной системы и утилиты (для ядер серии 2.6 перекомпилировать ядро нет необходимости).

При подготовке к компиляции ядра операционной системы Linux необходимо выполнить следующие действия:

1. В меню **Code Maturity Level Options** отметить пункт **Prompt for development and/or incomplete code/drivers**.
2. В меню **Filesystems** отметить пункт **Extended filesystem attributes (EXPRIMENTAL)**.
3. Затем отметить два подпункта: **Extended user attributes** и **Access Control Lists**.
4. В пункте **Second extended fs support** отметить подпункт **Extended attributes for ext2 (DANGEROUS)**.

После этого можно компилировать ядро операционной системы.

Для установки утилит необходимо скомпилировать пакет ACL, который также берется на сайте разработчиков. Процесс компиляции и установки подробно описан в документации, входящей в комплект пакета. На том же сайте берутся патчи к стандартным утилитам операционной системы. После этого можно произвести перезагрузку операционной системы.

Установка и изменение прав доступа

Управление списками контроля доступа производится при помощи двух утилит: getfacl и setfacl.

С помощью getfacl можно просмотреть текущие параметры доступа любого файла. Например, при вызове getfacl для домашнего каталога пользователя vasya мы получим следующее:

```
getfacl  /home/vasya
file: home/vasya
owner: vasya
group: users
user::rwx
group::---
other:---
```

Как можно видеть, каталог /home/vasya принадлежит пользователю vasya, группе users и значение прав доступа к каталогу — 0700. Каталог имеет только основные параметры доступа, поскольку изначально дополнительные права не устанавливаются.

Дополнительные права доступа к файлу устанавливаются и изменяются при помощи утилиты setfacl. Для этого используется следующий формат вызова:

```
setfacl -<опции> <ACL_структура>, <ACL_структура>,...,
<ACL_структура> <имя_файла> <имя_файла> ...
```

ACL-структура представляет собой одну из следующих конструкций:

- `[d:][u:][<пользователь>] [:[+|^]<режимы_доступа>]` — определяет режим доступа к файлу или каталогу пользователя. Если пользователь не указан, определяет режим доступа пользователя-владельца;
- `[d:] g:[<группа>] [:[+|^]<режимы_доступа>]` — то же, что и предыдущая конструкция, но для группы;
- `[d:] m [:[+|^] <режимы_доступа>]` — определяет действующие права доступа;
- `[d:] o [:[+|^] <режимы_доступа>]` — определяет режим доступа для остальных пользователей.

Для установки и изменения ACL используются следующие опции:

- `-s` — заменяет полностью ACL-файл, на указанный в командной строке;
- `-m` — изменяет режимы доступа к файлу (каталогу);
- `-x` — убирает правила доступа из ACL.

К примеру, вот что мы получим, применив setfacl к каталогу vasya:

```
setfacl -s u::rwx,g::---,o:---,u:us1:rwx,g:usrs2:rx,u:us2:--- /home/vasya
getfacl /home/dh

file: home/vasya
owner: vasya
group: users
user::rwx
user:us1:rwx
user:us2:---
group::---
group:usrs2:r-x
mask:rwx
other:---
```

Шифрование трафика

Большая часть существующих сетевых протоколов разрабатывалась по компьютерным меркам в чуть ли не доисторическую эпоху рыцарских традиций, когда о сетевых взломах и сетевом шпионаже можно было прочитать только в научной фантастике. Как результат — подавляющее большинство данных в сети Интернет передаются в открытом виде. И как обратная сто-

рона медали — существует большое количество утилит для прослушивания сетевого трафика. Многие из них умеют сами анализировать перехватываемые данные. С помощью таких утилит можно получить пароли пользователей для различных сетевых служб, тексты электронных писем, файлы, сообщения, переданные по ICQ, и т. д. Защитить себя от такого прослушивания можно с помощью шифрования трафика.

Наиболее распространенным протоколом шифрования является протокол SSL, разработанный Netscape Communications. Чаще всего он используется для шифровки протокола HTTP (HTTPS), но также может применяться для создания защищенных соединений с SMTP, POP3, IMAP и другими высокоуровневыми сетевыми протоколами.

Программа, осуществляющая поддержку протокола SSL почти для любых серверных и клиентских приложений под Linux и Windows, называется Stunnel. Основное ее применение состоит в создании надежного шифрованного канала между двумя и более хостами в сетях, где существует угроза прослушивания трафика.

Stunnel

Как обычно, рекомендуется получить с сайта разработчика последнюю версию программного пакета.

Установка

Для работы Stunnel необходимо установить OpenSSL. Обычно OpenSSL инсталлируется при установке операционной системы Linux (по крайней мере, в дистрибутиве Red Hat Linux), поэтому проблем возникнуть не должно. Пакет Stunnel обычно входит в состав дистрибутива в виде RPM-пакета.

Организация шифрованного туннеля

Stunnel может работать в двух режимах: сервера и клиента. В качестве сервера Stunnel открывает указанный порт, дешифрует все поступившие данные и передает их либо в указанную в параметрах запуска программу, либо на указанный порт на указанном хосте. В качестве клиента Stunnel открывает указанный порт, шифрует все поступившие на него данные и передает их в определенную программу или на определенный порт на заданном хосте.

Давайте организуем защищенное Telnet-соединение (хотя это и не имеет практической пользы, поскольку есть SSH) между двумя компьютерами А и Б.

На компьютере Б запускаем Stunnel в режиме сервера:

```
stunnel -d 999 -r 23
```

Опция `-d` указывает Stunnel работать в режиме отдельного демона, ждущего соединения по порту 999. Все данные, полученные в шифрованном виде на порт 999, в открытом виде передаются на порт 23 на локальной машине.

Затем на компьютере А запускаем Stunnel в режиме клиента:

```
stunnel -c -d 1055 -r B:999
```

Опция `-c` указывает на работу в режиме клиента, все данные, полученные в открытом виде на порт 1055, передаются в шифрованном виде на порт 999 на хосте Б.

После проделанных манипуляций можно устанавливать Telnet-соединение с компьютером Б. Команда запуска telnet на компьютере А будет выглядеть следующим образом:

```
telnet localhost 1055
```

Несколько непривычно, зато трафик полностью шифруется. Точно по такому же принципу организовывается шифрованный туннель и для других сетевых протоколов.

Stunnel и приложения, поддерживающие SSL

Достаточно часто возникает ситуация, когда одно из приложений поддерживает протокол SSL, а приложение с другой стороны его не поддерживает. В этом случае Stunnel можно запускать только с одной стороны — там, где приложение не способно поддерживать протокол SSL. Но тогда возникает проблема — какие порты используются приложением, поддерживающим протокол SSL.

Существует официальный список SSL-портов, который приведен далее:

```
https        443/tcp   # HTTP-протокол поверх TLS/SSL
smtps        465/tcp   # SMTP-протокол поверх TLS/SSL (называемый SSMTP)
nntps        563/tcp   # NNTP-протокол поверх TLS/SSL (называемый SNNTP)
imap4-ssl    585/tcp   # IMAP4+SSL
sshell       614/tcp   # SSLshell
ldaps        636/tcp   # LDAP-протокол поверх TLS/SSL (называемый SLDAP)
ftps-data    989/tcp   # FTP-протокол (данные) поверх TLS/SSL
ftps         990/tcp   # FTP-протокол (управление)поверх TLS/SSL
telnets      992/tcp   # Telnet-протокол поверх TLS/SSL
imaps        993/tcp   # IMAP4-протокол поверх TLS/SSL
ircs         994/tcp   # IRC-протокол поверх TLS/SSL
pop3s        995/tcp   # POP3-протокол поверх TLS/SSL (называемый SPOP3)
```

Сертификаты

Программа Stunnel имеет возможность проверки подлинности сертификатов тех хостов, к которым или с которых идет подключение. Для этого предна-

значена опция командной строки `-v`. После `-v` необходимо указать уровень проверки сертификата. Он может иметь следующие значения:

- `0` — никакой проверки наличия и подлинности сертификата не производится;
- `1` — сертификат проверяется на подлинность, если присутствует. Если сертификат не является подлинным — соединение не устанавливается;
- `2` — проверяется присутствие сертификата и его подлинность. Если сертификат отсутствует или не является подлинным — соединение не устанавливается;
- `3` — проверяется присутствие сертификата и его наличие в списке проверенных сертификатов. Если сертификат отсутствует или его нет в списке проверенных сертификатов — соединение не устанавливается.

Сертификат создается при сборке пакета и помещается вместе с секретным ключом, используемым при расшифровке входящего трафика, в файл stunnel.pem.

Более полную информацию по этому программному обеспечению смотрите в документации, идущей в комплекте с программой Stunnel.

Утилиты сканирования и защиты сети

Утилиты сканирования — это класс программного обеспечения, предназначенный для нахождения уязвимостей в конфигурации компьютера или сети. Они могут быть использованы и как средство для улучшения безопасности системы, и как инструмент для взлома системы.

SATAN

Это одна из старейших утилит сканирования. Говорят, что автора этого пакета уволили из фирмы, где он работал, из-за того, что он выложил SATAN на свой Web-сайт.

SATAN может работать на нескольких операционных системах. Он считается устаревшим, но, тем не менее, для проверки правильности основных сетевых настроек вполне пригоден. Работает от пользователя root, требует наличия Perl.

После запуска SATAN становится WWW-сервером и запускает браузер Netscape, поскольку его интерфейс Web-ориентирован. Для начала сканирования необходимо указать сканируемый хост или диапазон адресов и "уровень нападения", который может быть слабым, нормальным и тяжелым. После этого кнопкой **Start the scan** запускается сканирование.

По окончании сканирования необходимо перейти в раздел **Reporting & Data Analysis**, в котором можно ознакомиться с найденными проблемами, требующими устранения.

Portsentry

Это еще один программный продукт, предназначенный для обнаружения сканирования сетевых портов. Основные возможности программы Portsentry:

- обнаруживает практически все известные виды сканирования компьютеров;
- в реальном времени блокирует компьютер, производящий сканирование, посредством установленного на атакуемом компьютере брандмауэра, команду запуска которого можно задать в файле конфигурации;
- записывает в журнал операционной системы посредством syslogd информацию об атаке;
- может вызывать любую указанную в файле конфигурации программу, в ответ на сканирование или подключение к защищенному сетевому порту.

Установка и настройка

Процесс установки подробно описан в документации на программу и не вызывает трудностей, поэтому сразу перейдем к настройке программы.

Основной конфигурационный файл программы Portsentry называется portsentry.conf. Содержимое файла portsentry.conf представляет собой несколько строк, каждая из которых имеет вид:

```
ОПЦИЯ = "значение"
```

Далее приведен список основных поддерживаемых опций.

- `TCP_PORTS` — в этой опции через запятую перечисляются TCP-порты, которые проверяются программой Portsentry. При обнаружении подключения к перечисленным портам Portsentry записывает информацию об этом в системный журнал и выполняет команду, заданную пользователем, а после этого блокирует хост посредством брандмауэра. TCP-порты, открытые на защищаемом компьютере другими программами, в этот список включаться не должны;
- `UDP_PORTS` — то же, что и `TCP_PORTS`, но для UDP-портов;
- `ADVANCED_PORTS_TCP` — значение этой опции определяет верхнюю границу множества TCP-портов, которые проверяются Portsentry при работе в режиме Advanced Stealth Scan Detection Mode. Нижней границей является 1, т. е. при значении `ADVANCED_PORTS_TCP`, равном 2048, проверяется подключение к любому порту в промежутке от 1 до 2048;
- `ADVANCED_PORTS_UDP` — то же, что и `ADVANCED_PORTS_TCP`, но для UDP-портов;

- ADVANCED_EXCLUDE_TCP — TCP-порты, которые исключаются из промежутка проверяемых портов, заданного параметром ADVANCED_PORTS_TCP. Здесь обязательно нужно перечислить TCP-порты, открытые работающими программами на защищаемом компьютере;
- ADVANCED_EXCLUDE_UDP — то же, что и ADVANCED_EXCLUDE_TCP, но для UDP-портов;
- IGNORE_FILE — имя и путь к файлу с IP-адресами хостов, которые не блокируются при подключении к портам, проверяемым программой Portsentry;
- HISTORY_FILE — имя и путь к файлу с историей работы программы Portsentry. В файл записывается время блокирования, имя и IP-адрес хоста, атакованный порт, протокол;
- BLOCKED_FILE — строка, из которой формируется имя и путь к файлам, куда записывается информация о блокированных хостах;
- BLOCK_TCP — эта опция в зависимости от значения задает ответную реакцию Portsentry на сканирование портов:
  - 0 — не блокировать хост, не запускать заданную пользователем команду;
  - 1 — блокировать хост и запустить команду;
  - 2 — только запустить заданную команду.

  Команда задается при помощи опции KILL_RUN_CMD.
- BLOCK_UDP — то же, что и BLOCK_TCP, но для UDP;
- KILL_ROUTE — эта опция задает команду, которую надо выполнить для блокирования атакующего хоста. Для указания IP-адреса используется переменная $TARGET$. Переменная $PORT$ используется для указания порта, к которому было осуществлено подключение;
- KILL_HOSTS_DENY — эта опция задает строку, которая записывается в /etc/hosts.deny для блокирования доступа к сервисам, запускаемым через inetd;
- KILL_RUN_CMD — с помощью этой опции можно задать команду, запускаемую до блокирования хоста;
- SCAN_TRIGGER — данная опция задает количество разрешенных подключений к проверяемым программой Portsentry портам одного и того же хоста, прежде чем Portsentry начнет действовать. 0 определяет немедленную реакцию;
- PORT_BANNER — задает сообщение, которое будет выводиться при подключении к проверяемому Portsentry порту.

В файле portsentry.ignore необходимо перечислить IP-адреса компьютеров, которые не должны быть блокированы программой при подключении к проверяемому порту.

Запуск

Portsentry можно запускать в трех различных режимах, которые задаются в командной строке при вызове Portsentry. Одновременно можно задать только один режим работы для одного протокола.

- *Classic* — при работе в этом режиме Portsentry открывает порты, указанные в `TCP_PORTS` или `UDP_PORTS`, и ждет соединения. При попытке подключиться к такому порту происходит блокировка удаленного хоста. Этот режим работы задается опциями командной строки `-tcp` — для TCP-портов и `-udp` — для UDP-портов.
- *Enhanced Stealth Scan Detection* — этот режим используется для проверки перечисленных в `TCP_PORTS` или `UDP_PORTS` портов на предмет подключения или сканирования. Выявляет почти все виды Stealth-сканирования, а не только сканирование подключением. В отличие от режима Classic не держит открытыми порты, поэтому сканировщик получает достоверную информацию об открытых портах. Задается опциями командной строки `-stcp` — для TCP-портов и `-sudp` — для UDP-портов.
- *Advanced Stealth Scan Detection* — этот режим используется для проверки всех портов в промежутке от `1` до `ADVANCED_PORT_TCP` или `ADVANCED_PORT_UDP`. Порты, открытые другими программами и перечисленные в `ADVANCED_EXLUDE_TCP` или `ADVANCED_EXCLUDE_UDP`, исключаются из проверки. Любой компьютер, попытавшийся подключиться к порту в этом промежутке, тут же блокируется. Задается опциями командной строки `-atcp` — для TCP-портов и `-audp` — для UDP-портов.

Сетевая статистика

Очень часто администратору необходимо получить развернутую информацию по сетевому трафику — кто, когда, сколько и по какому протоколу отправлял/принимал информацию. Конечно, все это можно получить из различных LOG-файлов, однако незачем тратить время на изготовление анализаторов LOG-файлов, когда уже есть готовые программные решения.

NeTraMet

Этот программный пакет позволяет подсчитывать трафик по IP-адресам в локальной сети раздельно по типам трафика: SMTP, ICMP, HTTP, FTP, UDP, TCP и т. п. Также существует возможность подробного регистрирования трафика.

Программный пакет включает несколько составляющих:

- NeTraMet — программа-сборщик трафика. Собирает и хранит в оперативной памяти статистику с сетевых интерфейсов сервера;

- NeMaC — программа-менеджер сборщика NeTraMet. NeMaC собирает статистику и записывает ее в журнал;
- srl — компилятор правил для NeMaC;
- fd_filter — программа обработки журналов NeMaC;
- fd_extract — программа обработки результатов fd_filter.

Ключи запуска NeTraMet

Программа запускается со следующими ключами:

- `-i` *<интерфейс>* — определяет сетевой интерфейс, трафик которого будет считать NeTraMet;
- `-l` — предписывает использовать размер пакета из заголовка, а не аппаратный размер;
- `-m 614` — определяет UDP-порт, на котором будет соединяться NeTraMet с NeMaC;
- `-r` *<пароль>* — устанавливает пароль на чтение;
- `-w` *<пароль>* — устанавливает пароль на чтение/запись;
- `-f 60000` — определяет максимальное количество сетевых потоков в NeTraMet. Чем больше клиентов, трафика и степень детализации статистики, тем больше сетевых потоков.

Ключи запуска NeMaC

Программа запускается со следующими ключами:

- `-k 120` — каждые 120 секунд NeMaC будет проверять, не перезагрузился ли NeTraMet;
- `-F /var/ntm.log/$DATER.flows` — в этот файл записывать статистику;
- `-m 614` — определяет порт для управления NeTraMet;
- `-c 900` — предписывает забирать статистику с NeTraMet каждые 15 минут;
- `-p` — предписывает после записи в файл статистики данных закрывать его. Если файл не найден, то создается новый файл;
- `-L /var/ntm.log/$DATER.nemac` — журнал работы NeMaC;
- `-r /root/ntm.sh/short.3.rules` — файл с правилами.

Протоколирование

Нет смысла тратить много времени на защиту компьютера от взлома и не обращать внимания на систему протоколирования событий. Каким образом вы сможете узнать о попытке и способе взлома, не используя инструментов

для ведения LOG-файлов? В этом разделе мы познакомимся со стандартной системой ведения LOG-файлов — демоном syslogd.

Демон syslogd является частью пакета sysklogd, в который входят две программы: syslogd и klogd. Syslogd отвечает за протоколирование сообщений системы, а klogd — ядра.

Демон syslogd

Демон syslogd запускается автоматически при старте системы и обеспечивает протоколирование событий, которое используется большинством программ. Демон syslogd пишет сообщения в файлы /var/log/* в зависимости от настроек. Обычно записи в LOG-файле, создаваемом syslogd, содержат следующие поля: дата и время, имя компьютера, программа, сообщение.

Параметры запуска

В табл. 23.1 приведены основные параметры командной строки демона syslogd.

Таблица 23.1. *Основные параметры командной строки syslogd*

Параметр	Описание
`-d`	Включает режим отладки
`-f <file>`	Определяет альтернативный файл конфигурации
`-h`	По умолчанию демон не перенаправляет сообщения, которые он получает от других узлов. Этот параметр позволяет перенаправить сообщения другим хостам
`-n`	Этот параметр нужен, если syslogd запускается и контролируется программой init
`-p <socket>`	Позволяет задать другой сокет UNIX вместо /dev/log
`-r`	Позволяет принимать сообщения из сети
`-s <socket>`	Этот параметр позволяет указать дополнительный сокет, который syslog должен прослушивать
`-v`	Выводит версию syslogd

Файл конфигурации

По умолчанию используется файл конфигурации /etc/syslog.conf. Вы можете указать другой файл конфигурации с помощью опции `-f`. Типичный файл конфигурации приведен в листинге 23.1.

Листинг 23.1. Файл syslog.conf

```
# Все сообщения ядра операционной системы выводить на консоль
kern.*                                          /dev/console

# Все сообщения уровня info или выше протоколировать в файл
# /var/log/messages
# Кроме почтовых сообщений и сообщений аутентификации
*.info;mail.none;authpriv.none;cron.none        /var/log/messages

# Протоколирование аутентификации
# Файл протокола /var/log/secure
authpriv.*                                       /var/log/secure

# Все Log-сообщения почтовой системы сохранять в файле /var/log/maillog
mail.*                                           /var/log/maillog

# Все сообщения демона cron сохранять в файле /var/log/cron
cron.*                                           /var/log/cron

# Все получают аварийные сообщения
*.emerg                                          *

# Сообщения системы новостей уровня crit и выше сохранять в файле
# /var/log/spooler
uucp,news.crit                                   /var/log/spooler

# Все загрузочные сообщения хранить в файле /var/log/boot.log
local7.*                                         /var/log/boot.log
```

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` — отладочная информация;
- `info` — просто информация;
- `notice` — уведомление;
- `warn` — предупреждение;
- `err` — ошибка;
- `emerg` — критический уровень.

Первые три уровня протоколирования относятся к информационным сообщениям. Помимо этого существуют критические сообщения, которые выво-

дятся прямо на консоль. Для обозначения всех объектов и всех уровней протоколирования можно использовать символ `*`.

Сетевое протоколирование

Для обеспечения повышенной защищенности сети все сообщения можно хранить не на локальном компьютере, а передавать по сети на специальный сервер, на котором будет находиться база LOG-файлов компьютеров, подключенных к сети.

Для передачи сообщений используется протокол UDP. Для нормального функционирования необходимо в файле /etc/service раскомментировать строку `syslog 514/udp`.

После этого следует внести изменения в файл конфигурации /etc /syslog.conf — вместо файлов протоколов используйте параметр `@hostname`, где `hostname` — это имя компьютера, на который будут перенаправлены сообщения.

Имя узла желательно указать в файле /etc/hosts, поскольку демон syslogd обычно стартует раньше, чем сервер DNS.

Демон klogd

Демон klogd предназначен для перехвата и протоколирования сообщений ядра Linux. В табл. 23.2 приведены основные параметры командной строки демона klogd.

Таблица 23.2. *Основные параметры командной строки klogd*

Параметр	Описание
`-c <n>`	Устанавливает уровень сообщений, которые будут выводиться на экран
`-d`	Режим отладки
`-f <file>`	Записывает 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
`-k <file>`	Использует указанный файл в качестве файла, содержащего символьную информацию 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

Таблица 23.2 *(окончание)*

Параметр	Описание
-s	Заставляет демон klogd использовать системные вызовы для обращений к буферам сообщений ядра
-v	Выводит версию klogd

По умолчанию демон klogd запускается посредством системного вызова, для того чтобы препятствовать отображению всех сообщений на консоль. Это не распространяется на критические сообщения ядра (kernel panic), которые в любом случае будут отображены на консоли.

Защита системы после взлома

В этом разделе мы остановимся на сетевой безопасности, а именно на том моменте, когда взлом уже произошел. После обнаружения факта взлома стандартным решением является отключение взломанного компьютера от сети и полная переустановка операционной системы с последующей установкой всех обновлений программного обеспечения, используемого на компьютере. А что делать, если нет возможности вывести из работы взломанный компьютер, а защитить его все равно необходимо? Именно этот случай мы здесь и рассмотрим.

Как правило, серьезный взлом подготавливается долгое время, и о нем вы узнаете, например, от администратора какого-нибудь сервера на другом конце земного шара, да и то потому, что от вас на его сервер идет очень большой трафик или еще по каким-либо косвенным признакам. Такой взлом преследует чисто прагматические цели — воспользоваться вашей системой для дальнейшего взлома других компьютеров, устроить на вашем сервере хранилище файлов или что-нибудь в подобном роде. Причем взломщик после себя всегда оставляет на вашем компьютере набор специальных утилит, называемых rootkit.

Rootkit

Rootkit (набор инструментов администратора) — это набор утилит, которые взломщик устанавливает на взломанном компьютере после получения первоначального доступа. Rootkit обычно содержит сетевой sniffer (утилиту, способную получать и обрабатывать *весь* сетевой трафик вашей локальной сети, вне зависимости от того, какому компьютеру адресованы сетевые пакеты) для прослушивания сетевого трафика, программы для модификации LOG-файлов, позволяющие скрыть присутствие взломщика на вашем компьютере, и специально модифицированные системные утилиты, замещающие основные утилиты системы, например ps, netstat, ifconfig, killall, login.

Основное назначение rootkit — позволить взломщику возвращаться во взломанную систему и получать доступ к ней, не будучи при этом обнаруженным системным администратором. Обычно для этого используется модифицированная версия telnetd или sshd. Модифицированный сервис будет использовать сетевой порт, отличный от того, который этот демон по умолчанию прослушивает. Большинство версий rootkit снабжены модифицированными системными программами, которые замещают существующие во взломанной системе. Конкретный набор модифицируемых системных утилит весьма зависит от версии rootkit, а также нужд и квалификации взломщика, но, как правило, заменяются программы ps, w, who, netstat, ls, find, login и др., которые могут быть использованы для контроля за работой взломанной системы.

Для усложнения обнаружения подмены системных утилит большинство rootkit производят замену системных утилит на модифицированные версии, устанавливают точно такие же даты их создания и размеры файлов, поэтому простой список файлов с датой их создания и модификации, а также размером никакой пользы в обнаружении подмены системных утилит не принесут. Исходя из этого, пожалуй, лучший способ обнаружения подмены системных утилит — получить контрольную сумму файлов в системе и сохранить этот список в надежном месте — на другом компьютере или на компакт-диске.

В принципе, можно воспользоваться возможностями, предоставляемыми менеджером пакетов RPM — контрольной суммой пакета, рассчитанной по алгоритму MD5. При этом RPM использует контрольные суммы пакетов, хранящиеся в базе данных установленных RPM. Как легко заметить, данный способ не подходит для обнаружения опытных взломщиков. Причин тому две:

1. В вашей системе могут быть установлены программы не из RPM, а скомпилированные из исходных кодов — совершенно очевидно, что ваш менеджер пакетов абсолютно ничего не знает о программах, устанавливаемых без помощи RPM.
2. База данных RPM находится на взломанном компьютере, и взломщику не составляет труда модифицировать ее нужным образом или вообще повредить.

Для решения этой проблемы обычно используются специализированные программные пакеты, например Tripwire или AIDE, о которых мы поговорим далее.

Помимо перечисленного ранее, некоторые rootkit содержат сетевой анализатор пакетов и утилиты для записи нажатий клавиатурных кнопок, что позволяет взломщику с целью получения необходимой информации организовать сбор паролей и анализ сетевого трафика.

Наибольшую угрозу для безопасности вашей системы представляют rootkit, использующие загружаемые модули ядра (Loadable Kernel Module, LKM), что позволяет не подменять системные утилиты, а нарушать их правильное функционирование через ядро операционной системы.

Обнаружение rootkit

Сначала необходимо определить сам факт взлома системы. Возможным последствием взлома вашего компьютера и установки на нем rootkit может стать изменение в поведении системных утилит. Например, некоторые утилиты отказываются запускаться от имени пользователя, которому было разрешено пользоваться этими утилитами. Или ваша любимая утилита top стала выглядеть несколько иначе. Другие очень настораживающие признаки — изменение показателей сетевого трафика, а также резкое уменьшение свободного места на жестком диске.

Сканирование портов

После обнаружения взлома первое, что необходимо сделать после смены паролей, — лишить взломщика возможности проникновения в систему через сетевые порты. Поскольку взломанный компьютер не вызывает доверия, просканировать сетевые порты необходимо с другого компьютера.

Проще всего просканировать порты с помощью программы nmap. Для этого достаточно выполнить следующую команду:

```
nmap -p 1-65535 192.168.0.1
```

Указываем диапазон сканируемых портов — от 1 до 65 535, а также адрес сканируемого компьютера. После этого на консоль будет выдан список портов, протокол, применяемый для каждого порта, и сервис, который использует этот порт. Обычно всякие "специальные" программы обращаются к портам выше 1023, причем зачастую это порты с номером выше десяти тысяч.

Помимо nmap, можно воспользоваться программой lsof. Она позволяет получить список открытых на вашем компьютере сетевых портов. Для этого достаточно выполнить команду:

```
lsof -i
```

Использование RPM

Хотя чуть ранее мы утверждали, что использование RPM для обнаружения rootkit — дело бесперспективное, это не совсем так. RPM можно применить для быстрой проверки. Если он не найдет ничего подозрительного — воспользуемся другими средствами, если найдет — и на том спасибо — будем знать, что у нас не так в системе.

RPM записывает и проверяет контрольную сумму всех файлов в пакете, включая те файлы, которые должны изменяться с течением времени. О проверке контрольных сумм пакетов RPM см. *гл. 8*.

Сканер для rootkit

Пакет chkrootkit — набор утилит, используемых для выявления присутствия в системе уже известных rootkit. Chkrootkit удобен тем, что способен выявлять большое количество rootkit с помощью единственного приложения. Вдобавок к выявлению известных rootkit, он также включает несколько тестов, помогающих обнаружить новые rootkit.

Пакет chkrootkit состоит из следующих утилит:

- chkrootkit — используется для выявления сигнатур известных rootkit;
- ifpromisc — используется для обнаружения прослушивания сетевого трафика взломанным компьютером;
- chklastlog, chkwtmp, check_wtmpx — утилиты для проверки LOG-файлов;
- chkproc — предназначена для обнаружения "посторонних" загружаемых модулей ядра операционной системы.

С особенностями применения chkrootkit можно ознакомиться в документации, идущей в комплекте с пакетом.

После обнаружения

Что делать после обнаружения rootkit? Единственно верный способ избавиться от последствий взлома — заново полностью переустановить операционную систему и установить все обновления пакетов для вашего дистрибутива. Однако не всегда есть возможность проделать такие действия сразу — квартальный отчет, непрерывное производство — да мало ли что еще.

В дистрибутивах на основе RPM-пакетов вы можете определить поврежденные пакеты. После этого необходимо переустановить их, используя следующую команду:

```
rpm -U --force rpm_package_name.rpm
```

После переустановки пакетов вы должны удалить файлы, установленные в вашу систему взломщиком. Данные, полученные chkrootkit, помогут вам определить их местонахождение.

После удаления всех обнаруженных "чужих" файлов запустите top и ps для выявления и уничтожения оставшихся нежелательных процессов. Помимо этого, необходимо проверить стартовые скрипты операционной системы и убедиться, что эти скрипты не используются никакими посторонними программами.

LIDS

LIDS (Linux Intrusion Detection/Defence System, система обнаружения и защиты от вторжения) представляет собой дополнение к ядру операционной системы Linux, предоставляющее возможности для увеличения безопасности операционной системы. LIDS позволяет запретить или ограничить доступ пользователя root к файлам, памяти, устройствам, сетевым интерфейсам, запущенным приложениям и т. п. Это дает возможность надежно оградить даже взломанную операционную систему от дальнейшего вмешательства.

В отличие от других средств защиты операционной системы Linux, данную систему невозможно отключить, не зная пароля администратора LIDS, который в зашифрованном виде хранится в специальном файле, видимом только программой администрирования LIDS. Точно так же защищены и конфигурационные файлы LIDS. Даже узнав каким-то образом пароль администратора LIDS, отключить систему можно, только находясь за консолью компьютера.

LIDS позволяет распределять права доступа к файлам на уровне программ, а не на уровне пользователей, а также запретить перезапуск операционной системы, загрузку/выгрузку модулей ядра и многое другое.

Информация о всех действиях, имеющих отношение к защищаемым объектам, помимо записи в LOG-файлах может немедленно отправляться по электронной почте.

Помимо всего прочего, в LIDS присутствует встроенный детектор сканирования сетевых портов.

Установка

После получения пакета LIDS необходимо разархивировать его и наложить патч на исходные тексты ядра операционной системы Linux. После этого следуйте инструкции — там все понятно — компилируем, устанавливаем.

Далее, нам требуется перекомпилировать ядро операционной системы Linux с поддержкой LIDS. Для этого в пункте меню конфигурации ядра **Code maturity level options** необходимо включить опцию **Prompt for development and/or incomplete code/drivers**.

После этого в пункте меню **General setup** следует включить опцию **Sysctl support**.

Затем нужно зайти в меню **Linux Intrusion Detection System**. Это меню полностью относится к конфигурированию LIDS. Первым идет включение поддержки LIDS в ядре:

[*] Linux Intrusion Detection System support (EXPERIMENTAL)

После включения поддержки LIDS станет доступным список опций настройки LIDS:

- **Maximum protected objects to manage** — этот пункт позволяет установить максимальное количество защищаемых объектов;
- **Maximum ACL subjects to manage** — позволяет установить максимальное количество субъектов правил доступа LIDS;
- **Maximum ACL objects to manage** — позволяет установить максимальное количество объектов правил доступа LIDS;
- **Maximum protected proceeds** — позволяет установить максимальное количество защищаемых процессов;
- **Hang up console when raising securit alert** — разрешает закрытие консоли, с которой произошло нарушение безопасности;
- **Security alert when execing unprotected programs before sealing LIDS** — разрешает вывод сообщения о нарушении безопасности при запуске незащищенных программ;
- **Do not execute unprotected programs before sealing LIDS** — включает запрет на запуск незащищенных программ до установки способностей;
- **Try not to flood logs** — при включении этой опции LIDS не будет записывать в LOG-файлы дублирующиеся сообщения об одном и том же нарушении защиты;
- **Autorized time between two identic logs (seconds)** — устанавливается время в секундах, в течение которых проверяется появление двух идентичных сообщений, чтобы не записывать одинаковые сообщения в LOG-файлы;
- **Allow switching LIDS protections** — включает возможность отключения и включения LIDS в процессе работы системы после ввода пароля. При включении данной опции появляется возможность поменять любые параметры работы без перезагрузки операционной системы;
- **Numbers of attempts to submit password** — определяет количество попыток ввода пароля, по истечении которых отключение LIDS становится невозможным на заданный далее промежуток времени;
- **Time to wait after fail (seconds)** — время в секундах, в течение которого после ввода неправильного пароля указанное количество раз отключение LIDS становится невозможным;
- **Allow remote users to switch LIDS protections** — дает возможность удаленным пользователям отключать LIDS. С целью увеличения безопасности вашей операционной системы не включайте эту опцию;
- **Allow any program to switch LIDS protections** — позволяет любой программе отключать LIDS. Не включайте эту опцию;

- **Allow reloading config. File** — разрешает переконфигурирование LIDS без перезагрузки компьютера;
- **Port Scanner Detector in kernel** — позволяет в ядро операционной системы добавить детектор сканирования портов;
- **Send security alerts through network** — разрешает отправку электронной почты при нарушении безопасности на указанный электронный адрес с информацией о нарушении. Письмо отправляется незамедлительно при попытке совершения несанкционированных действий;
- **Hide klids network threads** — позволяет скрывать сетевые соединения LIDS;
- **Number of connection tries before giving up** — задается количество попыток соединения с SMTP-сервером;
- **Sleep time after a failed connection** — задает время в секундах между попытками соединения с почтовым сервером;
- **Message queue size** — определяет максимальное количество почтовых сообщений в очереди. При превышении данного количества самое старое неотправленное сообщение удаляется из очереди;
- **LIDS debug** — используется для включения вывода отладочных сообщений LIDS.

После конфигурирования можно компилировать и устанавливать ядро операционной системы.

Конфигурирование LIDS

После установки LIDS в каталоге /etc появляется каталог lids, содержащий следующие конфигурационные файлы:

- lids.cap — предназначен для хранения текущих значений установок способностей;
- lids.net — предназначен для настройки отправки электронных сообщений системой LIDS;
- lids.pw — в этом файле записан в зашифрованном виде пароль администратора. Изменять этот файл можно только с помощью lidsadm;
- lids.conf — файл содержит текущие установки правил доступа. Изменять этот файл можно только с помощью lidsadm.

Способности

Способности (capabilities) определяют возможность программ совершать какие-либо действия. LIDS позволяет использовать по отношению к программам большое количество способностей. В частности, LIDS поддерживает способность перезагружать компьютер, изменять владельца файла, загружать или выгружать модули ядра и многое другое.

Текущие установки способностей хранятся в файле lids.cap в формате:

```
[+|-] Номер:Способность
```

Здесь:

- + — включает способность;
- - — отключает способность.

Редактировать файл lids.cap можно с помощью любого текстового редактора. Включение способности влияет на все программы без исключения, а выключение — на все программы, кроме тех, которым напрямую указана данная способность с помощью правил доступа lidsadm.

Сразу после установки LIDS файл lids.cap содержит включенными следующие способности:

- CAP_CHOWN — устанавливает способность программ изменять владельца и группу владельца файла;
- CAP_DAC_OVERRIDE — разрешает программам, запускаемым пользователем root, не принимать во внимание режимы доступа к файлам. При отключении этой способности пользователь root теряет возможность изменять любые файлы, невзирая на права доступа;
- CAP_DAC_READ_SEARCH — то же самое, что и предыдущая способность, только по отношению к каталогам;
- CAP_FOWNER — разрешает операции с файлами, когда владелец файла должен совпадать с пользователем, совершающим операцию;
- CAP_FSETID — разрешает установку SUID- или SGID-бита на файлах, не принадлежащих пользователю root;
- CAP_KILL — разрешает процессам пользователя root "убивать" чужие процессы;
- CAP_SETGID — управляет способностью программ пользователя root изменять группу, под которой работает программа;
- CAP_SETUID — управляет способностью программ пользователя root изменять пользователя, под которым работает программа;
- CAP_SETPCAP — разрешает программам менять способности;
- CAP_LINUX_IMMUTABLE — управляет способностью снимать атрибуты S_IMMUTABLE и S_APPEND с файлов;
- CAP_NET_BIND_SERVICE — разрешает программам использовать сетевой порт, меньший чем 1024;
- CAP_NET_BROADCAST — управляет способностью программ рассылать широковещательные пакеты;
- CAP_NET_ADMIN — параметр управляет большим количеством различных способностей: конфигурирование сетевых интерфейсов, изменение пра-

вил брандмауэра, изменение таблиц маршрутизации и многих других, связанных с сетевыми настройками Linux;

- CAP_NET_RAW — управляет способностью программ использовать сокеты;
- CAP_IPC_LOCK — управляет способностью процессов пользователя root блокировать сегменты разделяемой памяти;
- CAP_IPC_OWNER — управляет доступом программ пользователя root к ресурсам межпроцессорного взаимодействия процессов, не принадлежащих пользователю root;
- CAP_SYS_MODULE — управляет способностью загружать модули ядра;
- CAP_SYS_RAWIO — управляет доступом на чтение/запись к таким устройствам, как /dev/mem, /dev/kmem, /dev/port, /dev/hdXX, /dev/sdXX;
- CAP_SYS_CHROOT — управляет способностью устанавливать корневой каталог для текущей командной оболочки;
- CAP_SYS_PTRACE — этот параметр включает способность программ использовать вызов функции ptrace(), которая позволяет процессу-родителю управлять выполнением процессов-потомков;
- CAP_SYS_PACCT — управляет способностью конфигурировать учет процессов;
- CAP_SYS_ADMIN — управляет множеством способностей: управление устройством /dev/random, создание новых устройств, конфигурирование дисковых квот, настройка работы klogd, установка имени домена, установка имени хоста, сброс кэша, монтирование и размонтирование дисков, включение-отключение Swap-раздела, установка параметров последовательных портов и многое другое;
- CAP_SYS_BOOT — управляет способностью перегружать систему;
- CAP_SYS_NICE — управляет способностью изменять приоритет процессов, не принадлежащих пользователю root;
- CAP_SYS_RESOURCE — управляет способностью изменять лимиты использования ресурсов системы: дисковые квоты, зарезервированное пространство на Ext2-разделах, максимальное количество консолей и т. п.;
- CAP_SYS_TIME — управляет способностью изменять системное время;
- CAP_SYS_TTY_CONFIG — управляет способностью изменять настройки tty-устройств;
- CAP_HIDDEN — управляет способностью программ делаться невидимыми в списке процессов. Не влияет на все программы;
- CAP_INIT_KILL — управляет способностью "убивать" процессы-потомки процесса init.

Как видите, впечатляющий набор возможностей. Самое время разобраться, что из этого нужно включить, а что выключить для вашей операционной системы.

Для инициализации параметров способностей в процессе загрузки используется команда

```
lidsadm -I
```

Обычно ее ставят в конце /etc/rc.d/rc.local, что позволяет произвести отключение способностей только после запуска всех необходимых для работы сервера программ.

Правила доступа

Все управление LIDS осуществляется с помощью программы — lidsadm. Lidsadm работает в двух режимах: настройка правил доступа и ввод команд администрирования. Установки правил доступа находятся в файле /etc /lids/lids.conf. Для просмотра текущих установок правил доступа необходимо выполнить следующую команду:

```
lidsadm -L
LIST
Subject ACCESS TYPE Object
-----------------------------------------------------
Any File READ /sbin
Any File READ /bin
Any File READ /boot
Any File READ /lib
Any File READ /usr
Any File DENY /etc/shadow
/bin/login READ /etc/shadow
/bin/su READ /etc/shadow
Any File APPEND /var/log
Any File WRITE /var/log/wtmp
/sbin/fsck.ext2 WRITE /etc/mtab
Any File WRITE /etc/mtab
Any File WRITE /etc
/usr/sbin/sendmail WRITE /var/log/sendmail.st
/bin/login WRITE /var/log/lastlog
/bin/cat READ /home/xhg
Any File DENY /home/httpd
/usr/sbin/httpd READ /home/httpd
Any File DENY /etc/httpd/conf
/usr/sbin/httpd READ /etc/httpd/conf
/usr/sbin/sendmail WRITE /var/log/sendmail.st
/usr/X11R6/bin/XF86_SVGA NO_INHERIT RAWIO
```

```
/usr/sbin/in.ftpd READ /etc/shadow
/usr/sbin/httpd NO_INHERIT HIDDEN
```

Правила доступа состоят из трех элементов: субъекта, объекта и цели. *Объектом* является любой файл или каталог, на который должны действовать правила доступа и защита LIDS. Если в качестве объекта указывается каталог, то все файлы в нем и вложенные каталоги с их файлами автоматически становятся объектами. *Субъектом* является любая защищенная программа, которой дают доступ к защищаемому объекту, поэтому прежде чем использовать программу в качестве субъекта, ее саму надо защитить средствами LIDS, применив к ней правила доступа как к объекту. Если субъект не указан, то субъектом является любая программа. *Целью* является тип доступа:

- READ — доступ на чтение;
- WRITE — запись;
- DENY — запрет на какой-либо доступ;
- APPEND — открытие только для записи в конец файла;
- IGNORE — игнорирование защиты.

Построение прав доступа подробно описано в документации на пакет LIDS, поэтому мы на этом здесь не останавливаемся.

После настройки LIDS необходимо перезагрузить операционную систему. В том случае, если с функционированием LIDS возникли проблемы, можно загрузить Linux с выключенным LIDS, для чего при загрузке необходимо передать ядру операционной системы параметр `security=0`. Например, для загрузчика LILO это будет выглядеть так:

```
LILO boot: linux security=0
```

Tripwire

Программный пакет Tripwire предназначен для обнаружения изменения файлов, позволяя обнаруживать порчу данных и взломы. База данных контрольных сумм файлов шифруется, что предотвращает ее подделку взломщиками.

Непосредственно после установки операционной системы необходимо установить Tripwire, который, используя правила, определенные политикой безопасности, создает базу данных, содержащую информацию обо всех файлах в системе (список файлов может задаваться администратором): размер, контрольная сумма, дата модификации и т. п. После создания базы данных она ежедневно сравнивается с текущим состоянием файловой системы, позволяя обнаружить добавленные, измененные и удаленные файлы. Получаемые при этом отчеты могут быть просмотрены с различной степенью детализации.

Пакет Tripwire входит в состав практически всех современных дистрибутивов Linux.

Portsentry

Программа предназначена для обнаружения попыток сканирования портов и организации адекватного (с точки зрения администратора) ответа. Основные возможности Portsentry:

- обнаруживает практически все известные виды сканирования UNIX-машин;
- в реальном времени блокирует хост, с которого происходит сканирование портов, посредством установленного на атакуемом компьютере брандмауэра;
- записывает в LOG-файлы посредством syslogd информацию об атаке;
- в ответ на сканирование или подключение к защищенному порту вызывает программу, указанную администратором при конфигурировании.

Пакет Portsentry прост в установке, конфигурировании и использовании. Его можно получить как в исходных кодах, так и в виде RPM-пакета.

LogSentry

Программа LogSentry предназначена для автоматического мониторинга LOG-файлов и уведомления системного администратора с помощью электронной почты о подозрительных происшествиях в системе. Гибко настраивается.

AIDE

Пакет AIDE — система обнаружения вторжений, основанная на использовании мониторинга изменения контрольных сумм защищаемых файлов операционной системы. Система AIDE разработана таким образом, что ее полная инсталляция помещается на одной дискете. Это позволяет избежать вмешательства взломщика в функционирование программы.

Функционально программа является аналогом Tripwire, только имеет более простые конфигурационные файлы и интерфейс.

RSBAC

RSBAC — это надстройка над ядром Linux и комплект утилит управления, позволяющие создать на базе Linux защищенную систему. Реализация механизмов защиты выполнена на уровне ядра системы, системные вызовы, за-

трагивающие безопасность, дополняются специальным кодом, выполняющим обращение к центральному компоненту RSBAC. Этот компонент принимает решение о допустимости данного системного вызова на основе многих параметров:

- типа запрашиваемого доступа (чтение, запись, исполнение);
- субъекта доступа;
- атрибутов субъекта доступа;
- объекта доступа;
- атрибутов объекта доступа.

Функционально RSBAC состоит из нескольких модулей, а центральный компонент принимает комплексное решение, основываясь на результатах, возвращаемых каждым из активных в данный момент модулей (какие модули задействовать и в каком объеме определяется на этапе настройки системы).

Вся информация, используемая RSBAC, хранится в дополнительном каталоге, который доступен только ядру системы.

RSBAC — мощная система защиты и разграничений прав доступа, но несколько тяжеловата в настройке.

Security-Enhanced Linux

Security-Enhanced Linux имеет аналогичное с RSBAC назначение и представляет собой дополнения к ядру, а также набор утилит. Разработка Security Enhanced Linux продвигается Агентством Национальной Безопасности США (National Security Agency, NSA). Security-Enhanced Linux обеспечивает гибкую мандатную архитектуру управления доступом, использующую язык описания конфигураций политики безопасности.

По сравнению с RSBAC система Security-Enhanced Linux менее гибкая, зато имеет очень хорошую предопределенную политику безопасности. Ее настройка достаточно сложна и невозможна без изучения специального языка конфигурации.

Литература и ссылки

- Мяснянкин В. В. Linux на защите информации. Открытые системы № 04/2001.
- REFERENCE MANUAL NeTraMet & NeMaC. Nevil Brownlee.
- **acl.bestbits.at** — официальная страница проекта Linux ACL.
- **bog.pp.ru/work/tripwire.html** — Bog BOS. Tripwire: принципы работы, установка и настройка.

- **freshmeat.net/projects/netramet/** — страница проекта NeTraMet.
- **gazette.linux.ru.net/lg75/articles/rus-maiorano.html** — Maiorano A. Инсталляция и использование AIDE. Перевод Куприна А.
- **linux.ru.net/~inger/RSBAC-DOC-ru.html** — начала RSBAC.
- **linuxrsp.ru/artic/portsentry.html** — Ерижоков А. А. Portsentry.
- **linuxrsp.ru/artic/posixacls.html** — Ерижоков А. А. Списки контроля доступа.
- **linuxrsp.ru/artic/stunnel.html** — Ерижоков А. А. Stunnel: Шифрование трафика.
- **linuxsecurity.com** — сайт, посвященный безопасности операционной системы Linux.
- **rootshell.com** — сайт, посвященный безопасности операционных систем.
- **stunnel.mirt.net** — официальный сайт пакета Stunnel.
- **www.chkrootkit.org** — официальный сайт chkrootkit.
- **www.cs.tut.fi/~rammer/aide.html** — страница разработчика AIDE.
- **www.false.com/security/linux/** — Secure Linux patches by Solar Designer — дополнения к ядру Linux, повышающие безопасность операционной системы.
- **www.insecure.org** — местонахождение программы nmap — сканера сетевых портов.
- **www.lids.org** — сайт проекта LIDS.
- **www.linuxrsp.ru/artic/lids.html** — Ерижоков А. А. LIDS — система обнаружения и защиты от вторжения.
- **www.linuxrsp.ru/artic/portsentry.html** — Ерижоков А. А. Portsentry.
- **www.monkey.org/~dugsong/dsniff** — страничка программы-снифера Dsniff.
- **www.opennet.ru/docs/RUS/netramet/index.html** — Матыцын Д. Сбор статистики по TCP/IP на базе NeTraMet.
- **www.psionic.com** — сайт Psionic Software, разработчика программы Portsentry.
- **www.softerra.ru/freeos/16901/** — Altunergil O. Понятие Rootkit. Перевод Захаровой И.
- **www.softerra.ru/freeos/16999/** — Altunergil O. Сканирование для обнаружения Rootkit. Перевод Захаровой И.
- **www.softerra.ru/freeos/17032/** — Колисниченко Д. Протоколирование.
- **www.securityfocus.com/infocus/1633** — Alan Neville, Eric Hines with additional research by Joseph Kelly Footprints in the Sand: Fingerprinting Exploits in System and Application Log Files.
- **www.tripwire.org** — сайт разработчиков Tripwire.

ПРИЛОЖЕНИЕ 1

Литература

Здесь приведен список литературы, использованной при написании книги. Практически все перечисленные книги заслуживают внимания. Некоторые из них интересны глубоким освещением теоретической части, некоторые — более привлекательны для практиков. Большая часть книг, как нам представляется, предназначена для специалистов среднего и высокого уровня.

- Немет Э., Снайдер Г., Сибасс С., Трент Р. Хейн. UNIX: руководство системного администратора / 2-е изд.: Пер. с англ. — К.: BHV, 1997.

  Уникальная книга. По охвату вопросов, раскрытию тем аналогичной книги, пожалуй, не найти. Написана специалистами-практиками и для практиков. Ориентирована на опытного пользователя. Первое и второе издание книги посвящены коммерческим реализациям UNIX и, на первый взгляд, имеют отдаленное отношение к операционной системе Linux. Однако знание UNIX — это 98% успеха в освоении Linux.

- Немет Э., Снайдер Г., Сибасс С., Хейн Т. Р. UNIX: руководство системного администратора. Для профессионалов: Пер. с англ. — СПб.: Питер; К.: Издательская группа BHV, 2002.

  Третье издание книги. Она существенно переработана. Уменьшено количество рассматриваемых операционных систем и, что очень важно, среди них появилась Linux. Эта книга должна быть на столе у каждого администратора.

- Карлинг М., Деглер С., Деннис Дж. Системное администрирование Linux / Учебное пособие: Пер. с англ. — М.: Издательский дом "Вильямс", 2000.

  Хорошая книга по администрированию системы. Некоторые вопросы затронуты неглубоко, однако направление поиска задают верно. Очень рекомендуется для системных администраторов и специалистов IT.

- Такет Джек (мл.), Гантер Д. Использование Linux / 3-е изд.: Пер. с англ. — К.; М.; СПб.: Издательский дом "Вильямс", 1998.

  Хорошая книга для начинающих пользователей. Переиздавалась несколько раз. В освоении операционной системы Linux на первых порах очень помогает.

- Зиглер Р. Брандмауэры в Linux. / Учебное пособие: Пер. с англ. — М.: Издательский дом "Вильямс", 2000.

  Книга полностью посвящена построению защищенной сети. Рекомендуется для системных администраторов.

- Максвел С. Ядро Linux в комментариях: Пер. с англ. — К.: ДиаСофт, 2000.

  Содержание этой книги понятно из названия. С выходом ядер версии 2.4 информация несколько устарела, однако книга весьма полезна для ознакомления с общей идеологией ядра операционной системы Linux.

- Робачевский А. М. Операционная система UNIX. — СПб.: БХВ — Санкт-Петербург, 1999.

  Хорошая теоретическая книга. В ней объясняются принципы функционирования операционной системы, основные понятия, протокол TCP/IP и многое другое.

- Рейчард К., Фолькердинг П. Linux: справочник. — СПб: Питер Ком, 1999.

  Хороший справочник по программам, утилитам и командам операционной системы Linux. Некоторые утилиты устарели, некоторые важные утилиты не описаны, но в целом эта книга — хорошее приобретение на книжную полку и пользователю, и администратору.

- Шевель А. Linux. Обработка текстов. Специальный справочник. — СПб.: Питер, 2001.

  Книга посвящена текстовым редакторам и системе CVS — управления и контроля версий текстов (исходных кодов программного обеспечения).

- Блам Р. Система электронной почты на основе Linux / Учебное пособие: Пер. с англ. — М.: Издательский дом "Вильямс", 2001.

  В книге рассматриваются настройка почтового сервера и конфигурирование почтовых клиентов. Хорошая книга для освоения начал работы с почтовыми сообщениями и построения простых почтовых серверов.

- Птицын К. Серверы Linux. Самоучитель. — М.: Издательский дом "Вильямс", 2003.

  Небольшая книга, предназначенная для начинающих. В краткой форме рассмотрены основные моменты установки и настройки сервера на базе дистрибутива Red Hat.

- Манн С., Митчелл Э., Крелл М. Безопасность Linux / 2-е изд. — М.: Издательский дом "Вильямс", 2003.

  В книге рассказывается об инсталляции, конфигурировании и сопровождении Linux-систем с точки зрения безопасности. Это руководство администратора по реализации стратегии защиты Linux, а также по утилитам защиты, существующим в Linux. Книга предназначена для пользователей средней и высокой квалификации.

- Немет Э., Снайдер Г., Хейн Т. Руководство администратора Linux — М.: Издательский дом "Вильямс", 2003.

  Хит от известных авторов, рекомендую. В книге рассмотрены три основных дистрибутива Linux: Red Hat 7.2, SuSE 7.3 и Debian 3.0. Это одна из немногих книг, предназначенных не для широкого круга пользователей, а для системных администраторов, работающих в среде Linux.

- Водолазкий В. Путь к Linux: учебный курс / 3-е изд. — СПб.: Питер, 2002.

  Книга от мэтра! Впервые была прочитана автором в году 96—97. Несмотря на такой почтенный возраст, книга современна и полезна для начинающих пользователей.

- Костромин В. А. Самоучитель Linux для пользователя. — СПб.: БХВ-Петербург, 2002.

  Книга от создателя "Виртуальная энциклопедия "LINUX по-русски" (**http://rus-linux.net**). Рекомендую.

ПРИЛОЖЕНИЕ 2

Ссылки

Здесь собраны Web-ресурсы, тем или иным образом связанные с операционной системой Linux. Приведенный список далеко не полон, вряд ли он охватывает даже малую часть таких ресурсов.

- **bog.pp.ru/work/apache.html** — Apache: HTTP-сервер. Установка, настройка и русификация.
- **www.cs.ifmo.ru/education/documentation/rapacheman/index.shtml** — Подстрешный А. Работа с Web-сервером Russian Apache.
- **www.apache.org** — официальный сервер Apache.
- **apache.lexa.ru** — сервер группы разработчиков русского модуля Apache.
- **www.Squid-cache.org** — официальный сайт Squid.
- **karjagin.narod.ru/solaris/Squid-faq-rus.html** — русский перевод Squid FAQ.
- **www.nlanr.net/Cache/ICP/ICP-id.txt** — протокол Internet Cache Protocol.
- **Squid.org.ua** — зона особого внимания: сайт, полностью посвященный программе Squid.
- **linux.webclub.ru/security/proxy/Squid.html** — Паскаль И. Настройка Squid.
- **www.bog.pp.ru/work/Squid.html** — Bog BOS: Squid (кэширующий Proxy для HTTP): установка, настройка и использование.
- **www.nitek.ru/~igor/Squid** — борьба с баннерами.
- **www.bog.pp.ru/work/xntpd.html** — Богомолов С. Bog BOS: xntpd (UNIX-сервер NTP — Network Time Protocol).
- **www.linuxoid.ru/how_to/samba5.html** — Басин И. Samba за пять минут.
- **www.samba.org** — официальный сайт проекта Samba.
- **www.webmin.com** — официальный сайт проекта Webmin.
- **www.culte.org/projets/developpement/gsmb/** — официальный сайт проекта GSMB.

- **boombox.campus.luth.se/sambasentinel.php** — сайт проекта sambasentinel.
- **www.linux.org.ru/books/HOWTO/SMB-HOWTO.html** — SMB HOWTO (русский перевод).
- **www.linuxcenter.ru/lib/soft/samba_pdc.phtml** — как настроить Samba 2.2 в качестве основного контроллера домена (Primary Domain Controller).
- **www.compu-art.de/mars_nwe** — домашняя страница mars_new.
- **www.osp.ru/pcworld/1998/05/44.htm** — Суханов А., Хименко В. Linux и Windows 95: эффективность совместной работы. Мир ПК № 5/98.
- **www.linuxrsp.ru/artic/print_server.html** — Лушня Ю. Печать в Linux с железными нервами.
- **linuxcenter.ru/lib/hardware/usbprinter.phtml** — Лушня Ю. Настраиваем USB-принтер под Linux.
- **linux.yaroslavl.ru/Docum/Rus/print.html** — Толпекин В. Настройка сетевого принтера для печати русского текста.
- **www.astart.com/lprng/LPRng.html** — страница проекта LPRng.
- **www.freebsd.org/~andreas/#apsfilter** — страница APSFILTER: Магический фильтр для печати.
- **metalab.unc.edu/pub/Linux/system/printing/** — lprMagic: Фильтр печати с неплохими возможностями.
- **feynman.tam.uiuc.edu/pdq/** — страница PDQ.
- **ftp://ppr-dist.trincoll.edu/pub/ppr/** — местонахождение PPR — системы буферизации печати, ориентированной на Postscript.
- **http://www.linuxdoc.org/** — сайт, содержащий много интересной документации по Linux на английском языке.
- **www.citycat.ru/linux/docs/index.html** — сайт, содержащий много интересной документации по Linux на русском языке.
- **www.l0pht.com/~weld/netcat/** — страница netcat, пакета для работы с принтером.
- Документация на принт-сервер Surecom.
- **www.penguincomputing.com/prtools/npadmin.html** — страница npadmin — программы для управления сетевыми принтерами. Управление осуществляется через SNMP.
- **www.linux.org.ru/books/gateway/** — Костарев А. Ф. ОС Linux как мост между локальной сетью и Internet.
- **lin-omts.airport.sakhalin.ru/departs/ccito/guide1.htm** — как установить, настроить и запустить Web-узел UNIX не тратя лишних денег, сил и здоровья.

- **people.ee.ethz.ch/~oetiker/webtools/mrtg/mrtg.html** — описание MRTG.
- **www.mrtg.org** — официальный сайт пакета MRTG.
- **rrdtool.eu.org** — официальный сайт пакета rrdtool.
- **www.geocities.com/SiliconValley/Pines/7895/PPP.DOC** — Водолазкий В. Установка PPP-соединения в Linux.
- **linux.perm.ru/doc/net/mrtg.html** — Богомолов С. Мониторинг загрузки каналов (и не только) MRTG.
- **www.bog.pp.ru/work/rrdtool.html** — Богомолов С. RRDtool — хранение и отображение данных мониторинга.
- **linux.uatel.net.ua/ipcount.phtml** — как оперативно подсчитать IP-трафик клиента.
- **ftp://ftp.kiev.farlep.net/pub/os/linux/soft/trafficcounter-snmp** — универсальный счетчик IP-трафика через SNMP.
- **www.vadim.org.ua//index.php?cmd=article3** — пример простой системы учета трафика.
- **exorsus.net/projects/net-acct/** — домашняя страница проекта net-acct.
- **www.infocity.kiev.ua/inet/content/inet091.phtml** — протоколы SLIP/CSLIP и PPP.
- **www.protocols.ru** — энциклопедия сетевых протоколов.
- **gazette.linux.ru.net/lg86/vinayak.html** — исследование TCP/IP с помощью TCPdump и Tethereal. Автор: Vinayak Hegde. Перевод: Песин И.
- **msk.nestor.minsk.by/kg/2003/06/kg30604.html** — компьютерная газета Linux в Сети. Сага четвертая. Подготовил X-Stranger.
- **msk.nestor.minsk.by/kg/2003/05/kg30503.html** — компьютерная газета Linux в Сети. Сага третья. Подготовил X-Stranger.
- **www.citforum.ru/nets/semenov/5/dia_5.shtml** — диагностика локальных сетей и Интернет. Семенов Ю. А. (ГНЦ ИТЭФ).
- **www.osp.ru/os/2001/02/073.htm** — Облаков И. Восход солнца вручную.
- **www.multik.ru/linux/linuxvpn/** — Калошин В. Установка VPN Linux-сервера.
- **www.bruy.info/vpn.html** — Бруй В., Карлов В. Linux-сервер: пошаговые инструкции инсталляции и настройки.
- **securitylab.ru/34649.html** — Разумов М. По материалам Sys Admin Magazine. Администрирование IPSec VPN под Linux. Часть первая. Администрирование IPSec VPN под Linux.
- **securitylab.ru/34764.html** — Разумов М. По материалам Sys Admin Magazine. Администрирование IPSec VPN под Linux. Примеры конфигураций.

- **www.opennet.ru/base/net/vpn_pptp.txt.html** — Коптев Д. Настройка сервера VPN (PPTP) под Linux.
- **www.freeswan.org** — официальный сайт FreeS/WAN.
- **www.linuxfocus.org/Russian/September1998/article63.html** — Яп К. Введение в сетевую загрузку и протокол Etherboot.
- **alst.odessa.ua** — Стахнов А. Удаленная загрузка. Сервер Linux. Клиентская часть DOS, Windows 3.1. Инструкция по установке и настройке.
- **www.homepc.ru/offline/2002/78/22489/** — Вагнер В. Всем сестрам по терминалу.
- **etherboot.sourceforge.net/** — сайт пакета Etherboot.
- **www.menet.umn.edu/~kaszeta/unix/xterminal/index.html** — Kaszeta R. A new use for old and outdated PCs.
- **ppg.ice.ru/ppg/xterminals** — Вагнер В. Использование бездисковых X-терминалов на базе Linux-PC.
- **netstation.sf.net** — организация бездисковых станций.
- **www.nilo.org** — сетевая загрузка бездисковых компьютеров.
- **pxes.sourceforge.net** — программное обеспечение для организации загрузки бездисковых клиентов с сетевыми картами, использующими PXE.
- **www.bog.pp.ru/work/ssh.html** — Bog BOS: SSH и OpenSSH: принципы работы, установка и настройка.
- **www.ssh.com** — сайт коммерческой реализации SSH.
- **www.openssh.com** — сайт некоммерческой реализации SSH.
- **www.tigerlair.com/ssh/faq/** — SSH FAQ.
- **lib.ru/LABIRINT/telnet.htm** — доступ к ресурсам Интернета в режиме удаленного терминала.
- **www.mnet.uz/citforum/internet/services/index.shtml** — Храмцов П. Б. Администрирование сети и сервисов Internet. Учебное пособие.
- **acl.bestbits.at** — официальная страница проекта Linux ACL.
- **bog.pp.ru/work/tripwire.html** — Bog BOS: Tripwire: принципы работы, установка и настройка.
- **freshmeat.net/projects/netramet/** — страница проекта NeTraMet.
- **gazette.linux.ru.net/lg75/articles/rus-maiorano.html** — Maiorano A. Инсталляция и использование AIDE. Перевод Куприна А.
- **linuxrsp.ru/artic/portsentry.html** — Ерижоков А. А. Portsentry.
- **linuxrsp.ru/artic/posixacls.html** — Ерижоков А. А. Списки контроля доступа.

- **linuxrsp.ru/artic/stunnel.html** — Ерижоков А. А. Stunnel: Шифрование трафика.
- **linuxsecurity.com** — сайт, посвященный безопасности операционной системы Linux.
- **rootshell.com** — сайт, посвященный безопасности операционных систем.
- **stunnel.mirt.net** — официальный сайт пакета Stunnel.
- **www.chkrootkit.org** — официальный сайт chkrootkit.
- **www.cs.tut.fi/~rammer/aide.html** — страница разработчика AIDE.
- **www.false.com/security/linux/** — Secure Linux patches by Solar Designer — дополнения к ядру Linux, повышающие безопасность операционной системы.
- **www.insecure.org** — местонахождение программы nmap — сканера сетевых портов.
- **www.lids.org** — сайт проекта LIDS.
- **www.linuxrsp.ru/artic/lids.html** — Ерижоков А. А. LIDS — система обнаружения и защиты от вторжения.
- **www.monkey.org/~dugsong/dsniff** — страничка программы-снифера Dsniff.
- **www.psionic.com** — сайт Psionic Software, разработчика программы Portsentry.
- **www.softerra.ru/freeos/16901/** — Altunergil O. Понятие Rootkit. Перевод Захаровой И.
- **www.softerra.ru/freeos/16999/** — Altunergil O. Сканирование для обнаружения Rootkit. Перевод Захаровой И.
- **www.softerra.ru/freeos/17032/** — Колисниченко Д. Протоколирование.
- **www.tripwire.org** — сайт разработчиков Tripwire.
- **www.opennet.ru/docs/RUS/netramet/index.html** — Матыцын Д. Сбор статистики по TCP/IP на базе NeTraMet.
- **www.linuxrsp.ru/artic/portsentry.html** — Ерижоков А. А. Portsentry.
- **www.securityfocus.com/infocus/1633** — Alan Neville, Eric Hines with additional research by Joseph Kelly. Footprints in the Sand: Fingerprinting Exploits in System and Application Log Files.
- **linux.ru.net/~inger/RSBAC-DOC-ru.html** — начала RSBAC.
- **netware.nwsoft.ru** — Wobus J. Протокол PPP. Перевод Горохова В.
- **www.dhcp.org** — сервер, полностью посвященный протоколу DHCP.
- **www.isc.org** — Internet Software Consortium разработчик DHCP.
- **www.nominum.com/resources/faqs/dhcp-faq.html** — Nominum's DHCP FAQ.

- **ezine.daemonnews.org/200207/dhcp.html** — Pham L. HOWTO — Setting Up ISC-DHCP 3.*x* Under FreeBSD.
- **www.biblioteka.agava.ru/nastroyka_dns.htm** — Калошин В. Настройка DNS.
- **www.4com.ru/support/DNSAdvanSetup.html** — Холл Э. Тонкая настройка DNS.
- **linux.webclub.ru/bind/pers_dns.html** — Водолазкий В. Мой личный сервер DNS.
- **www.webmin.com** — сайт программы webmin.
- **www.openldap.org** — официальный сайт OpenLDAP.
- **www.opennet.ru/docs/RUS/ldap/index.html** — Захаров М. Руководство по настройке аутентификации пользователей посредством LDAP.
- **www.keldysh.ru/metacomputing/ism99.html** — Валиев М. К, Китаев Е. Л., Слепенков М. И. ИПМ им. М. В. Келдыша РАН. Использование службы директорий LDAP для представления метаинформации в глобальных вычислительных системах.
- **www.opennet.ru/docs/RUS/ldap/index.html** — Захаров М. Руководство по настройке аутентификации пользователей посредством LDAP.
- **www.bog.pp.ru/work/ftpd.html** — описание конфигурирования сервера wu-ftp.
- **ftp.wu-ftpd.org** — исходный текст пакета wu-ftp.
- **www.westnet.com/providers/multi-wu-ftpd.txt** — описание настройки виртуальных FTP-серверов.
- **ftp.fni.com/pub/wu-ftpd/guest-howto** — HOWTO по настройке анонимного доступа на FTP-сервер.
- **malik.bishkek.su/doc/UNIX/innd/inn.htm** — Савин Ю. Сервер новостей InterNetNews (INN).
- **www.bog.pp.ru/work/inn.html** — конфигурирование сервера INN.
- **antonio.mccinet.ru/net/nntp.html** — протокол новостей (NNTP).
- **www.logic.ru/Russian/soft/ligs/node382.html** — система электронных новостей и Usenet.
- **ief.tup.km.ua/docs/Linux/NAG/nag19.html** — описание NNTP.
- **www.isc.org/products/INN** — официальный сайт INN.
- **www.switch.ch/switch/netnews/wg/newstools.html** — утилиты для пакета INN.
- **www.mibsoftware.com/userkt/inn/0346.htm** — утилиты для пакета INN.
- **bog.pp.ru/work/ipchains.html** — Bog BOS: ipchains: фильтрация пакетов в Linux: принципы работы, установка и настройка.

- **gazette.linux.ru.net/rus/articles/iptables-tutorial.html** — Andreasson O. Iptables Tutorial 1.1.19. Перевод Киселева А.
- **www.rfc-editor.org** — сайт, посвященный RFC.
- **msk.nestor.minsk.by/kg/2003/05/kg30503.html** — компьютерная газета Linux в Сети. Сага первая. Подготовил X-Stranger.
- **gazette.linux.ru.net/rus/articles/povest_ob_ip.html** — Песин И. Повесть об IP-адресации.
- **onix.opennet.ru/mail/mail.html** — Бешков А. Почтовая система среднего и малого офиса.
- **www.opennet.ru/docs/RUS/mail/index.html** — Мамаев М. Электронная почта.
- **www.compulenta.ru/dk/offline/2001/64/13214/print.html** — Зиммерманн Ф. Основы криптографии.
- **host.ru/documentation/v-www/0020.html** — использование PGP.
- **www.arh.ru/~zwon/gph/book1.html** — описание GNU Privacy Guard.
- **www.citforum.ru/internet/servers/** — Храмцов П. Организация и администрирование почтовых и файловых серверов Internet. Центр Информационных Технологий.

ПРИЛОЖЕНИЕ 3

Описание компакт-диска

На компакт-диске находится большинство программ, рассмотренных в этой книге. Для стабильных веток программ представлены самые последние версии на момент редактирования книги.

Программы сгруппированы по главам, таким образом, не составит труда найти нужную. Везде, где это возможно, программа представлена в виде TAR-архива.

Каталог	Источник	Описание
Chapter3		
ppp-2.4.1.tar.gz	ftp://cs.anu.edu.au/pub/software/ppp/	Демон pppd, версия 2.4.1
diald-1.0.tar.gz	http://diald.sourceforge.net/	Демон diald, версия 1.0
mgetty+ sendfax-1.0.0.tar.gz	http://www.leo.org/~doering/mgetty/	Программа mgetty, версия 1.0
Chapter4		
dhcp-latest.tar.gz	http://www.isc.org/products/DHCP/	Демон DHCP, версия 3.0pl2
Chapter5		
bind-9.2.3.tar.gz	http://www.isc.org/products/BIND/bind9.html	Демон bind, версия 9.2.3
webmin-1.130.tar.gz	http://www.webmin.com/	Утилита webmin, версия 1.0
usermin-1.060.tar.gz	http://www.webmin.com/	Утилита usermin, версия 1.0

(продолжение)

Каталог	Источник	Описание
Chapter6		
sendmail.8.12.11.tar.gz	http://www.sendmail.org/	MTA sendmail, версия 8.12.11
postfix-2.0.18.tar.gz	ftp://ftp.nl.uu.net/pub/unix/mail/postfix/	MTA postfix, версия 2.0.18
gnupg-1.2.4.tar.bz2	http://www.gnupg.org	Программа gpg, версия 1.2.4
Chapter7		
openldap-2.2.5.tgz	http://www.openldap.org	Программный пакет openldap, версия 2.2.5
Chapter8		
wu-ftpd-current.tar.gz	http://www.wu-ftpd.org/ wu-ftp 2.6.2	FTP-сервер wu-ftp, версия 2.6.2
Chapter9		
inn-2.4.1.tar.gz	http://www.isc.org/products/INN/	Сервер новостей inn, версия 2.4.1
Chapter10		
apache_1.3.29.tar.gz	http://www.apache.org	Apache, версия 1.3.29
httpd-2.0.48.tar.gz	http://www.apache.org	Apache, версия 2.0.48
Chapter11		
squid-2.5.STABLE4.tar.gz	http://www.squid-cache.org	Squid, версия 2.5.4
mrtg-2.10.13.tar.gz	http://people.ee.ethz.ch/~oetiker/webtools/mrtg/	Mrtg, версия 2.10.13
sarg-1.4.1.tar.gz	http://61.19.180.226/big-linux/download/program/	Sarg, версия 1.4.1
Chapter12		
ntp-4.2.0.tar.gz	www.ntp.org	Ntp, версия 4.2.0
Chapter13		
samba-3.0.1.tar.bz2	www.samba.org	Samba, версия 3.0.1
gsmb-0.5.tar.gz	http://www.culte.org/projets/developpement/gsmb/	Gsmb, версия 0.5
SambaSentinel-0.1.tar.gz	http://kling.mine.nu/sambasentinel.htm	SambaSentinel, версия 0.1

(продолжение)

Каталог	Источник	Описание
Chapter15		
mars_nwe-0.99.pl12.tgz	http://www.compu-art.de/mars_nwe/	Mars_nwe, версия 0.99.12
Chapter18		
rrdtool-1.0.46.tar.gz	http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/	Rrdtool, версия 1.0.46
rrdUtils-2.3.tar.gz	http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/	RrdUtils, версия 2.3
Chapter19		
net-acct-0.71-glibc2.tar.gz	http://exorsus.net/projects/net-acct/	net-acct, версия 0.71
netacct-mysql-0.76.tar.gz	http://netacct-mysql.gabrovo.com/	netacct-mysql, версия 0.76
Chapter20		
freeswan-2.04.tar.gz	http://www.freeswan.org/	Freeswan, версия 2.04
pptp-linux-1.4.0.tar.gz	http://pptpclient.sourceforge.net/	pptp-linux, версия 1.4.0
pptp_gui-0.06.1_src.tar.gz	http://pptpclient.sourceforge.net/	pptp_gui, версия 0.06.1
Chapter21		
etherboot-5.2.2.tar.bz2	http://etherboot.sourceforge.net/	Etherboot, версия 5.2.2
etherboot-doc-5.2.2.tar.bz2	http://etherboot.sourceforge.net/	Документация к пакету Etherboot
pxes-base-i586-0.7-1.src.rpm	http://pxes.sourceforge.net/	pxes-base, версия 0.7-1
pxes-ltsp-0.7-1.src.rpm	http://pxes.sourceforge.net/	pxes-ltsp, версия 0.7-1
pxesconfig-0.7-1.src.rpm	http://pxes.sourceforge.net/	Pxesconfig, версия 0.7-1
Chapter22		
openssh-3.7.1p2.tar.gz	http://www.openssh.com/	Openssh, версия 3.7.1
Chapter23		
stunnel-4.04.tar.gz	http://www.stunnel.org/download/	Stunnel, версия 4.04
portsentry-1.2.tar.gz	http://sourceforge.net/projects/sentrytools/	Portsentry, версия 1.2

(окончание)

Каталог	Источник	Описание
tripwire-2.3-47.bin.tar.gz	http://www.tripwire.org	Tripwire, версия 2.3.47
NeTraMet43.tar.gz	http://www.auckland.ac.nz/net/NeTraMet/	NeTraMet, версия 4.3
chkrootkit.tar.gz	http://www.chkrootkit.org	chkrootkit, версия 0.43
lids-1.1.2-2.4.21.tar.gz	http://www.lids.org	lids-1.1.2 для ядра 2.4.21
lids-2.0.3-2.6.0.tar.gz	http://www.lids.org	Lids 2.0.3 для ядра 2.6.0
aide-0.10.tar.gz	http://www.cs.tut.fi/~rammer/aide.html	Aide, версия 0.10
rsbac-admin-v1.2.2.tar.gz	http://www.rsbac.org	Утилиты администрирования Rsbac, версия 1.2.2
rsbac-v1.2.2.tar.gz	http://www.rsbac.org/	Rsbac, версия 1.2.2
nmap-3.50.tgz	http://www.insecure.org/nmap	Nmap, версия 3.50
dsniff-2.3.tar.gz	http://monkey.org/~dugsong/dsniff/	Сниффер Dsniff, версия 2.3

Предметный указатель

/

/etc/bashrc 70
/etc/fstab 70
/etc/host.conf 48
/etc/hosts 47
/etc/initscript 70
/etc/inittab 59
/etc/issue 70
/etc/motd 70
/etc/networks 47
/etc/resolv.conf 48
/etc/skel 70

8

802.2 281
802.3 281

A

AboutTime 252
Access Control Lists 228, 436
Access.log 236
ACL 228
Active Directory 260
Address Resolution Protocol 46
AIDE 460
AnalogX Atomic TimeSync 252
ARP 46

B

BGP 32
Bindery 281
BOOTP 400
BOOTPD 401

C

Cache digest 221
Callback-сервер 90
Canonical NAME 119
Chat 77, 361
Chkrootkit 452
Ckconfig 175
CNAME 119
Common UNIX Printing System 351
Control-panel 66
Ctlinnd 184
CUPS 351

D

Dhclient.conf 106
Dhclient.leases 108
DHCP 97, 400
Dhcpd.conf 101
Dhcpd.leases 104
DHCP-клиент 106
DHCP-сервер 101
Dial on demand 77, 361
Diald 77, 83, 361, 364
Dial-in-сервер 90
DNS 21, 47, 110
Domain Name System 110
Domain Network Service 47
DVMRP 32
Dynamic Host Configuration Protocol 97

E

EGP 31
Etherboot 400
Ethernet_II 281
Extended attributes 436

F

File Transfer Protocol 159
Firewall 299
FTP 22, 159
Ftpaccess 166
Ftpconversions 172
Ftpcount 174
Ftpd 173
Ftpgroups 173
Ftphosts 173
Ftprestart 175

Ftpservers 172
Ftpshut 174
Ftpusers 173
Ftpwho 174

G

Gated 45
GNU Privacy Guard 128
GOSIP 24
GPG 128
GQ 156

H

HDLC 51
High-Level Data Link Control 51
HINFO 119
Host INFOrmation 119

I

ICMP 22
IETF 29
Ifconfig 44
IGP 31
IGRP 31
IMAP 125
Init 57
INN 184
Innd 184
Innwatch 191
Insmod 44
Interactive Mail Access Protocol 125
Internet Protocol Security 389
IP 22
Ipchains 308
IPSec 389
Iptables 308
IPX 281

K

Killproc 70
Kldap 156
Klogd 448
Ksamba 277

L

LCP 51
LDAP 142
LDAP Data Interchange Format 150
Ldapadd 156
Ldapdelete 155
Ldapmodify 156
Ldapsearch 155
LIDS 453
Lightweight Directory Access Protocol 142
Line Printer Daemon 351
Link Control Protocol (LCP, протокол управления соединением) 49
Linux Intrusion Detection/Defence System 453
Linuxconf 66
Loopback 43
LPD 351
Lwared 290

M

Macntp 252
MAC-адрес 46
Magic-filter 354
Mars_nwe 284
Media Access Control 46
Mgetty 88
Microsoft Network Client for MS-DOS 405
Microsoft Point-To-Point Encryption 396
MIME 126
MPPE 396
MRTG 373
Multi Router Traffic Grapher 373
Multipurpose Internet Mail Extension 126

N

NCP 51, 281
NeTraMet 444
Network Control Protocols 50
Network File System 255
Network Information Service 140
Network News Transfer Protocol 178
Network Time Protocol 242
Network Virtual Terminal 416
NFS 22, 255
NIC 22
NIS 140
NIS+ 142
Nnrp.access 184
NNTP 178
Novell Netware 280
NTP 242
Ntp.conf 245
Ntpdate 250
Ntpq 250
Ntptrace 250

Ntsysv 66
NVT 416

O

OpenSSH 421
OSI 21
OSPF 31

P

PGP 128
Point-to-Point Protocol 49
Point-to-Point Tunneling Protocol 389, 394
POP3 124
Portsentry 442, 460
Post Office Protocol 124
Pppd 77, 89, 361
PPTP 389, 394
Pretty Good Privacy 128
Primary Domain Controller 260
Proxy-сервер 220, 300

R

R-команды (remote-команды) 420
Rc 65
Rc.local 70
Rc.sysinit 64
Rcp 420
Rdist 420
Request For Comments 38
Resource Records 117
Responsible Party 120
RFC 22
◊ RFC 1128 245
◊ RFC 1129 245
◊ RFC 1165 245
◊ RFC 1305 245
◊ RFC 2030 245
◊ RFC 2131 97
◊ RFC 2132 97
RIP 22, 31, 281
Rlogin 420
Rootkit 449
Round Robin Database 373
Route 45, 281
Routed 45
RRDtool 373
RSBAC 460
Rsh 420
Rsync 420
Run level 58

S

S/MIME 128
Samba 260, 261
SambaSentinel 277
SAP 282
Satan 441
Scp 432
Secure/Multipurpose Internet Mail Extensions 128
Security-Enhanced Linux 461
Serial Lines Internet Protocol 56
Setserial 76
Sftp 431
Simple Mail Transfer Protocol 124
SLIP 56
Smb.conf 262
Smbclient 277
Smbpasswd 277
Smbstatus 277
Smbtar 277
SMTP 22, 124
SNAP 282
Sniffer 449
SNMP 22
Squid 221
Squid.conf 222
SSH 421
Ssh_config 425
Ssh-add 431
Ssh-agent 431
Sshd_config 422
Ssh-keygen 430
Ssh-keyscan 433
Start Of Authority 117
Store.log 237
Stunnel 439
SWAT 277
Syslogd 446

T

Tc 372
TCP 22
TCP/IP 25
Telinit 63
Telnet 22, 415
TFTP 400
Top Level Domains 110
Traffic shaper 372
Transparent Proxy 234, 371
Tripwire 459
Trivial File Transfer Protocol 400

U

UDP 23
Useragent.log 238

V

Virtual Private Network 389
VPN 389

W

Webmin 277
WinSocket 405
Wu-ftp 164

X

Xferlog 175
Xntpd 245, 251
Xntpdc 251

Б

Бездисковый компьютер 397
Брандмауэр 299
Брандмауэр с фильтрацией 301

В

Виртуальная частная сеть 389

Д

Датаграмма 21
Демилитаризованная зона 329
Демон линейной печати 351
Доменная Система Имен 110
Домены верхнего уровня 110

М

Магический фильтр (magic-filter) 354

О

Общая система печати для UNIX 351

П

Протокол:
◊ ICP 221
◊ PPP 49
◊ динамического конфигурирования хостов 97
◊ передачи файлов 159

Р

Разделение внешнего канала 239
Расширенные атрибуты 436

С

Сервер точного времени 242
Сетевая файловая система 255
Сетевой протокол времени 242
Сигнал SIGKILL 70
Служба сетевой информации 140
Списки контроля доступа 436

У

Уровень выполнения 58