

# Защита сети

- ▶ Модели защиты
- ▶ Правила задания паролей
- ▶ Проблемы со службами, передающими информацию в явном виде
- ▶ Защита терминальной информации (OpenSSH)
- ▶ Защита служб электронной почты (POP3 и IMAP)
- ▶ Защита FTP-сервера
- ▶ Защита сервера Apache
- ▶ Профили защиты системы и защита ядра (securelevel)
- ▶ Использование брандмауэра
- ▶ Предотвращение вторжений и взломов
- ▶ Атаки на службы (DOS)
- ▶ Физическая защита
- ▶ Другие источники информации о защите

Из всех тем, близких сердцу системного администратора, наибольший интерес вызывает защита. Защита — определенно, наиболее важная часть работы системного администратора, независимо от того, управляет ли он сервером Windows, коммерческой UNIX-системой или ОС FreeBSD. Не случайно по теории защиты, практике защиты, взлому защиты, философии защиты и т.д. написано больше книг, чем по любой другой теме администрирования. Это очень сложная тема. Не стоит и надеяться раскрыть ее полностью в одной главе, но защита настолько важна для успешной работы системы в сети, что описание ОС FreeBSD будет явно недостаточным без обсуждения защиты.

Сегодня сеть Internet — не слишком дружественное место для серверов. Изобилие "хакерских" ("rootkit") средств и опубликованных сценариев атак в сочетании с бесчисленными личностями, не находящими ничего лучше, чем деструктивные увлечения, создают атмосферу, в которой администратор должен всегда предполагать худшее и готовиться к нему. Надо иметь в виду, что система испытывается на наличие слабых мест в защите круглые сутки, и с каждым опубликованным методом взлома ситуация еще более ухудшается. Единственная защита — поддержка самой современной версии системы, выполнение рекомендаций по защите и знание реальных опасностей и зон максимального риска.

Угрозы защите сетевого сервера можно разделить на три основные категории.

- **Получение доступа от имени пользователя root.** Атакующий использует информацию, передаваемую в незашифрованном виде, или известные ошибки в программном обеспечении сервера (чаще всего — "переполнения буфера") для получения доступа к системе от имени суперпользователя. Затем он устанавливает собственные программы, скрывающие его присутствие от средств контроля системы (таких как `last` и `ps`), и может похищать важные для системы данные, служащие основой для дальнейших взломов.
- **Нарушение конфиденциальности.** Если информация, передаваемая по сети с вашей системы и на нее, не шифруется (не скрывается), атакующий может просматривать любую ее часть, включая пароли (что позволяет получить доступ от имени пользователя `root`) и конфиденциальные или секретные данные любого пользователя.
- **Сбой служб.** Атакующий использует силовые методы, вроде переполнения сервера большими объемами вполне законной информации, что лишает его возможности обслуживать обычных клиентов и может привести к сбою системы.

Эта глава поможет вам разработать правила защиты для системы FreeBSD, работающей в качестве сервера или рабочей станции, с учетом факторов риска. Реальность подтверждает, что совершенной модели защиты не существует, и только нечеловеческими усилиями можно поддерживать защиту системы на таком уровне, что никакая атака ей не страшна. "Совершенная защита" — миф. Лучшее, что можно сделать, — прикрыть области наибольшего риска благодаря знанию их природы и способов защиты.

## Модели защиты

Есть несколько моделей защиты, которые вы можете принять для своей системы, — в зависимости от условий каждая из них может оказаться наиболее подходящей. Выбранная модель определяет, насколько тщательно придется выполнять определенные административные обязанности, например, внедрять правила использования паролей, открывать доступ к службам, шифровать передаваемую информацию и т.д.

- **"Я доверяю всем в Internet"**. Хотя эту модель практически никогда нельзя рекомендовать, именно ее придерживаются администраторы любительских серверов (и дорого за это платят). Часто встречающиеся в университетской среде, особенно среди эксплуатируемых многие годы (с того времени, когда сеть Internet еще не была переполнена взломщиками), системы этого типа предлагают много открытых служб, не требуют шифрования при удаленном доступе, имеют нестрогие правила задания паролей и ведения учетных записей, являясь тем самым привлекательными целями для атак взломщиков.
- **"Я доверяю всем в локальной сети"**. Типичная для сетей предприятий эта модель обычно используется, когда сервер защищен от остальной части Internet брандмауэром, а внутренняя сеть используется исключительно сотрудниками. Враждебно настроенные пользователи во внутренней сети встречаются редко, поэтому можно себе позволить не шифровать передаваемую информацию и даже отключать защиту учетных записей. Если брандмауэр надежен, сервер в такой среде может иметь самую слабую защиту и вполне приемлемо работать.
- **"Я доверяю локальным пользователям"**. Несколько более параноидальная, чем две предыдущие, эта модель диктует строгие правила сетевой защиты: пользователи не принимаются, пока для них не будут сделаны новые учетные записи, передаваемая сетевыми службами информация шифруется (шифрование либо требуется, либо настоятельно рекомендуется), ненужные службы отключаются, и используются устойчивые к взлому пароли. Однако локальным пользователям разрешается доступ к внутренним службам и просмотр конфиденциальной информации (например, зашифрованных паролей). Идея в том, что если уж пользователям выделены учетные записи, они могут использовать систему в полном объеме, а если им нельзя доверять в такой степени, то это является основанием для удаления из системы. Эта модель подходит для любительских систем, обслуживающих пользователей из группы "низкого риска" (например, Web-сайт фанатов футбольного клуба или служба электронной почты для жильцов дома), или для высокоуровневых коммерческих серверов Internet, где лишь немногие пользователи имеют учетные записи.
- **"Я доверяю только себе и другим администраторам"**. Наиболее параноидальная из всех эта модель не только предполагает такую же серьезную сетевую защиту, как и предыдущая, но и строгую защиту для локальных пользователей. Обычным пользователям запрещен доступ к файлам конфигурации системы и серверным программам благодаря специально сконфигурированным правам доступа. Администратор должен внимательно следить за каждым пользователем, чтобы гарантировать отсутствие неавторизованных действий, причем часто принимаются специальные меры (используются специализированные командные интерпретаторы, изменяется начальный каталог с помощью команды `chroot` и отключаются определенные команды), ограничивающие доступ пользователя к ресурсам системы. Эта модель подходит для высокоуровневых серверов, поддерживающих службы электронной почты или Web-хостинга для сотен или тысяч пользователей с неопределяемыми или неизвестными целями.

Выбрав подходящую модель защиты сети и пользователей, необходимо решить, какие области риска предполагаются этой моделью и что можно сделать, чтобы уменьшить вероятность взлома в этих областях. Наиболее типичными областями риска являются:

- небезопасные (слабые) пароли;
- службы, пересылающие информацию в явном (незашифрованном) виде;
- ненужные и легко взламываемые службы;
- открытая пересылка сообщений по протоколу SMTP;
- неограниченный сетевой доступ;
- устаревшее или уязвимое программное обеспечение.

Каждое из этих узких мест — потенциальная проблема в ОС FreeBSD в стандартной конфигурации. В этой главе описано, как решать каждую из них и представлены необходимые средства поддержки системы, предотвращающие взломы.

Открытая пересылка сообщений по протоколу SMTP скорее мешает быть добропорядочным членом сообщества систем Internet, чем является проблемой защиты. Ее детальное обсуждение представлено в главе 25.

## Правила задания паролей

Если пользователи имеют небезопасные пароли, все остальные принимаемые меры защиты могут оказаться неэффективными. Вероятно, наиболее ответственная задача администратора системы FreeBSD — установить правила задания паролей, требующие от пользователей применять пароли, которые сложно подобрать или взломать (или хотя бы побуждающие их к этому).

Многим пользователям необходимость ввода паролей кажется неудобством, а использование строгих правил задания паролей — двойным неудобством. Неконтролируемый пользователь будет задавать в качестве пароля свое регистрационное имя, номер телефона, имя хоста, слово типа "password" или строку символов, которую удобно набирать, например строку из повторяющихся букв или цифр. Если пароль устареет, первое, что попытается сделать пользователь в ответ на запрос нового пароля, — ввести старый пароль. Одна из аксиом защиты, однако, гласит, что "удобство и безопасность — взаимоисключающи", т.е. повышать одно можно только за счет снижения другого. Большое удобство означает меньшую безопасность. Этот факт обойти трудно.

При выборе пользователем пароля с помощью стандартной программы `passwd` или сценария, вызывающего те же подпрограммы, что и `passwd`, выполняется несколько несложных проверок. По умолчанию пароли должны состоять не менее чем из шести символов, но это, похоже, единственная встроенная мера против выбора пользователями слабых паролей. Давайте рассмотрим несколько методов, позволяющих существенно повысить уровень защиты при выборе паролей.

## Обеспечение безопасности паролей с помощью программы Crack

Идеальный пароль состоит не менее чем из восьми символов (чем длиннее, тем лучше) и содержит прописные и строчные буквы, цифры и знаки препинания или "метасимволы". Необходимо с помощью программы `passwd` проверять, все ли пользователи следуют этим принципам при задании пароля.

Напомню, что единственное ограничение, налагаемое стандартной программой `passwd`, — длина пароля должна быть не менее шести символов. Сообщество разработчиков FreeBSD прилагаются усилия по внедрению в программу `passwd` дополнительных проверок слабых паролей, прежде всего запрещающих пользователям выбирать небезопасные пароли. На момент написания этой главы, однако, лучший способ убедиться, что пользователи не используют легко подбираемые пароли, — периодически пытаться подобрать их самому. Это делается с помощью программы `Crack`, находящейся в наборе портированных приложений в каталоге `/usr/ports/security/crack`. Хотя эта программа может показаться "хакерской", она предназначена для контроля защиты системными администраторами. Программа `Crack` позволяет выполнять "атаки по словарю" (попытки подобрать в качестве пароля одно из английских слов), а также выявлять ряд других "удобных" паролей: повторяющиеся строки, регистрационное имя пользователя, группы цифр и т.д. Цель — показать, кто из пользователей применяет небезопасные пароли, чтобы можно было связаться с ними непосредственно и потребовать соблюдения установленных правил задания паролей.

После создания и установки портированной программы `Crack` (подробнее об установке дополнительного программного обеспечения см. в главе 15), появляется новый каталог, `/usr/local/crack`, с правами доступа, позволяющими просматривать содержимое или запускать находящиеся в нем программы только пользователю `root`. Для того чтобы проверить наличие слабых паролей в базе данных пользователей системы, перейдите в каталог `/usr/local/crack` и запустите программу `Crack` следующим образом (обратите внимание, что первая буква — прописная):

```
# ./Crack -fiat bad /etc/master.passwd
```

Программа `Crack` создаст ряд утилит и скомпилирует несколько словарей. Затем она применит свои проверки к файлу `/etc/master.passwd`, посылая результаты в рабочие файлы, которые можно анализировать с помощью программы `Reporter` следующим образом

```
./Reporter -quiet
-----passwords cracked as of Sun Jan 14 12:17:41 EST 2001 -----
979693112:Guessed frank [frank] Frank Jones [/etc/master.passwd /bin/tcsh]
979693187:Guessed joe [password] Joe User [/etc/master.passwd /usr/local/
'••bin/bash]
-----done -----
```

Будет сообщено только о пользователях, пароли которых были подобраны. В представленном выше примере взломанные пароли показаны в первых квадратных скобках. Пароль пользователя Frank — `frank`, а пароль пользователя Joe — `password`. Оба эти пароля — крайне слабые, так как без особых усилий могут быть угаданы атакующим. Теперь можно связаться с этими пользователями и напомнить им о правилах задания паролей, потребовав изменить пароли на более безопасные.

После завершения работы программы `Crack`, удалите утилиты времени выполнения и файлы результатов с помощью следующих двух команд:

```
# make tidy
# rm run/F-merged
```

#### ПРИМЕЧАНИЕ

Поскольку ОС FreeBSD продолжает развиваться, весьма вероятно, что в программу `passwd` будет включена поддержка автоматической проверки сложности паролей, аналогичная выполня-

емой вручную с помощью программы Crack. Библиотеки, используемые программой Crack, доступны в каталоге `/usr/ports/security/cracklib`. В системах типа Linux на основе библиотеки `cracklib` был разработан встраиваемый модуль аутентификации (pluggable authentication module — PAM) — механизм, поддерживаемый также и в ОС FreeBSD (см. страницу справочного руководства `man pam`), но на момент написания этой главы ОС FreeBSD не поддерживает полную интеграцию `cracklib` в систему модулей PAM. Сейчас же авантюристы, желающие поработать с исходным кодом и экспериментальным программным обеспечением и заинтересованные во включении возможностей библиотеки `cracklib` в программу `passwd`, могут узнать, как это можно сделать, по адресу <http://www.kearneys.ca/~brent/FreeBSD/passwd42.html>.

## Устаревание паролей

По умолчанию пароли в ОС FreeBSD не устаревают. Однако одним из стандартных правил обеспечения безопасности паролей является требование частой их смены, с заданным администратором сроком действия.

Чтобы сделать это в ОС FreeBSD, необходимо изменить файл `/etc/login.conf`. Этот файл обеспечивает централизованный способ управления возможностями и поведением системы (например, допустимым количеством одновременных процессов, максимально допустимым размером процесса, максимальным количеством одновременно открытых файлов, особенностями работы командных интерпретаторов и многими другими параметрами, описанными на странице справочного руководства `man login.conf`). Каждое из этих свойств может быть привязано к "классу" пользователей, который можно задавать с помощью команды `chfn` (как было описано ранее). Обычно пользователи не связаны с конкретным классом, поэтому значения стандартного класса `default` применяются для всех:

```
defaultл
:passwd_format=md5:\
:copyright=/etc/COPYRIGHT:\
:welcome=/etc/motd:\
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K,FTP_PASSIVE_MODE=YES:\
:path=/sbin /bin /usr/sbin /usr/bin /usr/games /usr/local/
^*sbin /usr/local/bin /usr/X11R6/bin ~/bin:\
:nologin=/var/run/nologin:\
:eputime=unlimited:\
:datasize=unlimited:\
:stacksize=unlimited:\
:memorylocked=unlimited:\
:memoryuse=unlimited:\
:filesize=unlimited:\
:coredumpsize=unlimited:\
:openfiles=unlimited:\
:maxproc=unlimited:\
:sbsize=unlimited:\
:priority=0:\
:ignoretimee:\
:umask=022:
```

Символы обратной косой (\) "маскируют" переводы строк, позволяя задавать все эти свойства по одному в строке, что упрощает понимание файла.

Для установки даты устаревания пароля необходимо добавить в класс `default` дополнительную строку, задающую свойство `passwordtime`. Ее можно включить в блок в любом месте, но проще всего — сверху, сразу после имени класса:

```
default:\
:passwordtime=90d:\ •
:passwd_format=md5:\
:copyright=/etc/COEYRIGHT:\
:welcome=/etc/niotd:\
```

В этом примере устанавливается устаревание паролей через 90 дней. Можно использовать и другие форматы задания времени, например 2y (2 года), 6w (6 недель) или 24h (24 часа). Теперь, поскольку файл `/etc/login.conf` представляет собой базу данных, которую необходимо скомпилировать в хеш-таблицу (как и таблицы в каталоге `/etc/mail`, рассматривавшиеся в главе 25), надо вызвать программу `cap_mkdb` для генерации хеш-таблицы с учетом изменений:

```
# cap_mkdb /etc/login.conf
```

С этого момента, если со времени последнего изменения пароля пользователем прошло более 90 дней, процедура регистрации потребует от пользователя выбрать новый пароль. Учтите, что если свойство `passwordtime` установлено, программа `passwd` записывает время последнего изменения пароля в шестое поле файла `/etc/master.passwd`:

```
frank:$1$IJC2kCuzD$70a8LyRf5jTOb.Xrx1B3d.:1060:100::999066364:O:~/home/frank:/bin/tcsh
```



### СОВЕТ

Можно использовать файл `login.conf` и для изменения стандартной минимальной длины пароля. Это делается путем задания значения свойства `minpasswordlen`:

```
:minpasswordlen=8:\
```

Тем самым минимальная длина пароля устанавливается равной 8 символам.

## Задание первоначальных паролей

Кажется удобным устанавливать простой начальный пароль для каждого нового пользователя, например `Temp 123` или `ChangeThis`. Однако это несет серьезную угрозу защите, особенно если для всех вновь добавляемых пользователей задается один и тот же пароль.

Эту угрозу можно существенно уменьшить, генерируя для каждого пользователя пароль случайным образом. При этом можно использовать любую схему генерации паролей (например, по первым буквам слов в названиях песен), но выполнение таких действий вручную вам быстро надоест. Есть хороший способ генерировать уникальные и неподбираемые пароли — с помощью программы `md5` и нескольких случайных нажатий на клавиши:

```
# md5 -s "asdsad"
MD5 ("asdsad") = b5b037a78522671b89a2c1b21d9b80c6
```

Затем можно взять первые семь или восемь символов из этой строки (например, `B5b037a7`) в качестве пароля нового пользователя, проинструктиввав пользователя о том, как с помощью программы `passwd` заменить пароль другим, проще запоминаемым. Имеет смысл реализовать эту схему в виде небольшого сценария на языке Perl, выполняющего для генерации нового пароля хеширование с помощью алгоритма MD5 результата выполнения функции `rand()`.

## Одноразовые пароли на базе системы S/Key

Если вы действительно серьезно озабочены безопасностью паролей, можете сделать то же, что и правительственные структуры и другие организации, использующие максимальную защиту: назначать пользователям одноразовые пароли, перенося часть бремени обеспечения защиты со своих плеч на плечи пользователей. Одноразовые пароли генерируются ключевой программой (*key*), разновидности которой существуют для всех основных платформ — есть даже не зависящая от платформы реализация на языке Java, см. <http://www.cs.umd.edu/~harty/jotp/src.htiiil>. (В ОС FreeBSD для вычисления ключей используется алгоритм MD4.)

Одноразовые пароли хорошо подходят для систем, в которых пользователи не обязаны использовать систему SSH вместо Telnet (систему SSH мы рассмотрим далее). Поскольку новый пароль должен генерироваться пользователем с помощью ключевой программы на локальной системе, которой передается выданная сервером фраза "запроса" ("challenge" phrase) и собственный секретный пароль пользователя, никогда не пересылаемый по сети (за исключением процедуры начальной установки ключа), перехват проходящих по сети пакетов никогда не даст никакой полезной информации. Однажды использованный пароль нельзя использовать повторно. Пользователь может передавать одноразовый пароль в виде обычного текста.

### ПРИМЕЧАНИЕ

Система S/Key не менее ценное средство для озабоченного защитой пользователя, чем требование соблюдать принципы защиты для администратора. Многие части системы S/Key, например программа *keyinit*, должны поддерживаться пользователем. Если пользователь хочет обеспечить конфиденциальность паролей, он может использовать одноразовые пароли, даже если в этом нет особой необходимости.

Предположим, необходимо сделать так, чтобы пользователь Frank не мог зарегистрироваться с удаленного хоста с обычным паролем UNIX, а был вынужден использовать одноразовые пароли S/Key. Зарегистрировавшись на сервере (предпочтительно по безопасному каналу, например, с помощью SSH), он должен воспользоваться программой *keyinit* для настройки аутентификации системы S/Key:

```
# keyinit
Adding frank:
Reminder-Only use this method if you are directly connected.
If you are using telnet or rlogin exit with no password and use
^•keyinit -s.
Enter secret password:
Again secret password:

ID frank s/key is 99 st28077
COL APT HELM TAB DRY TRIM
```

### ПРИМЕЧАНИЕ

Если пользователь Frank подключен по небезопасному каналу (например, при простом соединении с помощью Telnet), он должен использовать вызов *keyinit -e*. При указании опции **-v** программа *keyinit* использует генерацию ключа по ходу работы. Пользователь Frank вводит свой пароль для генерации секретного ключа (который используется исключительно для вычисления одноразовых паролей системы S/Key и не должен совпадать с его паролем в UNIX) и передает его по сети на сервер. Если подключение выполняется по небезопасному каналу, секретный пароль можно перехватить, что делает все дальнейшие действия по защите бесполезными. Опция **-s** требует от пользователя Frank использовать программу *key* локально, на его

собственной машине с ОС Windows, Macintosh или UNIX для генерации пароля, который затем надо будет ввести в программе **keyinit** в ответ на приглашение **s/key access password:** [которое выдается только при указании опции - a) .

После того как пользователь Frank применил программу keyinit для настройки своего механизма S/Key, в результате чего для соответствующего регистрационного имени появилась запись в файле /etc/skeykeys, необходимо создать файл /etc/skey.access (если он еще не существует) и добавить в него следующую строку:

```
deny user frank
```

Файл /etc/skey.access сообщает ОС FreeBSD, при каких условиях удаленный пользователь может использовать обычный пароль UNIX, а когда обязан использовать одноразовый пароль системы S/Key. Каждая строка в файле skey.access задает правило, начинающееся с ключевого слова permit (разрешающего регистрацию как с паролем S/Key, так и с паролем UNIX) или deny (требующего ввода пароля системы S/Key), за которым следует любое количество условий. Эти условия, полностью описанные на странице справочного руководства man skey.access, могут задавать конкретных пользователей, группы, имена удаленных хостов или сетей или терминалы, с которых выполняется регистрация. Строка в представленном выше примере указывает, что пользователь Frank при попытке регистрации может вводить только пароль системы S/Key, а не обычный пароль UNIX.

При следующей попытке пользователя Frank подключиться к системе с помощью Telnet приглашение регистрации для него будет таким:

```
# telnet stripes.somewhere.com
Trying 64.41.131.102...
Connected to stripes.somewhere.com.
Escape character is '^*]'.

FreeBSD/1386 (stripes.somewhere.com) (ttyp2)

login: frank
s/key 99 et28077
Password:
```

Теперь пользователь Frank должен вызвать программу key (или ее эквивалент) на локальной машине для получения необходимого пароля. Программе надо передать информацию запроса, представленную сервером: номер итерации (99 в нашем случае; это означает, что осталось 99 регистраций, прежде чем придется запускать программу keyinit снова) и строку "затравки" (в нашем примере, st28077 — ту же, что и в примере вызова программы keyinit). Эти числа в сочетании с секретным паролем пользователя Frank используются для генерации пароля S/Key, состоящего из шести коротких английских слов в верхнем регистре:

```
# key 99 st28077
Reminder - Do not use this program while logged in via telnet or rlogin.
Enter secret password:
COL APT HELM TAB DRY TRIM
```

Теперь пользователь Frank может ввести эту строку в качестве пароля, и ему будет предоставлен доступ. Если атакующий перехватит эти слова, ему это ничего не даст — они передаются по сети только один раз (в ответ на приглашение регистрации). При следующем запросе системы у пользователя Frank пароля S/Key номер итерации будет 98, и пароль будет уже другим. Номер итерации уменьшается, пока не станет

нулевым. В этом случае пользователь Frank должен запустить программу keyinit снова, чтобы номер итерации опять стал равен 99, в противном случае доступа он не получит, и придется вмешаться администратору. (Например, чтобы изменить для пользователя правило в файле /etc/skey.access, заменив ключевое слово deny на permit пока пользователь не переинициализирует систему S/Key.)

При желании пользователь Frank может сгенерировать сразу несколько ключей, вызвав программу key с опцией -p. Затем он может распечатать эти ключи и взять их с собой, если это покажется ему более удобным или безопасным:

```
# key -p 5 50 st28077
Reminder - Do not use this program while logged in via telnet or rlogin.
Enter secret password:
41 SHOT YOU BIEN GIN JUDD AS
42 CHOW AVIS DOES EMIT FLAM WORK
43 DOCK ATE ANN WAS JOCK OAT
44 WALE AWL ELK LETS AWK WALE
45 GIFT BERT ROD GRIN YANG EAST
```

Запрос пароля S/Key также выдается при вызове команды su, что помогает вообще предотвратить передачу фактического пароля пользователя root по сети.

Чтобы отключить поддержку паролей S/Key для пользователя, удалите запись этого пользователя из файла /etc/skeykeys, а также любое упоминание о нем из файла /etc/skey.access.

## Система Kerberos

Еще один достойный упоминания метод аутентификации — система Kerberos. Разрабатывавшаяся как централизованная система управления регистрацией для кластера Athena в МТИ система Kerberos обеспечивает аутентификацию пользователей центральным сервером сети с выдачей "мандатов" ("tickets") для работы со службами Telnet, FTP, POP3 и NFS без необходимости регистрироваться каждый раз. Если хосты в сети, между которыми передается информация, поддерживают систему Kerberos и подписаны на главном сервере, все проблемы аутентификации берет на себя система Kerberos. ОС FreeBSD позволяет настроить главный сервер Kerberos, на услуги которого подписываются все остальные хосты сети, или просто поддерживать работу с системой Kerberos в сети, где она уже работает.

До недавнего времени система Kerberos была действительно полезна только в коммерческой среде или в самом МТИ. Она упрощает выполнение типичных задач в больших локальных сетях, типичных для университетов, или иерархических сетях предприятий. Помимо этих, остается не так уж много ситуаций, когда система Kerberos действительно эффективна с учетом сегодняшних сетевых тенденций. Это не столько мера защиты, сколько способ избавиться от ненужных действий в средах, где используется исключительно ОС UNIX. Однако поскольку многие сети предприятий используют систему Kerberos для обеспечения централизованных, шифрованных служб регистрации и поскольку модель защиты Windows 2000 тесно связана с системой Kerberos (хотя и с модифицированной ее версией), она снова становится важной и широко распространенной. Может потребоваться интегрировать с ней машину с ОС FreeBSD.

Если необходимо обеспечить поддержку службы Kerberos в ОС FreeBSD, это можно сделать. При добавлении строки kerberos\_server\_enable="YES" в файл /etc/rc.conf

будет запускаться служба, управляющая главным сервером Kerberos. Если же необходимо работать с уже существующим главным сервером, можно убрать символ комментария с соответствующих строк в файле `/etc/pam.conf` для включения централизованной аутентификации служб, которые ее поддерживают, например:

```
login auth sufficient pamjeerberosrv.so try_first_pass
```

Дополнительную информацию по системе kerberos можно найти на странице справочного руководства `man kerberos` и в оперативном руководстве по ОС FreeBSD (<http://www.freebsd.org/handbook>).

## Проблемы со службами, передающими информацию в явном виде

Может показаться, что передача информации с клиентской машины на сервер (в приложениях типа Telnet, электронной почты и HTTP) вполне безопасна — пароли ваши скрыты (по крайней мере их не видно), и вся информация передается небольшими пакетами, проходящими по сети вместе с миллионами других пакетов, так что мысль, будто кто-то прослушивает ваш канал передачи, кажется смешной. Это можно делать только из того же сегмента локальной сети или из локальной сети на другом конце соединения, или в сети одного из недалековидных поставщиков услуг Internet по ходу передачи, причем маскировка и оборудование для этого должны быть лучше, чем у Джеймса Бонда. И даже если кому-то удастся организовать "прослушивание", придется следить за множеством подобных сеансов, прежде чем удастся обнаружить что-нибудь полезное. Ну и кому настолько нечего делать, чтобы заниматься такими вещами?

Да кому угодно. Безопасность служб, передающих информацию в явном виде (приложений, не шифрующих или не скрывающих данные транзакций), — миф. Что еще важнее, принцип "безопасность вследствие неочевидности" ("security through obscurity") — в данном случае представление о том, что каналы связи или службы безопасны, поскольку вам непонятно, кому они могут быть интересны, — не работает. Ложность этой аксиомы была многократно доказана за последние годы. Не поддавайтесь искушению игнорировать защиту служб, передающих информацию в явном виде, в угоду слепой вере, что никому не придет в голову взламывать вашу систему. Всегда надо предполагать наихудшее, например, что вся передаваемая вами информация постоянно перехватывается с враждебными целями.

## Использование утилиты `tcpdump` для контроля передаваемой информации

Продемонстрировать существенный риск использования в сети служб, передающих информацию в явном виде, можно с помощью *анализатора протокола* — программы, следящей за всеми пакетами в сегменте сети и выдающей те из них, которые представляют для вас интерес. Анализатор протокола переводит карту Ethernet в "неразборчивый" режим ("promiscuous mode"), в котором она принимает все пакеты, проходящие по сети, а не отвергает пакеты, адресованные другому интерфейсу (как описывалось в главе 22). Затем он применяет к просматриваемой информации разнообразные фильтры с широкими возможностями конфигурирования. Именно так злоумышленник может извлечь только нужные пакеты из потока информации и перехватить сетевые транзакции.

Встроенный анализатор протокола ОС FreeBSD — программа `tcpdump`, находящаяся в каталоге `/usr/sbin`. Его использование для обычных пользователей запрещено — они получают сообщение об ошибке прав доступа (`Permission denied`) к устройству `/dev/bpf0` (фильтр пакетов Беркли). Эту программу может использовать только администратор (`root`). Программа `tcpdump` предназначена не столько для регистрации реальных данных в просматриваемых пакетах TCP/IP (для этих целей имеет смысл использовать программу `EtherPeek` от `WildPackcls`), а скорее, как и система `Snack`, для проверки защиты — оценки объема информации, передаваемой в систему и из системы в незашифрованном виде, и отслеживания подозрительных действий в сети, возможно, связанных с враждебными намерениями.

#### ПРИМЕЧАНИЕ

Программу `tcpdump` можно использовать для скрытого слежения за действиями пользователей, независимо от их направленности. Предупреждение в файле конфигурации ядра `GENERIC` напоминает о необходимости учитывать этические аспекты, использования анализатора протокола. Все, естественно, зависит от того, в какого рода системе вы работаете, но использование анализаторов протокола аналогично прослушиванию телефонных переговоров или использованию скрытых видеокамер слежения. Поэтому использовать программу `tcpdump` стоит только в тех случаях, когда вас не смущало бы использование для сбора информации средств прослушивания и скрытых видеокамер.

Файл конфигурации программы `tcpdump` — весьма сложный и позволяет делать множество полезных и разнообразных вещей. Пока, однако, нас интересует только возможность продемонстрировать небезопасность передачи информации по протоколу TCP/IP в явном виде. Давайте настроим простой контролирующий фильтр для порта `Telnet` (TCP-порт 23):

```

# tcpdump -x port 23
tcpdump: listening on fxp0
20:14:19.076941 w044.2064002043.sjc-ca.dsl.enc.net.54109 >
w012.Z064002043.sjc-ca.dsl.cnc.net.telnet: S 1972342903:1972342903(0)
win 32768 <mss 1460,nop,wscale 0,nop,nop,timestamp 465710 0> (DF)
[tos 0x10]
                4510 003c e44b 4000 4006 8024 4002 2b2c
                4002 2b0c d35d 0017 758f 9077 0000 0000
                a002 8000 001b 0000 0204 05b4 0103 0300
                0101 080a 0007 1b2e 0000 0000
20:14:19.077050 w012.z064002043.sjc-
ca.dsl.cnc.net.telnet > w044.s064002043.sjc-ca.dsl.cnc.net.54109: S
1734674412:1734674412(0) ack 1972342904 win 17520 <mss 1460> (DF)
                4500 002c c9c2 4000 4006 9acd 4002 2b0c
                4002 2b2c 0017 d35d 6765 07ec 758f 9078
                6012 4470 349c 0000 0204 05b4
20:14:19.677472 w044.2064002043.sjc-ca.dsl.cnc.net.54109 >
w012.Z064002043.sjc-ca.dsl.cnc.net.telnet: ack 195 win 33580 (DF)
(tos 0x10)
                4510 0028 e458 4000 4006 802b 4002 2b2c
                4002 2b0c d35d 0017 758f 910a 6765 08af
                5010 832c 0c49 0000 5555 5555 5555
л
с
134 packets received by filter
0 packets dropped by kernel

```

Поскольку программа `tcpdump` не создавалась для "подслушивания", результат (при указании опции `-x`) выдается в шестнадцатеричном формате, который можно преоб-

разовать в обычный, понятный человеку текст, с помощью соответствующего редактора. Эти данные никак не зашифрованы. Требуются минимальные усилия, чтобы получить полное содержимое пакетов, посылаемых с использованием протокола Telnet. Еще проще это сделать с помощью средств, автоматически декодирующих пакеты, например с помощью программы EtherPeek. Если эти пакеты содержат регистрационную информацию пользователя, атакующий с помощью программы `terdump` или EtherPeek сможет ее узнать и зарегистрироваться в вашей системе. Это касается и протоколов POP3, IMAP, FTP и HTTP, а также почти всех небольших сетевых служб вроде Finger и Syslog. Особенно важно это учитывать при передаче паролей или любых других конфиденциальных данных. Проблема эта связана с обеспечением конфиденциальности не меньше, чем с обеспечением защиты.

К счастью, есть способ решить эти проблемы. Существуют шифрованные варианты реализации основных протоколов передачи данных, многие из которых входят в стандартную установку ОС FreeBSD. Надо только знать, что они существуют, внедрить их и убедить пользователей в необходимости их применять.

## Защита терминальной информации (OpenSSH)

Передача терминальной информации, обычно выполняемая приложениями Telnet или `rlogin`, вероятно, наиболее рискованный тип передачи информации в явном виде, но и самый простой для защиты. В состав ОС FreeBSD входит полный пакет SSH (Secure Shell — защищенный командный интерпретатор), разработанный для замены служб Telnet и `rlogin`, и позволяющий пользователям установить полностью зашифрованный канал обмена информацией с сервером, защищающий пароли регистрации и операции командной строки от попыток перехвата. Речь идет о пакете OpenSSH, первоначально разработанном для ОС OpenBSD, а теперь входящем в состав ОС FreeBSD.

Служба SSH работает на порту 22 как отдельный демон, порождающий новые процессы `sshd` (аналогично серверу Apache) для новых подключений. Чтобы включить поддержку сервера SSH, добавьте следующую строку в файл `/etc/rc.conf` (если ее там еще нет) и перезагрузите систему (или просто введите команду `sshd`):

```
sshd_enable="YES"
```

Клиент SSH заменяет программу Telnet. Нужно просто использовать команду `ssh` вместо `telnet` в командной строке:

```
# s sh stripes.somewhere.com
```

Программа Ssh сама запросит пароль, предполагая, что имя пользователя удаленной машины совпадает с именем локального. Другое имя пользователя можно задать несколькими способами:

```
# ssh stripes.somewhere.com -l frank
# ssh frank@stripes.somewhere.com
```

Программа `ssh` устанавливает шифруемое подключение и передает регистрационную информацию на сервер безопасным образом. С этого момента она работает как обычная реализация Telnet, — с точки зрения пользователя нет никакой разницы. Именно так пользователь операционных систем FreeBSD, Linux, UNIX или Mac OS X должен подключаться к вашей системе.

Пользователям настольных клиентских операционных систем типа Windows или классической Mac OS придется еще немного потрудиться. На этих платформах нет клиентских программ SSH для командной строки, но есть ряд отличных графических программ эмуляции терминала, включающих возможности как Telnet, так и SSH. У пользователей Windows есть программы SecureCRT (от Van Dyke, <http://www.vandyke.com>) и SSH (от SSH Communications Security, <http://www.ssh.com>), а пользователи Mac могут использовать программы NiftyTelnet/SSH или MacSSH. Программы для Windows — обычно коммерческие, а программы для Mac чаще распространяются свободно.

Необходимо убедить пользователей перейти на использование протокола SSH вместо Telnet. Можно сообщить им, опубликовав правила использования серверов, что, учитывая необходимость защиты, рекомендуется использовать протокол SSH. Однако это не гарантирует, что они будут его использовать, и администратор по-прежнему должен учитывать угрозу перехвата атакующим подключения одного из пользователей, решившего протокол SSH не использовать. Более трудоемкий, но результативный подход состоит в полном отключении службы Telnet и требовании использовать вместо нее службу SSH. Чтобы отключить службу Telnet, прокомментируйте строку `teinetd` в файле `/etc/inetd.conf`:

```
#telnet stream tcp nowait root /usr/libexec/telnetd teinetd
```

Затем перезапустите процесс `inetd`:

```
# killall -HUP inetd
```

### **О** ПРИМЕЧАНИЕ *ШЯШШШьШвшшя*

Протокол SSH имеет две популярные разновидности: SSH1 и SSH2. Операционная система FreeBSD поддерживает обе, но протокол SSH1 хуже продуман и потенциальная вероятность выявления в нем уязвимых мест защиты выше, чем в SSH2. Можно отключить поддержку SSH1, добавив следующую строку в файл `/etc/ssh/ssh_config`:

```
Protocol 2
```

Учтите, однако, что не все клиентские реализации SSH в полном объеме поддерживают протокол SSH2. Не отключайте протокол SSH1, если не уверены, что это можно делать.

## Защита служб электронной почты (POP3 и IMAP)

Вероятно, еще более рискованной с точки зрения похищения паролей, чем использование службы Telnet, является поддержка текстовых протоколов POP3 и IMAP (причем их и защитить сложнее). Если почтовые клиенты пользователей настроены для подключения к серверу, например, раз в пять минут, для проверки наличия новых сообщений, легко перехватываемые передачи регистрационного имени и пароля происходят при каждом таком подключении, что повышает вероятность перехвата пароля (это связано с предсказуемой периодичностью отправки этими службами конфиденциальных данных). Если вы внедрили на сервере протокол SSH вместо Telnet, в ваших же интересах обеспечить защиту такого же уровня для служб электронной почты.

В главе 25 описывалась защита программы `qpopper` путем использования встроенных в ОС FreeBSD средств SSL (Secure Sockets Layer — уровень защищенных сокетов) для шифрования подключений по протоколу POP3. Такой же метод можно при-

менить и для протокола ШАР с помощью пакета программ IMAP-UW. Включить поддержку SSL в системе IMAP-UW можно путем получения сертификата в бюро сертификации, как было описано в главе 25. Если у сайта уже есть сертификат, полученный для другой службы (например, qpopper), можно использовать его и для системы IMAP-UW. Подробнее об этом см. в документации по системе IMAP-UW по адресу <http://www.washington.edu/imap/>.

Альтернативный способ шифровать протоколы POP3 и IMAP, централизованно управляя сертификатами SSL и не опираясь на встроенную поддержку SSL каждой из служб, — использование программы stunnel. Эта программа доступна в наборе портированных приложений (в каталоге /usr/ports/security/stunnel) и позволяет настроить универсальный тоннель SSL для любой службы в системе. Чаше всего, ее используют для шифрования информации, передаваемой по протоколам POP3 и IMAP. Если установить эту программу из набора портированных приложений, стандартный сценарий запуска (/usr/local/etc/rc.d/stunnel.sh.sample) запускает процесс прослушивания на портах 993 (для протокола ШАР) и 995 (для протокола POP3), которые являются общепринятыми портами для защищенных версий соответствующих протоколов, как указано в файле /etc/services.

Не забудьте переименовать файл **stunnel.sh.sample** в **stunnel.sh**, как было описано в главе 11. Суффикс **.sample** задается, чтобы администратор обязательно просмотрел содержимое сценария и убедился в корректности путей к файлам сертификатов (с расширением **.pem**).

При использовании программы stunnel вес равно необходимо получить сертификат, как и в случае с qpopper и IMAP-UW. Сертификат для программы stunnel надо поместить в файл /usr/local/etc/stunnel.pem. После этого ваши клиенты протоколов POP3 и ШАР должны получить возможность подключаться к соответствующим портам (993 вместо 143 для ШАР и 995 вместо ПО для POP3) для установки безопасного подключения.

Проблема в том, что не все клиенты электронной почты поддерживают шифрование на базе SSL для протоколов POP3 и ШАР. Некоторые популярные клиенты, например Microsoft Outlook, поддерживают, другие — не поддерживают или поддерживают шифрование только одного из протоколов или эта поддержка неполна и необязательна. Требование использовать шифрование на базе SSL может означать требование сменить программу работы с электронной почтой, а это многим не нравится. Учтите также, что программа stunnel не заменяет протоколы POP3 или ШАР — она дополняет, добавляя общую поддержку SSL, любую выбранную службу. Это означает, что обычные службы POP3 и ШАР должны быть включены, — их нельзя удалить из файла /etc/inetd.conf. Если необходимо обязать клиентов подключаться только по безопасным каналам, придется использовать систему IPFW (как будет описано далее в этой главе), чтобы запретить подключения к соответствующим портам с любых хостов, кроме localhost.

## Защита FTP-сервера

Служба FTP, описанная в главе 27, — еще одна служба с передачей информации в явном виде. Она выполняет аутентификацию по паролю и поэтому может быть взломана злоумышленником, перехватывающим пакеты. FTP подвергает систему почти такому же риску, как и служба Telnet, поскольку часто используется, например, для

выгрузки на сервер Web-страниц, но не регулярно, с предсказуемыми интервалами, как POP3 или ШАР. Это делает ее менее рискованной, но не снимает проблемы защиты.

К счастью, защищенный вариант FTP обеспечивается протоколом SSH. Если в системе включена поддержка протокола SSH (как это сделать, было описано выше), защищенный вариант FTP уже доступен. Зашифрованные сеансы FTP работают по каналу SSH: клиент SSH устанавливает терминальное соединение, запускает серверную программу /usr/libexec/sftp-server и открывает необходимые подключения к клиенту по шифруемым каналам. Затем защищенный клиент FTP работает аналогично обычной программе FTP, прозрачно для пользователя.

В ОС FreeBSD работу клиентской части безопасного сеанса FTP обеспечивает встроенная программа sftp, входящая в состав пакета OpenSSH. В ОС Windows реализацию защищенного клиента FTP, работающего с ОС FreeBSD, предлагает пакет SSH Communications Security. Ранее упоминавшиеся клиенты Mac OS также поддерживают возможности защиты FTP.

Альтернативный способ передавать файлы по защищенному каналу состоит в использовании программы scp. Она позволяет копировать файлы на удаленный сервер и с удаленного сервера, используя аутентификацию учетной записи, и во многом аналогична программе гер (подробнее см. на страницах справочного руководства man гер и man scp), но передает данные по зашифрованному тоннелю SSH. Некоторые клиенты SSH, например NiftyTelnet/SSH для компьютеров Mac, поддерживают этот метод передачи файлов.

Чтобы переслать с помощью программы scp файл с локальной машины на удаленный сервер SSH, используется команда следующего вида:

```
# scp file.txt stripes:
frank@stripes's password:
file.txt 100% |*****| s h 00:00
```

После имени удаленного хоста указывается двоеточие (:), причем удаленный хост может быть как исходным, так и целевым в операции копирования. Можно также задавать полные имена файлов, определяющие их точное местонахождение. Это простой способ быстро и безопасно передать файлы, если не требуются разнообразные возможности настоящего FTP-клиента.

## Защита сервера Apache

Наконец мы добрались до протокола HTTP. Безопасность протокола HTTP жизненно важна для электронной торговли, когда защищать надо номера кредитных карточек и счетов клиента, а не регистрационные имена и пароли. Это не менее существенно, особенно если бизнес зависит от уверенности клиентов в соблюдении конфиденциальности передаваемой информации.

Защита протокола HTTP была одним из первых масштабных применений *системы* SSL, и хотя эта система защиты сегодня принята практически для всех популярных служб, интеграция ее в сервер Apache по-прежнему имеет признаки неорганизованности, столь свойственные SSL в первые годы развития. Существуют две различные, никак не связанные между собой версии сервера Apache с поддержкой SSL, разрабатываемых одновременно: Apache-SSL и mod\_ssl. Обе они включают библиотеки и средства OpenSSL системы FreeBSD, но поддерживаются разными *группами и решают различные* проблемы.

**ПРИМЕЧАНИЕ**

Защищенная и обычная текстовая версия протокола HTTP предназначены для совместной работы. Обслуживать все запросы HTTP через систему SSL — не очень хорошая и редко применяемая идея. Частично это связано с производительностью (скорость работы Web-сайта с большим количеством обращений резко снизится, что связано с дополнительными вычислениями при шифровании каждой страницы и изображения, независимо от наличия в них секретной информации), а частично — с неудобством и непривычностью такого способа взаимодействия. Большинство общедоступных данных Web не надо шифровать. Но переключение в защищенный режим при входе клиента на страницу покупки или в форму для сбора информации убеждает пользователей в том, что они вошли в более защищенную зону. Помните: вы должны удовлетворить ожидания пользователей и обеспечить конфиденциальность, а не только предоставить данные.

**Система Apache-SSL**

"Официальная" защищенная реализация сервера Apache, система Apache-SSL поддерживается собственно группой Apache Group, и имеет более ограниченный набор возможностей по сравнению с modssl. Эту реализацию используют скорее благодаря стабильности и производительности, чем из-за поддержки расширенных возможностей. Разработка системы Apache-SSL сейчас ведется не особенно активно в основном из-за строго контролируемого набора возможностей и отсутствия широко известных ошибок.

Двоичный модуль системы Apache-SSL называется httpsd, а не httpd. Идея в том, чтобы запускать стандартный демон httpd для обслуживания обычных запросов HTTP к порту 80, и демон httpsd для обслуживания зашифрованных запросов к порту 443. Конечно, это означает, что необходимо установить и обычную версию Apache, без поддержки SSL.

Систему Apache-SSL можно установить из набора портированных приложений (каталог /usr/ports/www/apache3-ssl), а ее официальный Web-сайт находится по адресу <http://www.apache-ssl.org>.

**Сервер Apache с модулем mod\_ssl**

Более полная и активно развивающаяся реализация протокола SSL для HTTP по сравнению с Apache-SSL, — modssl — представляет собой стандартный модуль Apache, подключающий библиотеку OpenSSL к серверу Apache с использованием современной модульной архитектуры сервера. Эта реализация более прозрачна, чем Apache-SSL, включает намного больше возможностей и более универсальную модель конфигурирования. Например, при установке портированного набора apache3-modssl устанавливается один выполняемый файл, httpd, как и для обычной версии apache3, но файл конфигурации специально настроен так, чтобы при необходимости включить поддержку протокола SSL:

```
<IfDefine SSL>
Listen 80
Listen 443
</IfD#fine>
```

К серверу Apache с модулем modssl можно добавлять и другие модули, например mod\_perl, mod\_jpht и т.п., из каталога /usr/ports/www. Основные отличия модуля mod\_ssl — богатый и полный набор возможностей, а также простота конфигурирования, хотя он и не настолько быстр и надежен, как система Apache-SSL. Хотя однозначных статистических данных об этом нет.

Сервер Apache с модулем modssl можно установить из каталога /usr/ports/www/apache3-modssl. Официальный Web-сайт проекта — <http://www.modssl.org>.

## Эксплуатация защищенного Web-сервера

Независимо от выбора реализации, Apache-SSL или Apache с модулем `mod_ssl`, поддерживать Web-сервер с защитой станет сложнее, чем без нее. Как уже было сказано, при использовании системы Apache-SSL предполагается, что одновременно работает обычный демон `httpd` для обслуживания обычных подключений по HTTP, передающих информацию в явном виде, и демон `httpsd` для зашифрованных запросов к порту 443. Это означает, что есть отдельная программа, `httpsdctl`, для управления системой Apache-SSL, работающая аналогично программе `apachectl` для обычного сервера Apache, а также имеется файл `httpsd.conf` в каталоге `/usr/local/etc/apache`, помимо файла `httpd.conf`. При установке сервера Apache с модулем `mod_ssl` никаких дополнительных компонентов не будет — все функциональные возможности включены в стандартный набор файлов, рассмотренный в главе 26.

При установке портированных приложений (как `apache 13-ssl`, так и `apache13-modssl`) устанавливаются полные деревья каталогов Apache, включая пиктограммы, примеры HTML-страниц, динамические модули и файлы конфигурации. Поэтому необходимо быть внимательным при обновлении параллельно установленных систем Apache-SSL и обычного сервера Apache. Версия с модулем `mod_ssl` представляет собой единый набор каталогов, заменяющий стандартную реализацию сервера Apache, поэтому сопровождение ее намного проще.

Сертификаты OpenSSL гораздо чаще читаются Web-браузером, чем любой другой защищенной службой, поэтому необходимо специально убедиться, что эти сертификаты соответствуют действительности. Если имя хоста в сертификате не соответствует имени хоста сервера или если сертификат не заверен общепризнанным сертификационным бюро (другими словами, если вы сами его заверили), пользователь получит диалоговое окно, показывающее полную информацию о вашем сертификате и запрашивающее подтверждения его приема браузером. Естественно, это может произвести неприятное впечатление на пользователя и лишить его уверенности в безопасности вашего сайта. Даже если сертификат правильно отражает информацию о сайте, пользователь может просмотреть сертификат с помощью средств информирования о защите (Security Information) своего браузера, так что все поля, заданные при генерации запроса на получение сертификата будут видны любой заинтересованной стороне. Помните об этом — информацию сертификата весьма сложно изменить после того, как он заверен.

## Плохо написанные сценарии CGI

Передача информации в явном виде — не единственная потенциальная дыра в защите сервера Apache. Не меньше надо беспокоиться о возможности нежелательных действий пользовательских сценариев в системе (как преднамеренных, так и случайных) и уничтожения ими файлов, особенно если модель защиты такова, что локальные пользователи могут не пользоваться доверием. Поскольку многие файлы в корневом каталоге документов сервера Apache принадлежат пользователю `nobody` (в частности, динамически создаваемые вашими же программами, выполняющимися на сервере) и поскольку *от имени того же пользователя nobody выполняются также и CGI-программы* всех пользователей, CGI-программа легко может удалить или изменить файл на сервере, принадлежащий пользователю `nobody`.

Это не так уж невероятно, как может показаться. Достаточно, чтобы CGI-программа удаляла ненужные собственные файлы пользователя, но по ошибке предвзяла имена файлов не тем именем каталога. То же самое может произойти с выда-

клялей данные в файл программой, которая может повредить данные другого пользователя. Даже самые заслуженные разработчики CGI-программ раньше попадались в подобную ловушку. Опасность значительно возрастает, если в системе есть злонамеренный пользователь, намеренно создающий деструктивный сценарий для выполнения сервером Apache от имени пользователя nobody.

Решение этой проблемы — запускать сервер Apache в программе -"обертке", которая перехватывает CGI-программы пользователей, проверяет их безопасность (убеждаясь в соответствии прав доступа) и запускает их от имени соответствующих пользователей, а не от имени nobody. Для этого традиционно использовалась программа suexec, поставляемая в составе сервера Apache. Но более простым и гибким решением является другая система, CGIWrap, созданная Натаном Ньюлингером (Nathan Nculingcr).

## Повышение безопасности сценариев CGI с помощью системы CGIwrap

Система CGIWrap, доступная в наборе портированных приложений в каталоге /usr/ports/www/cgiwrap, обеспечивает двойное преимущество: защищает пользователей и корневой каталог документов сервера от угрозы плохо написанных или враждебных CGI-сценариев и позволяет при этом пользовательским CGI-программам записывать файлы, которые сами пользователи смогут изменять или удалять в командном интерпретаторе. Если задуматься, намного больше смысла в выполнении CGI-программы пользователем, которому она принадлежит, а не непривилегированным пользователем nobody. В идеальном мире, где программы всегда совершенны и безошибочны, и никто не пытается намеренно уничтожить чужие файлы, именно так и должны работать все Web-серверы.

Однако наш мир не идеален — CGI-программы пишутся с ошибками, и кругом полно хакеров. Пользователю достаточно создать CGI-программу, принадлежащую root, и поместить ее в свой каталог, ожидая, пока она не будет выполнена от имени владельца, суперпользователя, и вся ее разрушительная мощь не обрушится на выбранные в системе файлы.

Система CGIWrap защищает сервер от подобных проблем. Это решение, конечно, — не полное и не идеальное, но оно снимает подавляющее большинство угроз защите, связанных с пользовательскими сценариями CGI, настолько, что администратор может направить свои усилия по защите на другие области. Проверяя перед выполнением все пользовательские программы CGI на предмет защиты и запуская каждую программу от имени владельца, система переносит все неотвратимые угрозы, которые несут некорректно написанные сценарии CGI, с сервера на владельца сценария.

При установке системы CGIWrap из каталога портированных приложений программа cgiwrap (скомпилированная двоичная программа) попадает в каталог cgi-bin верхнего уровня, /usr/local/www/cgi-bin. CGIWrap не работает в качестве оболочки сервера Apache, как система suexec, а должна вызываться пользователями непосредственно, путем указания URL следующего вида:

```
http://www.somewhere.com/cgi-bin/cgiwrap/frank/myscript.cgi
```

Или в директиве включения на стороне сервера:

```
<!--#include virtual="/cgi-bin/cgiwrap/frank/myscript.cgi"-->
```

В результате выполняется программа myscript.cgi из каталога /home/frank/public\_html/cgi-bin. Все CGI-программы пользователя должны устанавливаться в его

каталоге `public_html/cgi-bin`. Если этот каталог отсутствует, его необходимо создать. CGI-программы из других каталогов не будут обрабатываться системой CGIWrap, поэтому важно отключить возможность выполнения CGI за пределами дерева каталогов `DocumentRoot`-сервера, убедившись, что в блоках, определяющих каталоги пользователей, нет директивы `Options +ExecCGI`.

Официальная Web-страница системы CGIWrap с дополнительной информацией находится по адресу <http://cgiwrap.unixtools.org/>.

## Профили защиты системы и защита ядра (securelevel)

Ядро ОС FreeBSD может работать с пятью различными уровнями защиты, от -1 до 3, которые задаются с помощью опции `kern_securelevel` в файле `/etc/rc.conf`. Каждый из этих уровней соответствует профилю, управляющему параметрами вроде возможности замены ядра на диске, загрузки и выгрузки модулей ядра, установки и изменения определенных прав доступа к файлам и флагов, монтирования файловых систем по требованию, а также отключения или изменения утилит типа IPFW (встроенный брандмауэр, который мы вскоре рассмотрим). Как было описано в главе 17, значение опции `securelevel` по ходу работы можно только увеличить — для его снижения необходима перезагрузка. Дополнительную информацию по защите ядра можно найти на странице справочного руководства `man securelevel`.

В ОС FreeBSD есть также и другой набор профилей многоуровневой сетевой защиты. Вы видели его в ходе установки и можете увидеть снова в программе `/stand/sysinstall`, выбрав пункты меню "Configure", а затем "Security". Полученное меню позволяет выбрать один из четырех общесистемных профилей защиты: Low, Medium, High и Extreme. Эти профили управляют возможностью запуска служб типа `sendmail`, `sshd` и `inetd` и примерно соответствуют уровням защиты ядра. В табл. 29.1 для каждого профиля представлены устанавливаемые им опции в файле `/etc/rc.conf`:

**Таблица 29.1. Общесистемные профили защиты**

Имя профиля	Установки в файле <code>/etc/rc.conf</code>
Low	<code>sendmail_enable="YES"</code> <code>sshd_enable="YES"</code> <code>portmap_enable="YES"</code> <code>inetd_enable="YES"</code>
Medium	<code>sendmail_enable="YES"</code> <code>sshd_enable="YES"</code> <code>inetd_enable="YES"</code>
High	<code>kern_securelevel_enable="YES"</code> <code>sendmail_enable="YES"</code> <code>sshd_enable="YES"</code> <code>portmap_enable="NO"</code>

Имя профиля	Установки в файле /etc/rc.conf
	nfs_server_enable="NO"
	inetd_enable="NO"
Extreme	kern_securelevel="2"
	kern_securelevel_enable="YES"
	sendmail_enable="NO"
	sshd_enable="NO"
	portmap_enable="NO"
	nfs_server_enable="NO"
	inetd_enable="NO"

Как и можно было ожидать, профиль "Extreme" ограничивает систему почти до уровня бесполезности. Уровень защиты ядра `securelevel` устанавливается равным 2, вследствие чего ядро нельзя изменять (с помощью модулей) или устанавливать новое без перезагрузки в однопользовательский режим, а единственный способ смонтировать или демонтировать файловую систему — явно, с помощью команд `mount` и `umount` (неявное монтирование по требованию, обеспечиваемое демоном `amd`, не допускается). Кроме того, демон `inetd`, система `sendmail`, демон `sshd`, сервер NFS и другие службы отключены.

Профиль "High" — менее ограничительный. Он задает ядру уровень защиты `securelevel` 1, тем самым упрощая монтирование файловых систем, но ядро тоже нельзя изменять. Демоны `sendmail` и `sshd` включены, но все остальные службы, упомянутые в профиле "Extreme", отключены.

Конечно, поскольку профили защиты работают исключительно путем установки опций в файле `/etc/rc.conf`, вы можете при необходимости комбинировать представленные в табл. 29.1 установки, создавая профиль защиты, соответствующий выбранной модели защиты системы. Не следует включать службу без необходимости. Если она не используется по назначению, то может оставаться бесполезной и безвредной; однако в ее защите может быть обнаружено уязвимое место, открывающее систему для взлома. Предохраняйтесь, по возможности.

## Использование брандмауэра

Никто не может отрицать, что *брандмауэры*, или машины, работающие в качестве маршрутизаторов с фильтрами, становятся все более важной и даже незаменимой частью сети, подключенной к Internet. Легкодоступные средства взлома, применяемые изнывающими от безделья "юными хакерами" ("script kiddies"), требуют, помимо отказа от определенных служб и постоянного слежения за публикациями, посвященными защите, дополнительного уровня защиты. Необходим универсальный щит на уровне ядра, предотвращающий вообще доступ к определенным портам системы, с определенных хостов или по некоторым протоколам. Брандмауэры, в особенности брандмауэр IPFW, входящий в состав ОС FreeBSD, удовлетворяют эту потребность.

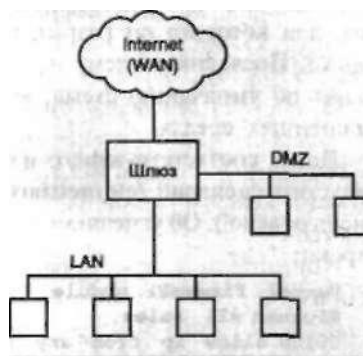
Брандмауэр может предотвратить подавляющее большинство случайных атак, пропуская только пакеты, которые администратор счел допустимыми. Однако еще одна аксиома сетевой защиты гласит, что даже самый дорогой и надежный брандмауэр становится бесполезным в случае неправильного конфигурирования. Причина неэф-

фektivности брандмауэров — результат неправильного конфигурирования; а не их качество. Постоянно обновляемые и хорошо продуманные правила защиты не заменит никакой "компетентный" брандмауэр, поэтому не обольщайтесь, что, купив более дорогой брандмауэр, сможете решить все проблемы защиты. Именно такое представление приводит к столь многочисленным сегодня взломам защиты в internet.

С помощью брандмауэра можно делать две вещи. Во-первых, фильтровать пакеты по заданным критериям, отвергая нежелательную информацию на уровне ядра (до того, как она достигнет критических служб системы). Во-вторых, можно вести учет, сохраняя статистическую информацию об использовании системы и следя за тем, какие объемы информации и откуда поступают. Система IPFW делает и то, и другое; ее можно запускать непосредственно на своей машине с ОС FreeBSD или использовать на системе, работающей как шлюз, защищающий несколько хостов локальной сети. На рис. 29.1 представлен как раз последний случай, когда машина с ОС FreeBSD и тремя сетевыми картами работает в качестве шлюзового маршрутизатора (см. главу 28), передающего пакеты между внутренней локальной сетью (LAN), "демилитаризованной зоной" (DMZ) и внешними сетями (WAN).

**Рисунок 29.1.**

*Диаграмма шлюзового маршрутизатора, обеспечивающего функции брандмауэра. Показаны интерфейсы LAN, DMZ и WAN*



#### 'ПРИМЕЧАНИЕ

"Демилитаризованная зона" — это сеть, непосредственно доступная для информации из глобальной сети, а не из локальной. Особенно полезная в случае преобразования IP-адресов в локальной сети (как было показано в главе 22) демилитаризованная зона представляет собой сеть для предприятия или поставщика услуг Internet, в которую помещаются машины с реальными адресами, например Web-серверы, шлюзы электронной почты и другие хосты, которые должны быть доступны из общей сети Internet.

Зона DMZ может быть защищена правилами брандмауэра на шлюзовом маршрутизаторе, а может и не защищаться. Защита обычно желательна, но в некоторых специальных случаях требуется, чтобы демилитаризованная зона была освобождена от общих правил брандмауэра локальной сети. Это зависит от конфигурации и назначения сети.

## Включение поддержки брандмауэра

Система IPFW не поддерживается в ядре GENERIC. Имеется ряд опций, которые необходимо скомпилировать в специализированном ядре для ее поддержки, как было описано в главе 17: Это опции IPFIREWALL, IPFIREWALL\_VERBOSE и IPFIREWALL\_VERBOSE\_LIMIT=10. Однако для использования системы IPFW вовсе не обязательно создавать новое ядро — она доступна в виде модуля ядра, автоматически загружаемого сценарием /etc/rc.network. Чтобы включить поддержку брандмауэра без перестройки ядра, добавьте следующие строки в файл /etc/rc.conf:

```

firewall_enable="YES"
firewall_type="open"

```

Если не указать, что брандмауэр должен быть открытого типа (`firewall_type="open"`), система IPFW запустится с единственным стандартным правилом с индексом 65535 (максимальная защита), запрещающим передачу любых IP-пакетов. Другими словами, после перезагрузки с включением системы IPFW стандартной конфигурации, машина будет полностью отрезана от сети, и, чтобы исправить это, необходим доступ к физической консоли.

^Зчень опасно экспериментировать с системой IPFW при отсутствии доступа к консоли, поскольку можно сделать машину недоступной. Пока вы не ознакомитесь с системой IPFW настолько, что будете точно знать последствия своих действий, всегда убеждайтесь в наличии доступа к консоли на случай, если что-то пойдет не так.

Установка типа брандмауэра `open` изменит стандартные правила так, что все IP-пакеты по умолчанию пропускаются (разрешаются), а не блокируются. Помните, что действительно защищенная система должна запрещать доступ ко всем хостам, кроме тех, для которого он разрешен явно, а не разрешать все, кроме нескольких исключений. Последнюю схему нельзя сделать полностью безопасной. Однако ограничивающая по умолчанию схема, вероятно, избыточна для систем, работающих в уже защищенных средах.

После соответствующего изменения файла `/etc/rc.conf` перезагрузите систему или запустите сценарий `/etc/netstart` с физической консоли (не с удаленного терминала — это очень опасно!). Об успешном включении брандмауэра будут свидетельствовать следующие строки:

```

kernel firewall module loaded
Flushed all rules.
00100 allow ip from any to any via lo0
00200 deny ip from any to 127.0.0.0/8
00300 deny ip from 127.0.0.0/8 to any
65000 allow ip from any to any
Firewall rules loaded, starting divert daemons:.

```

Чтобы убедиться, что модуль системы IPFW автоматически загружен, можно воспользоваться программой `kldstat`:

```

# kldstat
Id Refs Address      Size  Name
1      3 0xc0100000 355be4 kernel
2      1 0xc0eee000 6000  ipfw.ko
3      1 0xc0f19000 12000  linux.ko

```

Теперь доступна в полном объеме команда `ipfw`, позволяющая устанавливать правила пропуска пакетов и просматривать учетную информацию, накопленную системой IPFW.

## Ж ПРЕДУПРЕЖДЕНИЕ

При использовании системы NATD для совместного использования подключения к Internet, как описано в главе 28, все равно необходимо строить специализированное ядро, поскольку для системы NATD требуется включение в ядре опции **IPDIVERT**. Подробнее об этом см. в главе 28.

## Конфигурирование системы IPFW

Для добавления или удаления правил системы фильтрации и учета пакетов ядра используется команда `ipfw`. Правила строятся по синтаксису, отдаленно напоминающему обычный английский язык; они состоят из действия (например, `deny`), протокола, к которому применяется действие (например, `tcp`), и спецификации адреса, включающей конструкции `from` и `to`. Например, правило, блокирующее пакеты TCP, которые поступают с хоста `badhost.com`, будет иметь вид `deny tcp from badhost.com to any`, а добавить его к брандмауэру ядра можно так:

```
# ipfw add deny tcp from badhost.com to any
```

Возможны и определенные вариации этой конструкции. Вместо адреса можно задавать маску сети — либо в виде шаблона маски (например, `255.255.255.0`), либо с помощью битовой маски CIDR (например, `/24`). Можно также блокировать отдельные порты, а не всю систему, т.е. предотвратить доступ хостов с непредсказуемыми IP-адресами к определенным службам сервера. Такое правило может выглядеть, например, так:

```
# ipfw add deny all from evil.isp.com/16 to www.somewhere.com 80
```

Можно исключать хосты из предшествующих правил, указывая их отдельно в правилах `allow`, например, так:

```
# ipfw add allow all from goodhost.evil.isp.com to
"www.somewhere.com 80
```

Учтите, что правила системы IPFW проверяются в порядке задания. Каждое правило имеет индекс, обычно на 100 больший, чем у предшествующего, причем индекс можно задавать явно, после ключевого слова `add` в команде `ipfw`. Индексы позволяют задать порядок применения правил. Текущий набор правил можно получить с помощью команды `ipfw -a list`:

```
# ipfw -a list
00100  0      0  allow ip from any to any via lo0
00200  0      0  deny ip from any to 127.0.0.0/8
00300  0      0  deny ip from 127.0.0.0/8 to any
00400  0      0  deny tcp from badhost.com to any
00500  0      0  ipfw add deny all from evil.isp.com/16 to
"mra.somewhere.com 80
00600  0      0  ipfw add allow all from goodhost.evil.isp.com to
"www.somewhere.com 80
65000 1214  79688  allow ip from any to any
65535  1      40  deny ip from any to any
```

Индекс выдается в первом столбце. Правило с индексом 65535, как уже было сказано, — это стандартное запрещающее правило, которое отвергает все пакеты, не разрешенные предыдущими правилами. Поскольку был задан открытый тип брандмауэра, перед ним помещено разрешающее передачу любых пакетов правило `allow`, но тоже с достаточно большим индексом, так что оно должно проверяться после всех остальных правил, добавленных при обычном использовании.

Во втором и третьем столбцах представлена статистика использования, показывающая количество пакетов и байтов, соответствовавших каждому правилу. Так можно понять, эффективны ли заданные правила.

В файле `/etc/rc.conf` можно задавать различные типы брандмауэров. Каждое ключевое слово имеет особое значение, представленное в табл. 29.2. Точные определе-

ния этих профилей можно получить, разобрав код сценария командного интерпретатора `/etc/rc.ngrewau`.

Таблица 29.2. Доступные типы брандмауэре

<i>Ключевое слово</i>	<i>Значение</i>
<code>Open</code>	Разрешает прохождение всех входящих и исходящих пакетов
<code>closed</code>	Запрещает все пакеты IP, кроме проходящих через интерфейс заковыдывания <b>(100)</b>
<code>client</code>	Устанавливает правила для защиты только данной машины
<code>simple</code>	Устанавливает правила, предназначенные для защиты всей сети
<code>unknown</code>	Не загружает никакие правила, кроме стандартного запрещающего с индексом 65535
<code>&lt;имя файла&gt;</code>	Загружает правила из файла с указанным именем

Для ваших целей может подойти "готовый" профиль IPFW, например `client` или `simple`. Вероятно, однако, что при развитии системы потребуется специализированная конфигурация. Специализированный набор правил необходимо поместить в выбранный файл конфигурации; например `/etc/Grewall.conf`. Укажите необходимый набор правил в этом файле, не задавая саму команду `ipfw`.

```
add deny tcp from badbost.com to any
add deny all from evil.isp.com/16 to www.somewhere.com 80
add allow all from goodhost.evil.isp.com to www.somewhere.com 80
add 65000 allow all from any to any
```

Теперь измените тип брандмауэра (значение `Grewall_type`) в файле `/etc/rc.conf`:

```
firewall_type="/etc/firewall.conf"
```

При следующей перезагрузке или запуске файла `/etc/netstart` правила из файла `/etc/firewall.conf` будут загружены с индексами 100, 200, 300 и т.д. Правило `allow aD` с индексом 65000 задает стандартное поведение — пропускать, а не отвергать пакеты. Если хотите, задайте и его.

Дополнительную информацию о системе IPFW можно найти на странице справочного руководства `man ipfw` и в руководстве *FreeBSD Handbook* по адресу <http://www.freebsd.org/handbook>.

## Предотвращение вторжений и взломов

Брандмауэры, правила задания паролей и шифрование являются существенной защитой системы от неавторизованного доступа. Но всего этого не достаточно для защиты от действительно целеустремленного хакера, у которого есть "rootkit" или другое средство, разработанное для использования одного из известных уязвимых мест в той или иной службе. Имеется множество средств защиты, выходящих далеко за рамки возможностей простого брандмауэра. Они динамически блокируют подозрительные хосты, следят за вторжениями и избирательно контролируют доступ каждого хоста к отдельным службам. Давайте рассмотрим некоторые из этих средств.

## Использование PortSentry

PortSentry, предлагаемый компанией Psionic Software, — это лемон, контролирующий входящие сетевые пакеты, прослушивающий указанные порты и выявляющий пакеты, которые могут свидетельствовать о возможности сканирования портов — предварительной атаки, в ходе которой взломщик ищет поддерживаемые системой службы, чтобы попытаться их взломать. Выявив такие пакеты, демон PortSentry блокирует доступ соответствующего хоста к системе, перенаправляя его обращения в "черную дыру" или добавляя для него правило системы IPFW, так что все дальнейшие попытки подключения с этого хоста отвергаются. Демон PortSentry контролирует как TCP-, так и UDP-пакеты; он динамически строит таблицу запрещенных подключений, используемую как антитело против вируса, и реагирует на подозрительные действия превентивно, блокируя их прежде, чем они смогут нанести вред.

Являясь программой с открытым исходным кодом, PortSentry может быть установлена из каталога портированных приложений (/usr/ports/security/port Sentry). Двоичная программа portsentry устанавливается в каталог /usr/local/bin, а в качестве конфигурационного используется файл /usr/local/etc/port Sentry.conf. Откройте этот файл в любом текстовом редакторе — для правильной работы демона PortSentry его необходимо отредактировать.

Прежде всего необходимо решить, какой набор портов должен контролировать демон PortSentry. Имеется три готовых набора в виде пар, начинающихся с TCP\_PORTS и UDP\_PORTS. Первый набор — достаточно большой; он предназначен для реализации очень строгих правил, по которым любой подозрительный удаленный доступ к порту автоматически вызывает срабатывание блокирующей правила. Второй набор — более либеральный, а третий — минимальный, задающий слежение только за портами, к которым удаленный пользователь не должен подключаться, если только он не выполняет сканирование портов или не пытается организовать атаку. Второй, либеральный набор ("если вы хотите знать"), используется по умолчанию. Чтобы перейти на любой из остальных двух наборов, прокомментируйте этот набор и уберите комментарий с того набора, который необходимо задействовать. При необходимости можно создать собственный набор. Убедитесь, что в соответствующий список не входят базовые службы! Например, порт 143 указан во всех трех готовых наборах, но 143 — это порт для протокола IMAP. Если вы поддерживаете службы IMAP, не забудьте удалить порт 143 из списка. В противном случае оставьте этот порт в списке — это предотвратит попытки воспользоваться недостатками в защите протокола IMAP.

Затем необходимо выбрать метод блокировки подозрительных хостов. Это можно делать одним из двух способов: с помощью системы IPFW (если она задействована, как было описано ранее в данной главе) или с помощью маршрутов в никуда (black-hole routes) (если система IPFW не используется). Выбор метода состоит в снятии комментария с одной строки, начинающейся ключевым словом KILL\_ROUTE и задающей команду системы, которую демон PortSentry должен использовать для блокировки подозрительного хоста.

Для системы, в которой работает IPFW, раскомментируйте строку, содержащую вызов /sbin/ipfw, например:

```
# Те, кто использует ОС FreeBSD (и другие совместимые с ней
# системы), могут также воспользоваться встроенными функциями
# брандмауэра.
KILL_ROUTE='sbin/ipfw add 1 deny all from $TARGETS:255.255.255.255 to any'
```

Если система IPFW не задействована, используйте метод маршрутов в никуда, который, несмотря на разные мнения по этому поводу, не менее действенен, чем использование системы IPFW:

```
# FreeBSD (не очень хорошо протестирован.)
KILL_ROUTE="route add -net $TARGET$ -netmask 255.255.255.255 127.0.0.1 -
^•blackhole"
```

После установки метода блокирования демон PortSentry готов к работе. Однако на момент написания этой главы в состав портированного приложения PortSentry не входит сценарий автоматического запуска. Можете использовать сценарий, представленный в листинге 29.1 (доступен на поставляемом с книгой компакт-диске в файле portsentry.sh). Проверьте, установлено ли для него право на выполнение и скопируйте его в каталог /usr/local/etc/rc.d, чтобы демон PortSentry запускался при каждой перезагрузке системы.

Листинг 29.1. Пример файла запуска PortSentry.

```
#!/bin/sh
PORTSENTRY="/usr/local/bin/portsentry"
case "$1" in
  start)
    ${PORTSENTRY} -tcp SS echo " Starting PortSentry TCP mode
    ${PORTSENTRY} -udp f& echo " Starting PortSentry OTP mode

  stop)
    killall 'basename ${PORTSENTRY} '

  *)
    echo ""
    echo "Osage: 'basename $0' { start | stop } '•
    echo ""
esac
```

### СОВЕТ

При работающем демоне PortSentry с помощью команды sockstat можно узнать, какие порты он прослушивает

```
# sockstat
USER  COMMAND  PID  FD  PROTO  LOCAL  ADDRESS  FOREIGN  ADDRESS
root  portsent 2432  0  udp4   *      1        *      *
root  portsent24 32  1  udp 4   *      7        *      *
root  portsent24 32  2  udp 4   *      9        *      *
root  portsent24 32  3  udp 4   *      69       *      *
root  portsent24 32  4  udp4   *      161     *      *
root  portsent24 32  5  udp4   *      162     *      *
```

При каждом выявлении попытки атаки соответствующий хост и порты, которые он сканировал, что вызвало срабатывание системы, регистрируются в файле /usr/local/etc/portsentry.blocked.tcp для атак с использованием протокола TCP и в файле /usr/local/etc/portsentry.blocked.udp для атак с помощью протокола UDP. Так, демон PortSentry следит за тем, чтобы уже заблокированные не блокировались повторно. Эти файлы также позволяют понять, действия каких хостов были перехвачены системой выявления атак. Эти файлы автоматически очищаются при каждом запуске де-

мона PortSentry. Учтите, что при перезагрузке исчезают как дополнительные правила системы *IPFW*, так и записи в таблице маршрутизации, добавленные по ходу работы, так что однажды заблокированный хост может снова получить доступ к машине после перезагрузки, по крайней мере до следующей попытки просканировать ее порты.

#### ^ СОВЕТ

При использовании метода **IPFW KILL\_ROUTE** текущие блокирующие правила можно получить с помощью команды `ipfw -a list`:

```
# ipfw -a list
00001 1 44 deny ip from 209.237.26.165 to any
```

При использовании маршрутов в никуда контролировать работу можно с помощью команды `netstat -rn`:

```
# netstat -rn
Routing tables
```

Internet:

Destination	Gateway	Flags	Refs	Ose	Netif	Expire
209.237.26.165/32	127.0.0.1	UGScB	0 0	1o0		

Флаг B отмечает маршрут в никуда ("black-hole" route), по которому пакеты просто отвергаются.

#### ap СОВЕТ

При каждом выявлении атаки в файл `/var/log/messages` добавляется строка, показывающая, что предпринял в ответ на нее демон PortSentry:

```
Jun 2 23:50:56 stripes portsentry[2430]: attackalert: Connect from host:
209.237.26.165/209.237.26.165 to TCP port: 1
Jun 2 23:50:56 stripes portsentry[2430]: attackalert: Host 209.237.26.165
has been blocked via wrappers with string: "ALL: 209.237.26.165"
Jun 2 23:50:56 stripes portsentry[2430]: attackalert: Host 209.237.26.165
has been blocked via dropped route using command: "/sbin/ipfw add 1 deny
all from 209.237.26.165:255.255.255.255 to any"
```

Еще одно средство от создателей демона PortSentry — программа Logcheck {доступная в каталоге `/usr/ports/security/logcheck`} — позволяет анализировать этот и другие журнальные файлы и посылать администратору ежедневный отчет о любых необычных событиях или выявленных атаках.



#### ПРЕДУПРЕЖДЕНИЕ

Как и в случае системы IPFW, будьте внимательны при тестировании демона PortSentry. Очень легко настроить его так, чтобы система, с помощью которой вы управляете своей машиной с ОС FreeBSD, оказалась заблокированной, например, при попытке подключиться с помощью Telnet к порту 1, чтобы узнать, что получится. В результате ваш хост окажется заблокированным, и все *остальные попытки подключения по завершении времени ожидания окончатся неудачей. Придется* подключиться с другого хоста или с физической консоли, чтобы удалить добавленное по ошибке правило с помощью команды `ipfw delete 1` либо `route delete <IP-адрес>/32`

Если необходимо защитить определенные хосты (например, ваши собственные машины) от блокирования демоном PortSentry, добавьте их IP-адреса в файл `/usr/local/etc/portsentry.ignore`

## Использование файла `/etc/hosts.allow`

Файл `/etc/hosts.allow` позволяет блокировать доступ определенных хостов к указанным службам системы. Его можно использовать как неавтоматическую версию

демона PortSentry. Можно задать блок правил для службы так, что каждое правило применяется к определенному IP-адресу или набору адресов и либо разрешает, либо запрещает соответствующим хостам доступ к службе. Вот пример блока правил из стандартного файла /etc/hosts.allow, в который добавлено несколько дополнительных строк, иллюстрирующих допустимые синтаксические конструкции:

```
sendmail    localhost    allow
sendmail    .nice.guy.example.com  allow
sendmail    .evil.cracker.example.com  deny
sendmail    231.21.15.0/255.255.255.0  deny
sendmail    12.124.231.    : deny
sendmail    ALL          allow
```

Поскольку работа системы IPFW уже рассмотрена, формат этих правил понять несложно. Правило состоит как минимум из трех полей: служба (задаваемая именем соответствующего процесса), имя или IP-адрес обращающегося хоста и предпринимаемое действие (или несколько действий, если полей — четыре и более). Первые два столбца могут содержать списки (несколько записей, разделенных пробелами), а столбец хостов может задаваться во множестве форматов: значение, начинающееся точкой, позволяет задать целую подсеть системы DNS, а заканчивающееся точкой — подсеть на базе IP-адреса. Указав IP-адрес и после косой черты маску сети, можно задать сеть. Блок должен заканчиваться "стандартным" правилом, определяющим, должен ли быть разрешен или запрещен доступ к службе во всех остальных случаях. В общем случае, если уж служба поддерживается, значит, это действительно нужно, поэтому стандартное правило будет "разрешающим".

С помощью файла /etc/hosts.allow можно делать вещи и поинтереснее, задавая другие действия, помимо "allow" и "deny". Можно сделать так, чтобы при неавторизованном доступе к службе администратору посылалось сообщение электронной почты или запускалась программа, выполняющая ту или иную встречную проверку уязвимости соответствующего удаленного хоста (хотя это, вероятно, не лучшая идея). При каждом срабатывании правила можно выполнить любую команду командного интерпретатора. Стандартное правило для службы **fingerd** показывает пример такой конфигурации:

```
fingerd    : ALL \
            : spawn (echo Finger. | \
            /usr/bin/mail -s "tcpd\ : %u@%h[%a] fingered me!" root) & \
            deny
```

Коды %u, %h и %a и дополнительные опции конфигурации описаны на страницах справочного руководства `man 5 hosts_access` и `man hostsoptions`

## Использование системы Tripwire

Особняком от непосредственного блокирования подозрительных хостов стоит задача выявления вторжения. Достаточно коварный хакер сможет обойти защиту, независимо от того, насколько тщательно она организована. Если это произошло, необходимы средства, позволяющие понять, насколько далеко зашло вторжение и не нанес ли он вред системе. Если в систему вторглись, хотелось бы узнать об этом сразу и оценить реальный ущерб от вторжения.

Популярным средством для решения этой задачи является система Tripwire, доступная в каталоге /usr/ports/security/tripwire. Это средство вычисляет запись аутентичности для всех программ в системе, сравнивая каждую из них ежедневно с контрольной записью аутентичности, созданной при первом запуске сразу после

установки. При выявлении любых отличий (например, если случайно изменился размер выполняемого файла `sshd` или произошло изменение его содержимого или метаданных, система Tripwire уведомит об этом администратора по электронной почте. Это позволяет оценить, была ли взломана система.

При первоначальной установке система Tripwire строит начальную базу данных "отпечатков" программ в ходе выполнения фазы `make install` и записывает ее в файл в каталоге `/var/adm/tcheck`. Однако с этим связана потенциальная угроза защите — атакующий, получив доступ к системе и обнаружив наличие базы данных Tripwire, просто изменит ее, так что система не сможет обнаружить его присутствие. Вот почему имеет смысл хранить базу данных не на машине, а на другой системе либо, что наиболее удобно, на дискете.

ОС FreeBSD предлагает простой способ создать такую архивную дискету. Необходимо только поместить дискету в дисковод перед началом фазы `make install`, а затем задать присвоение значения переменной `TRIPWIRE_FLOPPY=YES` в командной строке `make install`:

```
# make install TRIPWIRE_FLOPPY=YES

### Фаза 3: Создание базы данных с информацией о файлах
###
### Предупреждение: база данных помещается в файл
###                      ./databases/tw.db_stripes.somewhere.com.
###                      Не забудьте перенести этот файл и файл
###                      конфигурации на безопасный носитель!
###
###                      (Tripwire ищет этот файл в каталоге
###                      ^/var/adm/tcheck/databases*.)
# готовим дискету
/dev/rfd0c: 2880 sectors in 80 cylinders of 2 tracks, 18 sectors
          1.4MB in 5 cyl groups (16 c/g, 0.28MB/g, 32 i/g)
super-block backups (for fsck -b #) at:
32, 632, 1184, 1784, 2336"
mount /dev/fd0c /mnt
# переносим базу данных на дискету
# Не забудьте вынуть дискету и защитить ее от записи.
```

Теперь на дискете находится копия начальной базы *данных* системы Tripwire, а также утилиты `tripwire`, `tweheck` и `gunzip` и копия файла `tw.config` — все, что необходимо для восстановления начальной базы данных с дискеты.

После этого можно организовать запуск программы `tripwire` каждую ночь как периодической задачи, — как это сделать, описано в главе 14. Запущенная без аргументов программа `tripwire` работает в режиме проверки согласованности, сканируя все файлы, указанные в файле `/var/adm/tw.config`, в поисках отличий от базы данных "отпечатков" файлов. Выявив несогласованность, программа сообщает о ней, как в следующем примере, в котором изменилось время модификации файла `/usr/sbin/`

```
# tripwire

### Phase 3: Creating file information database
### Phase 4: Searching for inconsistencies
*#<
###          Total files scanned:          16803
*#<          Files added:                   0
```

```

###          Files deleted:          0
###          Files changed:         14321
###
###          After applying rules:
###          Changes discarded:     14320
###          Changes remaining:      1
**#
changed:  -r-xr-xr-x  root  197940  (null)  /usr/sbin/sshd
### Phase 5: Generating observed/expected pairs for changed files
MI
### Attr   Observed (what it is)     Expected (what it should be)
###-----! . . . . . -' .
/usr/sbin/sshd
  st_mtime: Sun Jun 3 00:55:51 2001 Sat Apr 28 21:17:19 2001
  st_ctime: Sun Jun 3 00:55:51 2001 Sat Apr 28 21:17:19 2001

```

Может оказаться, что вы ожидаете отличия этого файла от записанного в базе данных системы Tripwire. Например, после установки обновленной версии демона sshd. После любого обновления файлов, контролируемых системой Tripwire, необходимо обновить и базу данных Tripwire, чтобы в ней была отражена новая информация. Это делается с помощью опции `-update`:

```
# tripwire -update /usr/sbin/sshd
```

В результате в текущем каталоге будет создан подкаталог базы данных, содержащий новый файл базы данных (который необходимо перенести в `/var/adm/tcheck/databases`) и резервную копию старой базы данных. Эти базы данных не сжаты; может потребоваться сжать их с помощью программы `gzip` и скопировать на дискету Tripwire:

```

# gzip databases/tw.db stripes.somewhere.com
# mount /dev/fd0 /floppy
# cp databases/tw.db_stripes.somewhere.com /floppy
# umount /floppy

```

### ГДЕЛЬНОЕ ХРАНЕНИЕ ДАННЫХ

Идея хранения базы данных системы Tripwire на дискете связана с принципом пространственного промежутка (`air gap`) — принципом защиты, состоящим в том, что никакой автоматический процесс не может перенести данные с одной стороны этого промежутка на другую. Любая система, в которой данные передаются программно из точки А в точку Б, потенциально подвержена вторжению талантливого взломщика. Имея достаточно времени и приложив определенные усилия, тот, кто намерен заполучить ваши данные, сможет получить доступ к архивам, даже если используются сложные схемы типа второго "скрытого" жесткого диска, монтирующегося автоматически ночью для выполнения резервного копирования. Это все равно автоматизированная процедура, поэтому она уязвима для желающего взломать ее защиту.

Использование промежуточного пространства — это "последнее" средство, дающее гарантированную защиту за счет менее удобной процедуры. Если данные хранятся физически отдельно от машины, подключенной к сети, никакой хакер не сможет их заполучить. Именно так устроены сети медицинских и государственных учреждений: критические базы данных хранятся на машинах, не подключенных к сети. Они отделены пространством и поэтому защищены (по крайней мере настолько, насколько можно доверять администратору, имеющему физический доступ к системам).

Если секретные данные хранятся на дискете, компакт-диске CD-R или на другом съемном носителе отдельно, значит, до ваших "ключей от города" не смогут добраться, даже если вся сеть будет взломана. Конечно, если вы не забыли дискету в дисковом.

## Если кажется, что система взломана

Независимо от вашей внимательности и количества принятых превентивных мер, невозможно быть на 100 процентов уверенным в защите системы. Абсолютная защита — настолько же недостижимый идеал, как и абсолютный нуль температуры или полный вакуум. И поскольку остается хоть минимальный шанс наличия бреши в защите системы, в случае любых сомнений приходится готовиться к худшему.

Хотя системы Tripwire и PortSentry могут существенно облегчить администратору работу по предотвращению и выявлению вторжения, он все равно должен следить за изменениями в поведении системы. Следите за результатами выполнения команды `top`, контролируйте загрузку системы в течение длительных периодов времени — не становится ли она больше. Если — да, исследуйте, с чем это может быть связано. Не игнорируйте странные изменения поведения, например необычно сформатированные приглашения регистрации или непривычные результаты выполнения стандартных команд. Периодически просматривайте каталоги `/tmp` и `/var/tmp`, обращая внимание на выполняемые файлы, установленные биты `setuid` или слишком большие файлы, и регулярно очищайте эти каталоги. Просматривайте журнальные файлы системы в каталоге `/var/log`; обращайте внимание на подозрительные сообщения, например, с длинными строками или мусорными символами — это почти наверняка результат попыток взлома путем переполнения буферов. Короче говоря, постоянно следите за всеми необычными явлениями. Такая постоянная подозрительность — иногда единственный способ заметить, что система ведет себя не так, как должна.

### ПРИМЕЧАНИЕ

### щщ

Распространенное место поиска свидетельств деятельности взломщиков — каталог `/dev`. Именно там часто помещают средства перехвата пакетов взломщики, использующие готовые средства из серии "rootkits" или сценарии. Однако, поскольку теперь ОС FreeBSD использует динамически генерируемую файловую систему DEVFS для специальных файлов устройств, этот каталог уже не вызывает такого беспокойства, но все равно не помешает за ним следить.

Если вы действительно подозреваете, что произошел взлом, и особенно если нашли свидетельства этого, надо предполагать, что ущерб — больше, чем кажется. Наиболее частой ошибкой администратора, обнаружившего свидетельство взлома защиты, является отключение ряда служб и предположение, что тем самым атака отбита. Часто так и бывает; однако если на все инциденты подобного рода реагировать именно так, недалеко и до катастрофы. Атакующему достаточно установить тот или иной "черный ход" в системе, который позволит ему вернуться и нанести более существенный ущерб, чем первоначально.

Если есть подозрение взлома системы, необходимо выполнить несколько действий.

1. Немедленно отключите систему от сети. Независимо от того, какие черные ходы установил атакующий, он ничего не сможет сделать, если система не подключена к сети. Это не даст взломщику, обнаружившему, что он выявлен, замести следы, стерев весь жесткий диск.
2. Проверьте, нет ли новых записей в таблицах `/var/cron/tabs` и `/etc/crontab`; проверьте также очередь `atq` на наличие заданий, оставленных взломщиком для выполнения в его отсутствие. Система может быть отключена от сети, но задания `atq` все равно будут работать, и взломщик может "добить" вашу систему таким способом, если не удалить все подозрительные задания.

3. Не пытайтесь связаться с атакующим или дать ему знать, что "вы его выявили". Хотя вы и вывели свою систему из-под удара, атакующий скроется, если поймет, что вы пытаетесь его выследить. Да и правоохранительным органам будет сложнее после этого его найти. Пусть он думает, что вы просто отключили машину для восстановления.
4. Соберите вместе журнальные файлы из каталога `/var/log` и из любых других каталогов, в которых их могут создавать используемые программы, а затем поищите в них записи, которые могут показать, откуда проник атакующий и как он получил доступ. Если в системе поддерживаются службы, для которых недавно вышли предупреждения об уязвимых местах в защите, и вы не обновили эти службы, чтобы устранить их уязвимые места, — почти наверняка через эти службы атакующий- и пробрался в систему.
5. Соберите всю полезную информацию, как указано на Web-сайте Национального центра защиты инфраструктуры (National Infrastructure Protection Center — NIPC): <http://www.nipc.gov> (или с сайта аналогичной национальной структуры по борьбе с компьютерными преступлениями, если вы живете не в США), и заполните заявление об инциденте. Получив от вас достаточно информации, ФБР сможет быстро выявить вторгшегося. Большая часть попыток взлома выполняется "вандалами-любителями" ("script kiddies") — случайными лицами, применяющими готовые, созданные другими средства, использующие ряд известных уязвимых мест. Взломщиков этого типа обычно легко находят и наказывают.
6. Создайте резервные копии важных данных — Web-документов, файлов конфигурации, начальных каталогов и всего содержимого каталога `/usr/local` — и переустановите операционную систему. Для большей надежности очистите предварительно жесткий диск, переустановите ОС FreeBSD "с нуля" и восстановите локальные данные. Используйте ежедневные отчеты системы Tripwire, чтобы понять, что необходимо пересоздать, и помните о "черных ходах", которые могут быть установлены вместе с остальными программами из каталога `/usr/local`.
7. Замените все службы самыми новыми версиями, просмотрев все соответствующие руководства по защите, прежде чем снова подключать систему к сети. Будьте особенно осторожны первые несколько дней после включения системы — могут быть повторные попытки взлома. Особенно внимательно в этот период просматривайте журнальные файлы; чем больше доказательств вы соберете, тем легче будет работать сотрудникам центра NIPC и ФБР.

## Атаки на службы (DoS)

Хотя технически эта проблема и не связана с защитой, но такая разновидность враждебной сетевой деятельности в последнее время требует все больше внимания системных администраторов. Речь идет об атаках на службы (Denial of Service или DoS).

Атаки на службы не связаны с изломом защиты системы или нарушением конфиденциальности. Это простая передача большого потока запросов, вследствие чего по сети посылаются так много пакетов, что в этом море теряются пакеты законных пользователей служб. Часто целью является остановка сервера, не справляющегося с огромным потоком данных и количеством запросов. От атак такого рода гораздо

сложнее защищаться, чем от прямых попыток взлома, которые можно отсечь с помощью систем IPFW, PortSentry и других описанных ранее средств. Атаки на службы нельзя предотвратить; можно только попытаться уменьшить их влияние, поскольку атаки задействуют только вполне допустимые пакеты, не отличающиеся от потока действительно нужных для работы данных. Проблема только в том, что их слишком много.

Иногда атака на службы оказывается исходящей из определенного источника, и ее можно заблокировать путем добавления правила брандмауэра, запрещающего прием пакетов с этого источника. Однако во многих последних атаках настоящий источник скрывался. Так, атаки путем отправки широкоэмитированных сообщений ping (ICMP) выглядят исходящими из определенного адреса, который на самом деле является адресом жертвы, подвергающейся главному удару. Распределенные атаки на службы, или атаки DDoS, являются еще более изощренными, вовлекая сотни и даже тысячи взломанных настольных компьютеров, ненамеренно участвующих в атаке, так что выявить настоящего виновника практически невозможно.

Определенные опции конфигурации различных серверных программ и ядра помогут системе остаться в рабочем состоянии в ходе атаки на службы. Сейчас мы рассмотрим некоторые из возможных мер. Помните, однако, что они могут только повысить шансы системы на выживание в ходе атаки на службы, но не могут нейтрализовать саму атаку или гарантировать, что атакующий не повторит попытки с большей интенсивностью, пока не добьется своего.

## Ограничение количества порождаемых серверных процессов

Многие атаки DoS направлены на службы типа Apache, Sendmail и другие, работающие путем "порождения" нового процесса для обработки каждого поступающего запроса. Если атакующий пошлет достаточно много запросов, в системе будет порождено столько процессов, что ресурсы процессора и памяти скоро исчерпаются и станет невозможной и стабильная работа системы. Ущерб от атак, провоцирующих порождение процессов, можно уменьшить, проверив, все ли подобные службы имеют встроенные ограничения на количество одновременно работающих процессов. Эти ограничения могут сказаться на способности сервера выполнять законные запросы в ходе обычной работы, но именно этот компромисс может спасти систему при атаке на службы.

Сервер Apache поддерживает директиву MaxClients со стандартным значением 150, не позволяющую обслуживать одновременно больше указанного количества запросов. Однако сложность в том, что продвинутый взломщик может вызывать интенсивно нагружающий процессор сценарий CGI до тех пор, пока сервер Apache не станет создавать процессы быстрее, чем они смогут завершаться, что приведет к сбою сервера гораздо быстрее, чем многочисленные запросы статических HTML-страниц. Часто единственным способом восстановления в этой ситуации, если система все же позволит зарегистрироваться с помощью Telnet или SSH, является остановка сервера Apache (apachectl stop) до завершения атаки. К счастью, большинство атак на службу HTTP можно проследить вплоть до конкретного IP-адреса клиента, заблокировав его с помощью правил системы IPFW или директивы deny from самого сервера Apache. Если это не поможет, можно уменьшить значение MaxClients, чтобы даже при его достижении клиенты не влияли на стабильность работы сервера.

Подобная возможность есть и в системе Sendmail: директива MaxDaemonChildren, отключенная по умолчанию. Ее можно включить, убрав перед ней символ комментария непосредственно в файле /etc/mail/sendmail.cf и перезапустив сервер (make restart). Кроме того, в системе Sendmail имеется встроенный "тормоз", предотвращающий запуск новых процессов при загрузке системы выше 12 процентов. Однако этот механизм срабатывает слишком медленно и не позволяет своевременно отреагировать на быстро развивающуюся атаку, так что может потребоваться явно ограничить количество порожденных процессов, поддерживаемых системой Sendmail в определенный момент времени. .\*\*

Потенциальным общим решением проблемы атак, провоцирующих порождение процессов, которое позволит защититься даже от тех, что исходят из самой системы (от недовольного или продажного пользователя, например), является изменение файла /etc/login.conf и установка в нем ограничений на использование пользователем процессора, оперативной памяти и количество одновременно открытых файлов. Создайте класс для пользователя, от имени которого работает соответствующая служба — nobody в случае сервера Apache, или отдельного локального пользователя, ресурсы которого необходимо ограничить, и отнесите пользователя к данному классу с помощью команды chfn. Вот пример такого класса в файле login.conf:

```
baduser:\
    :cputime=30m:\
    :openfiles=2 4:\
    :maxproc=32:\
    :memoryu8e=16m:\
    :tc=default:
```

Затем выполните команду `cap_mkdb /etc/login.conf` для включения этого нового класса в базу данных и установки ограничений для всех входящих в него пользователей.

## Защита от атак с плацдарма

Атака "с плацдарма" использует для достижения своих целей ресурсы сети жертвы. В то время как для атак грубой силой или путем порождения процессов необходимо использование больших вычислительных мощностей, при атаке с плацдарма атакующему достаточно послать хитро составленный набор запросов, чтобы сама инфраструктура сети жертвы стала наихудшим ее врагом. Например, атака путем широковещательной рассылки запроса ping (или атака "smurf") связана с посылкой атакующим обычного запроса ping на широковещательный адрес вашей сети (что распространяет запросы на все хосты сети), адрес отправителя в котором подделан и является адресом другого хоста — несчастной жертвы, страдающей при такой атаке намного больше вас, поскольку каждый хост вашей сети в ответ переполняет жертву ответами ping. Эффект умножения при атаке такого рода может обернуться для жертвы катастрофой, а организатора атаки будет очень сложно (или даже невозможно) выявить. Другая разновидность атаки с плацдарма запускает UDP-пакет между портами службы echo двух серверов, вызывая их вхождение в бесконечную "эхо-войну", остановить которую можно только отключением порта службы echo (что и сделано в ОС FreeBSD по умолчанию).

Атаки с плацдарма лучше всего предотвращать на уровне основного маршрутизатора сети. Атаки типа smurf можно предотвратить, сконфигурировав маршрутизатор так, чтобы он не отвечал на широковещательные запросы ping. Если в качестве мар-

шрутизатора используется машина с ОС FreeBSD, установка `icmp_bmcastecho="NO"` в файле `/etc/defaults/rc.conf` предотвращает ответы на такого рода запросы, которые практически никогда не используются с хорошей целью.

По умолчанию в стандартном ядре GENERIC скомпилирована опция `ICMP_BANDLIM`, ограничивающая частоту ответов на ICMP-сообщения об ошибках (еще один типичный вариант атаки с плацдарма), ограничивая тем самым эффективность подобных атак.

## Физическая защита

Даже если принять все возможные меры сетевой защиты, остается куда более существенная проблема защиты физической, от непосредственного доступа. Самая защищенная система в мире легко будет взломана, если неавторизованное лицо может получить физический доступ к серверу, поскольку никакая программная защита не спасет от взломщика с отверткой.

Безопасное размещение (*secure co-location facilities*) жизненно важно для коммерческого или любого высокоуровневого сервера Internet. Необходимы закрытые серверные стойки в закрытых для посторонних комнатах, причем открывать стойки и иметь доступ к машинам должны иметь право только сотрудники организации, обеспечивающей это безопасное размещение. Ваша система может иметь промышленное исполнение с закрытой передней панелью и средствами защиты BIOS, уведомляющими вас в случае снятия панели.

Любой человек, получивший физический доступ к машине, может перезагрузить ее в однопользовательский режим, в стандартной конфигурации которого пароль пользователя `root` не запрашивается. Это можно изменить, потребовав ввода пароля путем указания в файле `/etc/ttyx`, что консоль "небезопасна", т.е. что вы не можете гарантировать, что с нее обращается только авторизованный персонал:

```
console none          unknown off insecure
```

Однако это не мешает злоумышленнику загрузиться с дискеты или CD-ROM и взломать систему. Другие подключенные к машине устройства, например модемы или беспроводные сети, также могут использоваться для получения несанкционированного доступа, поэтому их не должно быть на машине, физический доступ к которой вы пытаетесь ограничить. Вывод: при отсутствии гарантированной физической защиты полная защита недостижима.

## Другие источники информации о защите

В этой главе рассмотрен ряд общих проблем защиты в применении к ОС FreeBSD. Однако тема сетевой защиты — обширна, и проблемы добавляются с каждым днем, поскольку все больше злонамеренных пользователей пытается найти способы остановить базовые службы сети Internet.

Есть ряд отличных источников информации по защите (как специфической для ОС FreeBSD, так и общей), с которыми имеет смысл ознакомиться.

### Страница справочного руководства `man security`

Составленная Мэтью Диллоном (Matthew Dillon), страница справочного руководства `man security` содержит описание общих проблем защиты и правильной практики администрирования, а также различные советы по предотвращению взломов и

атак на службы. Эта страница является базой для нескольких сетевых ресурсов, включая часть руководства FreeBSD Handbook.

## Списки рассылки

Подпишитесь на список рассылки [freebsd-security@freebsd.org](mailto:freebsd-security@freebsd.org). Для этого пошлите сообщение по адресу [majordomo@freebsd.org](mailto:majordomo@freebsd.org) с текстом `subscribe freebsd-security` в теле сообщения. Именно в этом списке рассылки обсуждаются наиболее актуальные проблемы защиты. Администратор должен быть в курсе последних достижений, чтобы прикрыть слабое место в защите сразу же после его обнаружения.

Еще один полезный список рассылки по защите, посвященный проблемам защиты UNIX вообще, — [Bugtraq](http://bugtraq.org). В этот список приходят рекомендации по всем основным проблемам, возникающим в защите Internet, иногда даже раньше, чем станет понятно их влияние на ОС FreeBSD. Список рассылки Bugtraq поддерживается на сайте <http://www.securityfocus.com>, где можно подписаться или выполнить поиск в архивах.

## Руководства по защите ОС FreeBSD

Руководства по защите (security advisories) рассылаются ответственным за защиту ОС FreeBSD (FreeBSD Security Officer) через списки рассылки [freebsd-announce@freebsd.org](mailto:freebsd-announce@freebsd.org) и [freebsd-security@freebsd.org](mailto:freebsd-security@freebsd.org) и предупреждают о новых выявленных уязвимых местах. Каждое такое руководство также архивируется в каталоге <http://www.freebsd.org/security/>.

Руководство содержит полное описание сути и влияния обнаруженного уязвимого места: находится ли оно в части базовой системы FreeBSD или в одной из портированных программ, специфична ли эта проблема для ОС FreeBSD, как обойти или решить проблему. Поскольку обсуждение сути проблемы до обнаружения ее решения было бы приглашением для взломщиков к началу атак, руководства выходят только после того, как найдено решение проблемы. Это еще одна веская причина подписаться на список рассылки [freebsd-security@freebsd.org](mailto:freebsd-security@freebsd.org), поскольку там уязвимое место обсуждается еще до выпуска руководства по его устранению.

Для устранения уязвимых мест в портированных приложениях или пакетах обычно достаточно обновить дерево каталогов и перестроить уязвимое приложение (см. главу 15). Исправление же в базовой системе FreeBSD, однако, обычно вносится в соответствующее дерево исходного кода (-STABLE или -CURRENT). Чтобы использовать его, необходимо перестроить соответствующую часть системы после обновления исходных кодов. Если исправление затрагивает достаточно фундаментальную часть системного кода, для обеспечения защиты системы может потребоваться полная перестройка, `make world`. Инструкции о том, как это сделать, представлены в главе 18.

## Web-ресурсы

Страница информации о защите FreeBSD, <http://www.freebsd.org/security/>, содержит ресурсы и ссылки, полезные для администратора ОС FreeBSD или разработчика. Тут же представлен архив руководств по защите, а также различные советы и рекомендации по устранению факторов риска.

Документ FreeBSD Security How-To (<http://people.freebsd.org/~jkb/howto.html>) предлагает описание различных методов защиты системы FreeBSD. В нем рассмотрены многие темы, затронутые в данной главе, и множество тем, которые здесь даже не упомянуты.

Сайт CERT (<http://www.cert.org>) — наиболее известный сайт по защите в Internet — содержит информацию об уязвимых местах в защите всех операционных систем и считается авторитетным источником предупреждений об уязвимых местах и информации о восстановлении. На сайте CEiRT также принимаются отчеты об инцидентах. Вы можете сообщить на сайт о взломе, и вас свяжут с соответствующими органами для поимки взломщика.

SecurityFocus — сайт, на котором поддерживается список рассылки Bugtraq. Это сайт новостей о защите, посвященный широкому спектру тем: от систем выявления вторжения до защиты от вирусов. Там также есть многочисленные статьи по эффективным приемам защиты и ответственному отношению к администрированию системы. Там не так уж много специфического материала по ОС FreeBSD, но большая часть информации применима к любой платформе. Адрес сайта — <http://www.securityfocus.com>.

## Книги

В файле `/etc/rc.firewall` рекомендуются две книги: "Брандмауэры и защита Internet" (*Firewalls & Internet Security*) Уильяма Чесвика (William R. Cheswick) и Стивена Беллоуина (Steven M. Bellowin) по общим проблемам защиты сетей, и "Построение брандмауэров Internet" (*Building Internet Firewalls, 2nd Edition*) Брента Чэпмса (Brent Chapman) и Элизабет Ивики (Elizabeth Zwicky) как более полное описание теории и практики использования брандмауэров.

Дополнительные книги и статьи по защите указаны и оценены с точки зрения полезности на сайте SecurityFocus в разделе "Library" ("Библиотека").