7

Организация защиты системы

В защищенной системе доступ к ресурсам компьютера открыт лишь авторизованным пользователям в дозволенных целях. Даже если в системе нет важных данных, большое значение имеют время процессора, память и полоса пропускания. Вообще многие администраторы, считавшие свои системы маловажными и не заботившиеся об их защите, обнаруживали, что их машины невольно стали ретрансляторами (relays) – плацдармом для атак, наносящих урон целым корпорациям. Кому хочется проснуться утром от восхитительных звуков, сопровождающих появление в вашем доме сотрудников правоохранительных органов, потому что ваш незащищенный компьютер был использован для проникновения в банк.

Конечно, есть вещи похуже, чем подростки, захватившие власть над серверами, – скажем, перелом обеих ног. Однако вторая по значимости неприятность – однажды прийти на работу и обнаружить, что веб-страница компании теперь сообщает: «Ха-ха, здесь был Вася!» Еще более масштабные вторжения вызывают куда больше головной боли. Большинство вторжений, с которыми мне приходилось иметь дело (не в качестве злоумышленника, а в качестве консультанта, нанятого пострадавшей стороной), исходили из стран, где доступ к Интернету находится под цензурой правительства, а анализ трафика показал, что злоумышленники лишь искали неограниченный доступ к новостным сайтам. Хотя я и сочувствую этим людям, но когда речь заходит о стабильности работы моих серверов, подобные вторжения я считаю недопустимыми.

К сожалению, за последние несколько лет стало намного проще взламывать удаленные компьютеры. Программы с «мышиным» интерфейсом, позволяющие громить компьютеры, легко можно найти, зайдя на поисковые машины. Стоит только одному способному взломщику написать программу, вскрывающую уязвимые места в защите, как тысячи скучающих подростков не находят ничего лучшего, как загрузить этот код

и усложнить жизнь некоторым из нас. Даже если вас не заботит сохранность ваших данных, системные ресурсы все равно надо защищать.

Вообще говоря, взламываются не операционные системы, а программы, работающие в них. Даже изначально безопасная операционная система, самая надежная в мире, не может защитить плохо написанные программы от них самих. Иногда проблемы, заложенные в таких программах, накладываются на особенности операционной системы и подвергают ее нешуточному риску. Чаще всего это выражается в *переполнении буферов*, когда программа злоумышленника попадает прямо в область памяти, из которой процессор берет команды для выполнения в данный момент, а операционная система выполняет этот код. Система FreeBSD прошла всестороннее тестирование и доработку, призванную исключить возможность переполнения буферов, однако нет никакой гарантии, что такая возможность полностью устранена. Новые функции и программы появляются каждый день, а их взаимодействие с устоявшимися функциями может быть непредсказуемым.

Система FreeBSD имеет большое число инструментальных средств, которые помогут обезопасить систему и сделать ее неуязвимой для нападающих, как извне, так и изнутри. Ни один из этих инструментов не является достаточным для обеспечения надежной защиты, поэтому было бы желательно использовать их все. Все, что вы узнаете о безопасности системы, следует воспринимать как один из инструментов, а не как решение всех ваших проблем. Например, простое повышение уровня безопасности системы не сделает ее полностью безопасной. Этот шаг поможет только в комбинации с расстановкой прав доступа, флагами файлов, регулярным наложением «заплат», внимательным отношением к паролям и другими приемами, составляющими общую политику безопасности. Расширенные способы обеспечения безопасности будут рассматриваться в главе 9, но без базовых приемов защиты, которые рассматриваются здесь, эти способы не смогут обезопасить систему.

Кто враг?

Прежде всего, я произвольно распределю потенциальных взломщиков на четыре группы: малыши со скриптами, бот-сети, недовольные пользователи и опытные взломщики. Более точную градацию взломщиков можно найти в книгах, посвященных безопасности, но такие подробности не относятся к теме данной книги. Эти категории легко объяснить, легко понять; под них подпадают 99% всех взломщиков, с которыми вам, вероятно, придется столкнуться.

Малыши со скриптами

Наиболее многочисленная группа взломщиков – *малыши со скриптами* (*script kiddies*). Малыши со скриптами – не системные администраторы. Они не искусны. Они загружают маленькие программы с «мыши-

ным» интерфейсом, предназначенные для организации атак, и ищут потенциальные цели. Малыши со скриптами похожи на грабителей, которые охотятся за кошельками старых леди, держащих свои сумки слишком свободно. К счастью, от них защититься легче всего; для этого достаточно своевременно обновлять систему и программы на сервере, ставить на них заплатки. Они как саранча, их легко раздавить, но их *слишком много*!

Бот-сети

Бот-сети состоят компьютеров, зараженных червями или вирусами. Авторы вирусов контролируют бот-сети и используют их для всего, чего угодно, начиная от поиска дополнительных уязвимых хостов и рассылки спама, и до вторжения на защищенные сайты. Подавляющее большинство бот-сетей состоят из компьютеров, работающих под управлением операционной системы Windows, но это не говорит, что компьютеры, работающие под управлением UNIX-подобных систем, не могут быть ассимилированы в бот-сети. Например, в операционной системе Solaris 10 демон telnet (in.telnetd) имел уязвимость, которая впоследствии эксплуатировалась червем, вызывавшим хаос и разрушение в домашних каталогах пользователей.

К счастью, защититься от бот-сетей ничуть не сложнее, чем от малышей со скриптами, – достаточно своевременно обновлять систему и следовать общепринятым рекомендациям.

Недовольные пользователи

Третья группа, вызывающая большую часть проблем безопасности, – это ваши пользователи. И правда, недовольные служащие пробивают большинство брешей в защите, потому что они как никто другой знают уязвимости в системе безопасности, считают, что правила к ним не относятся, и имеют время на взлом системы. Если вы сообщите сотруднику, что политика компании запрещает ему доступ к ресурсам компьютера, а сам сотрудник будет считать, что он должен обладать доступом к этим ресурсам, он наверняка начнет искать возможность обойти установленные ограничения. Любой, кто считает себя особенным, и считает, что к нему правила не имеют отношения, представляет риск для безопасности системы. Вы можете на всех ваших серверах своевременно устанавливать заплатки, задать человеконенавистнические правила фильтрации трафика, но если в коммутационном шкафу есть модем, который позволяет любому желающему, знающему пароль, войти в сеть, то ждите неприятностей.

Лучший способ остановить таких людей – не быть сентиментальным. Когда сотрудник покидает компанию, удалите его учетную запись, измените все административные пароли и сообщите всем служащим о его уходе и о том, что с ним больше не надо делиться конфиденциальной информацией. Выработайте политику обеспечения безопасности

с действенными наказаниями и неуклонно проводите ее в жизнь. Избавьтесь от незащищенного модема, недокументированного telnet-сервера, доступного на порту с нестандартным номером, и любого другого серьезного инструмента, в спешке размещенного там, где его якобы никто не сможет найти.

Опытные взломщики

Последняя группа действительно опасна: опытные взломщики. Это квалифицированные системные администраторы, исследователи систем безопасности и специалисты по преодолению защиты, желающие заполучить специфические ресурсы. Проникновение в компьютерные системы в наше время превратилось в доходный преступный бизнес, особенно если у жертвы имеются ресурсы, достаточные для проведения распределенных атак типа «отказ в обслуживании» (Distributed Denial of Service, DDoS) или массовой рассылки спама. Взлом веб-сайта, состоящего из нескольких компьютеров, и превращение его в инструмент злоумышленника оплачивается достаточно высоко. Если у вас имеется важная секретная информация компании, вы можете стать целью для одного из таких злоумышленников. Если один из них *действительно* захочет проникнуть в систему, он, вероятно, это сделает.

Тем не менее надлежащие мероприятия по обеспечению защиты, которые остановят первые три группы злоумышленников, могут заставить

Хакеры, злоумышленники и другие негодяи

Нередко можно услышать, как слово *хакер* употребляется для описания людей, которые взламывают компьютерные системы. Это слово имеет различные значения в зависимости от того, кто его произносит. В техническом мире хакером называют человека, который проявляет интерес к внутренней работе технических систем. Одни хакеры интересуются буквально всем, другие имеют более узкую область интересов. В сообществе FreeBSD *хакер* – это знак уважения. Основной технический список рассылки называется *FreeBSD-hackers@FreeBSD.org*. Если вам действительно интересен термин *хакер*, прочитайте статью в словаре жаргонных слов, которая доступна по адресу: *http://www.catb.org/jargon/html/H/hacker.html*.

Я рекомендую во избежание путаницы вообще не употреблять это слово. В этой книге взломщики систем будут именоваться *злоумышленниками*.[1] Технических кудесников можно величать по-разному, но они редко возражают против обращения «О, Великий и Могучий!»

[1] Лично я даю им гораздо менее приятные имена.

опытных взломщиков изменить тактику. Вместо того чтобы проникать в компьютеры по сети, они будут вынуждены прикидываться ремонтниками телефонной компании и тайком устанавливать анализатор пакетов либо разгребать мусор в поисках листков с записанными паролями. В этом случае им придется быть у всех на виду, и они могут посчитать, что проникновение в систему не стоит возможных неприятностей. Если вам удастся заставить злоумышленника составлять план взлома, напоминающий голливудский сценарий, *независимо от того, насколько полной информацией о вашей сети он обладает*, значит вам удалось достаточно высоко поднять уровень безопасности.

Сообщения, относящиеся к безопасности FreeBSD

Лучший способ остановить всех взломщиков – своевременно обновлять систему. Другими словами, надо знать, когда обновлять систему, что именно обновлять и как обновлять. Устаревшие системы – это лучшие друзья малышей со скриптами.

В Проекте FreeBSD есть команда разработчиков, специализирующихся на аудите исходного кода и обеспечении безопасности как базовой операционной системы, так и добавляемого программного обеспечения. Эти разработчики поддерживают почтовую рассылку с небольшим трафиком, *FreeBSD-security-notifications@FreeBSD.org*, на которую стоит подписаться. Общие уведомления появляются и в других почтовых рассылках (таких как BugTraq и CERT), но рассылка *security-notifications* представляет собою единственный источник информации о проблемах обеспечения безопасности, относящейся к FreeBSD. Порядок оформления подписки на эту рассылку описывается на странице *http://lists.freebsd.org/mailman/listinfo*. Группа безопасности FreeBSD публикует свои рекомендации в этой рассылке сразу же, как только они будут выработаны.

Внимательно читайте эти рекомендации и действуйте незамедлительно, если они имеют к вам отношение, так как малыши со скриптами уже вышли на поиск уязвимых систем. Лучшее, что можно сделать, – это ликвидировать проблему как можно быстрее.

Безопасность и пользователи

Помните, я говорил, что наибольшую опасность представляют ваши собственные пользователи? В этом разделе вы узнаете, как заставить этих маленьких детишек следовать установленным правилам. В операционной системе FreeBSD имеется большое разнообразие средств, которые позволят пользователям выполнять свою работу, но не дадут им безраздельно властвовать над системой. Здесь мы рассмотрим самые важные инструменты, начав с тех, которые позволяют добавлять новых пользователей.

Создание учетной записи пользователя

В операционной системе FreeBSD используются стандартные для UNIX программы управления пользователями, такие как passwd(1), pw(8) и vipw(8). Кроме того, FreeBSD включает в себя удобную интерактивную программу adduser(8), которая позволяет добавлять новых пользователей. Разумеется, создать новую учетную запись может только пользователь root. Просто введите в командной строке команду `adduser`, и вы попадете в интерактивную оболочку.

На первом запуске программа adduser(8) попросит установить соответствующие значения по умолчанию, которые будут использоваться при создании всех новых учетных записей. Следуя за примером сеанса работы с программой, который приводится ниже, вы сможете определить значения по умолчанию для своей системы.

```
# adduser
❶ Username: gedonner
❷ Full name: Gregory E Donner
❸ Uid (Leave empty for default):
```

Здесь username ❶ – это имя учетной записи. Имена учетных записей в моей системе составляются из инициалов и фамилии пользователя. Вы можете выбирать имена для учетных записей, исходя из любой другой схемы. Full name ❷ – это настоящее имя пользователя. Далее FreeBSD предложит выбрать числовой идентификатор пользователя (UID) ❸. Нумерация идентификаторов пользователей в системе FreeBSD начинается с числа 1000, хотя вы можете выбрать любое число. Все идентификаторы, лежащие ниже 1000, зарезервированы для нужд системы. В этом месте я рекомендую просто нажать клавишу Enter, чтобы выбрать первый незанятый UID.

```
❶ Login group [gedonner]:
❷ Login group is gedonner. Invite gedonner into other groups? []: webmasters
❸ Login class [default]:
❹ Shell (sh csh tcsh nologin) [sh]: tcsh
❺ Home directory [/home/gedonner]:
```

Группа пользователя по умолчанию ❶ – это очень важно, не забывайте, что права доступа в UNIX основаны на принадлежности владельцу и группе. По умолчанию FreeBSD создает для каждого пользователя отдельную группу, что наиболее предпочтительно для большинства систем. Во всех толстых книгах по системному администрированию предлагается несколько схем группировки – вы можете использовать ту из них, которая окажется наиболее близка к вашим потребностям. Помимо группы по умолчанию, пользователя можно добавить и в другие группы ❷, если в этом есть необходимость.

Login class (класс доступа) ❸ определяет, к каким ресурсам имеет доступ создаваемый пользователь. Подробнее о классах доступа мы поговорим немного ниже, в этом же разделе.

Shell (оболочка) ❹ – это интерпретатор команд. Хотя в системе по умолчанию выбирается интерпретатор */bin/sh*, я предпочитаю для начинающих пользователей выбирать интерпретатор команд tcsh.[1] Если у вас имеется глубокая привязанность к другому интерпретатору – выбирайте его. Опытные пользователи могут самостоятельно изменить выбор интерпретатора команд.[2]

Home directory (домашний каталог) ❺ – это каталог на диске, где будут находиться файлы пользователя. Владельцами этого каталога будут назначены сам пользователь и группа по умолчанию.

```
❶ Use password-based authentication? [yes]:
❷ Use an empty password? (yes/no) [no]:
❸ Use a random password? (yes/no) [no]: y
❹ Lock out the account after creation? [no]: n
```

Использование паролей дает определенную степень гибкости. Если все ваши пользователи знают, что такое SSH и общедоступные ключи, возможно, вам удастся обойтись без паролей. А пока большинство из нас предпочитают использовать пароли ❶.

Use an empty password (использовать пустой пароль) ❷ – эта возможность предусмотрена на тот случай, если необходимо, чтобы пользователь сам назначил себе пароль. Любому, кто попытается подключиться к системе с указанной учетной записью, будет предложено сначала установить пароль. Это делает идею оставить пароль пустым похожей на мысль заполнить дирижабль водородом, а потом зажечь спичку внутри него.

С другой стороны, отличным выбором будет назначение случайного пароля (random password) ❸ для новой учетной записи. Генератор случайных паролей в системе FreeBSD неплохо подходит для повседневного использования. Случайные пароли очень сложны для запоминания, и это лишний раз подталкивает пользователей поскорее изменить его.

Если учетная запись заблокирована (locked) ❹, никто не сможет использовать ее для входа в систему. Вообще говоря – это неэффективно.

После ввода всей необходимой информации программа adduser повторно выведет ее на экран, чтобы вы могли подтвердить или отменить

---

[1] Для интерактивной работы – да. Но никогда, никогда, *никогда* не используйте C-подобные командные интерпретаторы для написания программ. Прочитайте классический труд Тома Кристиансена (Tom Christiansen) «Csh Programming Considered Harmful», где вы найдете подробные объяснения. (Найти его можно по адресу: *http://www.faqs.org/faqs/unix-faq/shell/csh-whynot*)

[2] По умолчанию во многих версиях FreeBSD и так устанавливается */bin/tcsh*, хотя намного удобнее работать в bash, и поэтому после установки системы имеет смысл с помощью adduser установить bash в качестве интерпретатора команд по умолчанию для всех будущих пользователей. – *Прим. научн. ред.*

ее. Как только будет получено подтверждение, adduser проверит параметры учетной записи и выведет пароль, сгенерированный случайным образом. Затем программа предложит создать другую учетную запись.

Настройка adduser: /etc/adduser.conf

Процедура создания новых учетных записей в некоторых версиях UNIX сопряжена с необходимостью вручную редактировать файл */etc/passwd*, перестраивать базу паролей, редактировать файл */etc/group*, создавать домашний каталог, устанавливать права доступа к этому каталогу, устанавливать скрытые файлы (имена которых начинаются с символа «точка» (.)) и т. д. Все это вынуждает вас выработать свою собственную процедуру настройки – если все параметры устанавливаются вручную, вы легко можете управлять своими локальными учетными записями. adduser(8) сама выполняет большую часть рутинных операций и использует некоторые значения по умолчанию. Для серверов с различными требованиями вы можете определить различные значения по умолчанию в файле */etc/adduser.conf*, что позволит обеспечить соответствие предъявляемым требованиям и одновременно сохранить высокий уровень автоматизации.

Чтобы создать свой первый файл *adduser.conf*, нужно запустить команду `adduser -C` и ответить на ряд вопросов.

```
❶ Login group []:
❷ Enter additional groups []: cvsup
❸ Login class [default]:
❹ Shell (sh csh tcsh nologin) [sh]: tcsh
❺ Home directory [/home/]: /nfs/u1/home
❻ Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]: yes
Lock out the account after creation? [no]: no
```

Login group ❶ – это группа пользователя по умолчанию. Пустое значение означает, что в качестве группы по умолчанию пользователю будет назначаться его собственная группа (принято в FreeBSD по умолчанию).

Существует возможность указать дополнительные группы ❷, к которым по умолчанию будет принадлежать новая учетная запись, а также класс доступа ❸.

Укажите командный интерпретатор по умолчанию ❹ для своих пользователей.

Ваш выбор местоположения домашних каталогов ❺ пользователей может отличаться от устоявшегося стандарта, принятого в системе FreeBSD. В данном примере я указал, что домашние каталоги пользователей будут размещаться на разделе NFS, где находятся домашние каталоги пользователей, имеющих учетные записи на нескольких компьютерах.

В заключение определяется порядок установки пароля ❻ для нового пользователя.

Эти параметры по умолчанию помогут уберечь вас от ручного ввода с клавиатуры, но как только будет создан файл с базовыми настройками, у вас появляется возможность добавить в него более изощренные функции. В табл. 7.1 приводятся дополнительные параметры для файла *adduser.conf*, которые я считаю наиболее полезными.

Таблица 7.1. Полезные параметры настройки для файла adduser.conf

Параметр	Назначение
`defaultGroup`	Имя группы по умолчанию, к которой будут добавляться новые пользователи (если значение не определено, для каждого пользователя будет создаваться его собственная группа)
`defaultclass`	Класс доступа по умолчанию
`passwdtype`	Может иметь значения `no` (учетная запись останется заблокированной, пока root не назначит пароль), `none` (пароль не установлен), `yes` (пароль устанавливается при создании учетной записи) или `random` (будет назначен случайный пароль)
`homeprefix`	Каталог, где будут размещаться домашние каталоги пользователей (например, */home*)
`defaultshell`	Командная оболочка, назначаемая по умолчанию (здесь можно указать любую командную оболочку из */etc/shells*)
`udotdir`	Каталог, где находятся заготовки файлов пользователя, имена которых начинаются с символа «точка»
`msgfile`	Файл, содержащий текст электронного письма, отправляемого каждому пользователю сразу после создания учетной записи.

Используя эти параметры, вы сможете изменять поведение по умолчанию программы adduser, чтобы оно наиболее точно соответствовало вашим потребностям.

Изменение учетных записей: passwd(1), chpass(1) и другие

Управление пользователями заключается не только в создании и удалении учетных записей. Время от времени эти учетные записи необходимо изменять. Система FreeBSD включает в себя некоторые инструменты, предназначенные для редактирования учетных записей, самыми простыми из которых являются passwd(1), chpass(1), vipw(8) и pw(8). Все они работают с тесно взаимосвязанными файлами */etc/master.passwd*, */etc/passwd*, */etc/spwd.db* и */etc/pwd.db*. Для начала мы рассмотрим эти файлы, а затем перейдем к обзору указанных выше инструментов.

Файлы */etc/master.passwd*, */etc/passwd*, */etc/spwd.db* и */etc/pwd.db* хранят информацию об учетных записях пользователей. Каждый файл имеет свой формат и свое назначение. Файл */etc/master.passwd* является

источником информации для аутентификации и содержит пароли пользователей в зашифрованном виде. У обычных пользователей нет прав для просмотра содержимого файла */etc/master.passwd*. Однако им должна быть доступна основная информация, хранящаяся в учетных записях, иначе как непривилегированные программы смогут идентифицировать пользователя? В файле */etc/passwd* перечислены все учетные записи без привилегированной информации (например, без зашифрованных паролей). Содержимое этого файла доступно для чтения любому пользователю, оттуда он может извлекать основные сведения об учетной записи.

Информация из учетной записи требуется многим программам, а синтаксический анализ текстовых файлов, как известно, – процедура достаточно медленная. В наши дни мощных ноутбуков слово *медленная* теряет свой смысл, но это было насущной проблемой во времена, когда стиль диско шагал по Земле. По этой причине в системах BSD появились файлы базы данных, которые создаются на основе */etc/master.passwd* и */etc/passwd*. (Другие UNIX-подобные системы обладают похожей функциональностью, реализованной на основе других файлов.) Файл */etc/spwd.db* создается непосредственно из */etc/master.passwd* и содержит секретную информацию о пользователях, этот файл доступен для чтения только пользователю root. Файл */etc/pwd.db* доступен для чтения всем пользователям, но содержит ограниченный набор сведений, содержащихся в файле */etc/passwd*.

Всякий раз, когда какая-либо программа управления пользователями изменяет информацию об учетной записи, хранящейся в файле */etc/master.passwd*, FreeBSD запускает pwd_mkdb(8) для обновления трех других файлов. Например, все три программы, passwd(1), chpass(1) и vipw(8), позволяют вносить изменения в основной файл с паролями и все три программы вызывают pwd_mkdb(8) для обновления информации во взаимосвязанных файлах.

Изменение пароля

Для изменения пароля используется программа passwd(1). Пользователи могут изменять свои собственные пароли, а пользователь root имеет право изменять пароли любых пользователей. Чтобы изменить свой собственный пароль, достаточно просто ввести в строке приглашения к вводу команду `passwd`.

```
# passwd
Changing local password for mwlucas
Old Password:
New Password:
Retype New Password:
```

Если вы изменяете свой собственный пароль, passwd(1) сначала потребует ввести текущий пароль. Сделано это для того, чтобы никто другой не смог изменить пароль без ведома самого пользователя. Вообще, ко-

гда вы покидаете терминал, было бы желательно всегда завершать сеанс работы с системой, но даже если вы этого не делаете, эта простая проверка, которую проводит passwd(1), обезопасит вас от проделок шутников. После этого нужно будет дважды ввести новый пароль и все. Если вы обладаете привилегиями суперпользователя и вам требуется изменить пароль другого пользователя, просто передайте программе passwd имя этого пользователя в виде аргумента.

```
# passwd mwlucas
Changing local password for mwlucas
New Password:
Retype New Password:
```

Обратите внимание: пользователю root необязательно знать старый пароль другого пользователя – пользователь root может изменить учетную информацию любого пользователя в системе, как ему заблагорассудится.

Управление пользователями и переменная $EDITOR

Такие инструменты управления пользователями, как chpass и vipw (как и многие другие инструменты администрирования системы), запускают текстовый редактор, когда возникает необходимость внести изменения. Вообще говоря, эти программы определяют предпочтительную программу текстового редактора, исходя из содержимого переменной окружения `$EDITOR`. С помощью этой переменной можно назначить редактором по умолчанию vi, Emacs или какой-нибудь другой имеющийся редактор. Я рекомендую использовать Vigor (*/usr/ports/editors/vigor*), клон редактора vi(1) с анимированным помощником в виде скрепки, благодаря которому пользователи, привыкшие к Microsoft Office, будут чувствовать себя увереннее.

Изменение учетных записей с помощью chpass(1)

В учетной записи хранится не только пароль, но и масса другой информации. Утилита chpass(1) дает пользователям возможность редактировать все доступные для их учетной записи сведения. Например, если chpass запустить из командной строки, откроется текстовый редактор со следующим содержимым:

```
#Changing user information for mwlucas.
Shell: /bin/tcsh
Full Name: Michael W Lucas
Office Location:
Office Phone:
```

```
Home Phone:
Other information:
```

В данном случае мне разрешено редактировать шесть информационных полей, входящих в мою учетную запись. В первом поле, оболочка (Shell), можно установить любой командный интерпретатор из перечисленных в файле */etc/shells* (раздел «Интерпретаторы команд и /etc/shells» на стр. 244). Я могу изменить свое полное имя (Full Name), если, к примеру, мне захочется указать свое отчество полностью или я захочу, чтобы другие пользователи системы знали меня как *мистера Зануду*. Я могу изменить адрес своего офиса (Office Location) и номер рабочего телефона (Office Phone), по которым мои коллеги смогут меня отыскать. Это одна из особенностей, которая была полезна в университетских городках, где выросла BSD и где пользователи системы почти не знали о физическом местоположении других пользователей. Теперь, когда в нашем распоряжении имеются электронные каталоги и масса компьютеров, это поле перестало быть таким полезным. Обычно в поле с номером домашнего телефона (Home Phone) я указываю номер 911 (999 – в Великобритании), а в поле с прочей информацией (Other) могу внести некоторые дополнительные сведения. Кроме того, следует отметить, какую информацию я *не могу* изменять, будучи обычным пользователем. Местоположение моего домашнего каталога определяется системным администратором, и я не могу изменить его, даже если в систему был установлен новый жесткий диск огромного объема, который так пригодился бы для размещения моих файлов MP3. Аналогичным образом мои числовые идентификаторы пользователя и группы (UID и GID) присваиваются системой или системным администратором.

С другой стороны, если пользователь root запустит команду `chpass mwlucas`, его привилегированное положение позволит увидеть иную картину.

```
#Changing user information for mwlucas.
Login: mwlucas
❶ Password: $1$4d.nOPmc$uuBQy6ZL6hPQNTQef1jty.
Uid [#]: 1001
Gid [# or name]: 1001
Change [month day year]:
Expire [month day year]:
Class:
Home directory: /home/mwlucas
Shell: /bin/tcsh
Full Name: Michael W Lucas
Office Location:
Office Phone:
Home Phone:
Other information:
```

Как суперпользователь вы сможете сделать с бедным пользователем все, что угодно. Изменение имени учетной записи на *megaloser* – это только начало. Вы даже имеете доступ к зашифрованному паролю пользователя ❶. Не изменяйте его значение, если вы не в состоянии

производить шифрование в уме. Программа passwd(1) позволяет сделать это более безопасно. Вам будет даже позволено изменить местоположение домашнего каталога пользователя, при этом следует помнить, что chpass(1) не выполняет перемещение файлов пользователя – вам придется сделать это вручную.

Вы также можете установить дату изменения пароля и срок действия учетной записи. Срок действия пароля (Change) удобно определять в случаях, когда необходимо вынудить пользователя задать свой пароль при первом входе в систему. Срок действия учетной записи (Expire) удобно назначать, когда кто-то просит создать ему учетную запись на ограниченный период времени. Вы можете забыть вовремя удалить эту учетную запись, но FreeBSD никогда не забудет. В обоих случаях требуется указать дату в формате *месяц день год*, при этом следует указывать только первые три символа названия месяца. Например, чтобы пароль пользователя перестал действовать 8 июня 2008 года, я мог бы ввести в поле `Change` значение `Jun 8 2008`. Как только пользователь изменит свой пароль, поле срока действия пароля будет очищено, а что касается срока действия учетной записи, то только системный администратор сможет продлить его.

Большая кувалда: vipw(8)

Программу chpass удобно использовать для редактирования отдельных учетных записей, а как быть, если возникает необходимость отредактировать сразу множество записей? Предположим, что в системе имеются сотни пользователей, и в нее был установлен новый жесткий диск, предназначенный специально для размещения домашних каталогов, – неужели у вас хватит терпения запускать chpass(1) несколько сотен раз? Это тот случай, когда на помощь приходит программа vipw(8).

vipw позволяет редактировать непосредственно файл */etc/master.passwd*. По окончании внесения изменений vipw проверит синтаксис файла паролей, чтобы убедиться, что вы ничего не разрушили, затем сохранит обновленный файл паролей и запустит pwd_mkdb(8). vipw может защитить ваш файл паролей от массы досадных ошибок, но не стоит выпускать процесс из-под контроля. Чтобы должным образом использовать vipw(8), вам необходимо знать формат файла паролей.

Если информация в файле */etc/master.passwd* не совпадает с информацией в других файлах, программа предполагает, что верная информация находится в файле */etc/master.passwd*. Например, в файле */etc/group* отсутствуют группы, которые определены как основные группы пользователей. Группа, указанная в качестве основной в файле */etc/master.passwd*, считается правильной, даже если она отсутствует в файле */etc/group*.

Каждой учетной записи соответствует отдельная строка в файле */etc/master.passwd*, которая состоит из 10 полей, разделенных двоеточиями. Это следующие поля:

Имя учетной записи

Это или имя учетной записи, созданной системным администратором, или имя учетной записи, созданной во время установки какой-либо системной службы. Система FreeBSD включает такие имена учетных записей для нужд системного администрирования, как root, daemon, games и т. д. Каждая из таких учетных записей является владельцем некоторой части основной системы. В этот перечень входят также учетные записи для таких наиболее распространенных служб, как пользователь www, необходимый для работы веб-сервера. Дополнительное программное обеспечение может добавлять собственные учетные записи.

Зашифрованный пароль

Второе поле в строке – это пароль в зашифрованном виде. Системные пользователи не имеют пароля, поэтому вы не сможете войти в систему под одной из таких учетных записей. Для обычных пользователей это поле содержит случайные, на первый взгляд, последовательности символов.

Числовой идентификатор пользователя (UID)

Третье поле – это *числовой идентификатор пользователя*, или *UID*. Каждый пользователь имеет свой, уникальный UID.

Числовой идентификатор группы (GID)

Четвертое поле – это *числовой идентификатор группы*, или *GID*. В этом поле приводится идентификатор основной группы пользователя. Обычно он совпадает с UID, а сама группа имеет то же имя, что и учетная запись пользователя.

Класс доступа

Следующее поле – это класс доступа, определение которого находится в файле */etc/login.conf* (раздел «Ограничение на использование ресурсов системы» на стр. 255).

Срок действия пароля

Это то же самое поле срока действия пароля, что и в программе chpass(1), однако здесь время хранится в виде числа секунд, прошедших с начала времен. Для преобразования реальной даты в секунды можно использовать команду `date -j` с форматом вывода `+%s`. Чтобы перевести полночь 1 июня 2008 года в число секунд, прошедших от начала эпохи, можно воспользоваться командой `date -j 200806010000 '+%s'`.

Срок действия учетной записи

Чтобы заблокировать учетную запись в заданный день, нужно установить значение этого поля так же, как если бы это было поле, определяющее срок действия пароля.

Личные данные

Это поле известно также под названием *gecos* по малопонятным историческим причинам. Это поле содержит действительное имя пользователя, адрес офиса, рабочий телефон и домашний телефон. Все элементы этого поля отделяются друг от друга запятыми. Не используйте двоеточия внутри этого поля – двоеточия используются для разделения полей самой записи из */etc/master.passwd*.

Домашний каталог пользователя

Девятое поле – это домашний каталог пользователя. По умолчанию в это поле записывается значение в формате */home/<username>*, но вы можете указать любой другой каталог по своему усмотрению. Кроме того, при изменении этого поля вам потребуется вручную переместить файлы пользователя в новый каталог. Пользователи с несуществующим домашним каталогом по умолчанию не могут заходить в систему, хотя такое поведение можно изменить с помощью параметра настройки `requirehome` в файле *login.conf*.

Командный интерпретатор

Последнее поле – это командный интерпретатор пользователя. Если это поле оставить пустым, система будет предоставлять старый, скучный интерпретатор */bin/sh*.

Если программа chpass(1) дает вам возможность калечить отдельные учетные записи, то vipw(8) вверяет вам в руки всю базу данных о пользователях. Будьте внимательны при работе с ней!

Удаление пользователя

Учетные записи пользователей удаляются с помощью программы rmuser(8). Программа попросит ввести имя учетной записи, которую необходимо удалить, и спросит, следует ли удалять домашний каталог пользователя. Вот и все, что необходимо сделать, – разрушать всегда легче, чем созидать.

Использование pw(8) в сценариях

Команда pw(8) предоставляет мощный интерфейс командной строки для доступа к ученым записям пользователей. В то время как программа useradd(8) проведет вас по всем этапам создания учетной записи в диалоговом режиме, pw(8) позволит выполнить все необходимые действия одной-единственной командой. Я считаю, что pw(8) слишком громоздка для повседневного использования, но если вам приходится администрировать большое число учетных записей, ее помощь будет неоценима.

Я часто использую pw(8) для блокировки учетных записей. Хотя заблокированная учетная запись остается активной, никто не может войти с ней в систему. Я использую эту замечательную возможность, когда сумма на счете клиента опускается ниже нуля, – пользователи

очень быстро реагируют на отсутствие возможности войти в систему, даже при том, что их веб-сайты продолжают работать, а электронная почта продолжает поступать.

```
# pw lock mwlucas
```

Разблокировка учетной записи производится командой `pw unlock` *username*.

Если вам потребуется писать сценарии для управления своими пользователями, вам определенно стоит прочитать страницу руководства pw(8).

Интерпретаторы команд и /etc/shells

Интерпретатор команд (*shell*) – это программа, которая предоставляет в распоряжение пользователя командную строку. Различные интерпретаторы команд отличаются поведением, поддерживают различные комбинации клавиш для быстрого вызова и предоставляют различные функциональные возможности. Многие пользователи привязаны к определенным командным интерпретаторам и выражают свое неудовольствие, если любимый командный интерпретатор отсутствует в системе. Вы можете выполнить установку большого числа командных интерпретаторов из «портов» (глава 11).

Файл */etc/shells* содержит список командных интерпретаторов, доступных пользователю. При установке командного интерпретатора из «порта» или «пакета» в */etc/shells* будет добавлена соответствующая запись. Однако если не применять «порт», а компилировать исходный код, то в этот файл необходимо будет добавить полный путь к исполняемому файлу командного интерпретатора.

Демон FTP отклонит подключение пользователя, командный процессор которого не перечислен в */etc/shells*. Если вы используете */sbin/nologin* в качестве пользовательского командного процессора исключительно для доступа по FTP, то вы должны эту программу добавить в */etc/shells*. Однако более правильный путь состоит в том, чтобы управлять такими пользователями через классы доступа, о чем будет рассказываться далее в этой главе.

root, группы и права доступа

Система безопасности UNIX считается в некотором смысле грубой, поскольку один суперпользователь, *root*, может делать все, что угодно. Другие пользователи – безропотные пеоны, покорно носящие кандалы, в которые их заковал root. Проблема в том, что у суперпользователя не такой уж широкий выбор кандалов, и он не может подбирать их индивидуально для каждого пользователя. Это в общем так, но приличный администратор может комбинированием групп и прав доступа решать почти все вопросы безопасности.

Пароль root

Для некоторых действий нужен полный контроль над системой, в том числе возможность манипулировать ключевыми системными файлами, такими как файлы ядра, драйверов устройств и файлы системы аутентификации. Учетная запись root разработана как раз для выполнения таких действий.

Для использования пароля root надо либо войти в систему с консоли как root, либо, если вы – член группы wheel, выполнить команду смены пользователя (switch user) su(1). (Группы будут обсуждаться ниже, в этом же разделе.) Я рекомендую su; информация о пользователях, выполнивших su, заносится в протокол. Кроме того, su можно применять на удаленной системе. Выполнить эту команду очень просто:

```
# su
Password:
#
```

Проверьте текущее имя пользователя с помощью команды id(1):

```
# id
uid=0(root) gid=0(wheel) groups=0(wheel), 5(operator)
#
```

Теперь вы владеете системой – да, она *в вашей власти*. Будьте внимательны при каждом нажатии клавиши; небрежность может превратить жесткий диск в неформатированный кусок металла – такой же чистый, каким он был изначально. Если уж использовать пароль root, то ответственно, ибо каждый, кто знает пароль root, может причинить непоправимый вред системе.

Не забывайте, только пользователи из группы wheel могут воспользоваться паролем root, чтобы получить привилегии этого пользователя с помощью команды su(1). И любой может воспользоваться паролем root с консоли, вот почему так важно обеспечить физическую защиту системы. Если пароль root попадет в руки обычного пользователя, не имеющего физического доступа к консоли, он может вводить команду `su`, пока не надоест, – команда все равно работать не будет.

Естественно, возникает вопрос: «А кому нужен доступ с правами root?» Для большинства действий по конфигурированию, обсуждаемых в этой книге, необходим пароль root. Но как только система настроена должным образом, доступ к паролю root можно значительно сократить и даже приостановить. Для решения остальных задач, которые требуют привилегий пользователя root, я рекомендую использовать sudo (*/usr/ports/security/sudo*). Один из самых простых способов уменьшить необходимость передачи прав root состоит в правильном использовании групп.

Группы пользователей

В UNIX пользователи классифицируются по *группам*, в каждую из которых входят пользователи, выполняющие сходные административные функции. Системный администратор может определить группу с именем *webmasters*, добавить в нее учетные записи пользователей, редактирующих веб-страницы, и определить права доступа к соответствующим файлам так, чтобы члены группы могли редактировать эти файлы. Он может также создать группу *email*, добавить в нее учетные записи администраторов, управляющих почтовым сервером, и соответствующим образом определить права доступа к файлам, имеющим отношение к электронной почте. Такое использование групп представляет собой мощный инструмент управления системой.

Любой пользователь может определить свою принадлежность к группам с помощью команды id(1). В предыдущем примере видно, что пользователь root входит в состав групп wheel и operator. Однако root – это специальный пользователь, он может делать все, что пожелает. Ниже приводится моя учетная запись, более похожая на запись обычного среднего пользователя:

```
# id
uid=1001(mwlucas) gid=1001(mwlucas) groups=1001(mwlucas), 0(wheel),
68(dialer), 1006(cvsup)
```

Мой идентификатор пользователя (UID) равен 1001, а имя учетной записи – mwlucas. Идентификатор моей основной группы (GID) равен 1001, а моя основная группа также называется mwlucas. Это типичный пример первого созданного пользователя, но и последующие пользователи будут отличаться друг от друга только числовыми идентификаторами пользователя и группы. Гораздо интереснее узнать, для чего я назначил остальные группы: помимо своей основной группы я вхожу также в группы wheel, dialer и cvsup. Члены группы wheel могут использовать пароль root, чтобы получить привилегии root, члены группы dialer могут пользоваться программой tip(1) без необходимости получать права суперпользователя, а члены группы cvsup могут пользоваться репозиторием CVS в локальной системе. В моей системе каждая из этих групп имеет особые привилегии и как член этих групп, я наследую эти привилегии.

Информация о группах находится в файле */etc/group*.

/etc/group

Файл */etc/group* содержит информацию о всех группах, за исключением основных групп пользователей (которые определяются вместе с учетными записями в файле */etc/master.passwd*). Каждая строка в файле */etc/group* содержит четыре поля, разделенных двоеточиями: имя группы, пароль группы, числовой идентификатор группы и список членов группы. Ниже приводится пример записи:

```
wheel:*:0:root,mwlucas,gedonner
```

Имя группы – это понятное для человека название группы. Данная группа называется wheel. Имена групп могут быть совершенно произвольными. При желании группу можно назвать, например, minions (любимчики). Впрочем, имена для групп лучше выбирать, исходя из их предназначения. Вы можете запомнить, что члены группы minions могут редактировать веб-страницы, но догадаются ли об этом ваши коллеги?

Второе поле содержит пароль группы. Пароли групп способствовали укоренению плохих, с точки зрения безопасности, привычек, поэтому в большинстве современных систем UNIX они не поддерживаются. Однако прежние программы написаны в расчете на присутствие этого поля, поэтому вместо того, чтобы оставить это поле пустым или удалить его, используйте звездочку (*) в качестве «заполнителя».

Третье поле содержит уникальный числовой идентификатор группы (GID). Для идентификации групп в большинстве внутренних программ FreeBSD применяются именно GID, а не имена. Группа wheel имеет числовой идентификатор, равный 0, а максимальное значение GID равно 65 535.

Последнее поле – список всех пользователей, входящих в эту группу. Имена пользователей разделяются запятыми. В этом примере в состав группы wheel входят пользователи root, mwlucas и gedonner.

Изменение состава групп

Для добавления пользователя в группу добавьте имя его учетной записи в конец строки с определением требуемой группы. Например, в группу wheel входят пользователи, которые обладают правом пользоваться паролем root. В следующем примере я добавил пользователя rwatson в группу wheel:

```
wheel:*:0:root,mwlucas,gedonner,rwatson
```

Конечно, мои шансы убедить пользователя rwatson (президент фонда FreeBSD Foundation) взять на себя обязанности по администрированию любой из моих систем находятся в диапазоне от ничтожно малых до вообще никаких, но попробовать стоит.

Создание групп

Чтобы создать новую группу, достаточно иметь имя группы и числовой идентификатор группы. Технически вам даже не нужно иметь членов группы – некоторые программы работают с правами группы, и сама система FreeBSD использует привилегии группы для управления такими программами точно так же, как это делают пользователи.

Традиционно в качестве значения GID выбирается первое свободное число, превышающее максимальное значение GID в списке имеющихся групп. Вообще говоря, числовые идентификаторы групп со значе-

ниями ниже 1000 зарезервированы для нужд операционной системы. Программы, которым требуется отдельное значение GID, обычно используют одно из чисел в этом диапазоне. Значения идентификаторов групп для учетных записей пользователей начинаются с числа 1001. Идентификаторы некоторых специализированных групп начинают получать значения от 65 535 и ниже.

Использование групп во избежание передачи пароля root

Кроме проблемы обеспечения безопасности, передача пароля суперпользователя может вызвать разногласия в любой организации. Многие системные администраторы не торопятся передавать пароль root тем, кто отвечает за сопровождение отдельных частей системы, но, не предлагая альтернативы, они тем самым препятствуют тому, чтобы люди выполняли свои обязанности. Другие системные администраторы бесконтрольно раздают пароль root почти всем желающим, а потом жалуются, что система стала нестабильной. В конечном итоге оба варианта малопригодны. Будучи пользователем, я совсем не претендую на получение пароля root, но я настаиваю, чтобы системный администратор создал группу, которая обладала бы достаточными привилегиями для решения поставленных задач. Хотя это довольно удобно – обладать привилегиями суперпользователя, тем не менее не нести ответственность за нарушения в работе системы еще удобнее.

Одна из типичных ситуаций – когда младший системный администратор отвечает за некоторую часть системы. В моем подчинении было много администраторов DNS[1] – они никогда не устанавливали программное обеспечение, не пересобирали ядро и не выполняли другие обязанности системного администратора. Они лишь отвечали на электронные письма, обновляли файлы зон и перезапускали демон named. Начинающие системные администраторы часто считают, что им необходим пароль root для выполнения работ такого рода. Создав группы для тех, кто выполняет сходные административные функции, вы избежите бесконтрольной передачи пароля root и позволите людям выполнять свою работу. В этом разделе мы реализуем на основе групп управление доступом к файлам сервера имен. Те же самые принципы применимы к любым файлам, которые требуется обезопасить. Конфигурационные файлы веб-сервера и сервера электронной почты – другой наиболее вероятный кандидат для организации управления доступом на основе групп.

Системные учетные записи

В операционной системе FreeBSD некоторые учетные записи резервируются для встроенных программ. Например, сервер имен работает под учетной записью с именем bind и в группе bind. Не нужно входить

[1] Некоторым даже удалось выжить.

в систему под учетной записью программного пользователя, чтобы выполнять такого рода работу! Если злоумышленник взломает сервер имен, он получит доступ к системе всего лишь с привилегиями пользователя bind.

Более того, не допускайте, чтобы системные учетные записи и системные группы выступали владельцами файлов, созданных специально для таких программ. Создайте отдельного пользователя и группу, которые будут владеть файлами программ. В этом случае наш гипотетический взломщик сервера имен не сможет даже отредактировать файлы, которые используются сервером DNS, благодаря чему сводятся к минимуму возможные разрушения. Если программа регулярно обновляет файлы (например, файлы базы данных), то необходимо дать программе право на запись, но при этом вполне возможно, что человеку никогда не потребуется редактировать такие файлы. Точно так же нет никаких причин, по которым сервер базы данных должен иметь право на запись в свои собственные конфигурационные файлы.

Создание административных групп

Самый простой способ создать группу, которая будет владеть файлами, – это создать нового пользователя с помощью adduser(8) и затем использовать основную группу этого пользователя для назначения выбранным файлам привилегий этой группы. Так как у нас уже есть пользователь bind, мы создадим административного пользователя dns. Имя учетной записи не играет большой роли, но лучше выбирать такие имена, которые лучше запоминаются.

В качестве командной оболочки для этого пользователя укажите *nologin*, что соответствует интерпретатору */sbin/nologin*. Это предотвратит попытки войти в систему под этой учетной записью.

При желании для пользователей этой категории можно выбирать определенные значения UID и GID. Я выбираю такие значения UID и GID, которые напоминали бы числовые идентификаторы учетных записей, используемых соответствующими системными службами. Например, UID и GID пользователя bind имеют значение 53. Для простоты запоминания я мог бы присвоить UID пользователя dns значение 10053. В других случаях я присваиваю административным группам числовые идентификаторы, начиная с 65535 и далее вниз.

Не добавляйте таких административных пользователей в другие группы. И уж ни при каких обстоятельствах не добавляйте их в привилегированные группы, такие как wheel! Каждый пользователь должен иметь домашний каталог. Для административного пользователя вполне подойдет каталог с именем */nonexistent*. В конце концов, файлы этого пользователя находятся в другом месте. И, наконец, в программе adduser(8) нужно сделать учетную запись неактивной. Хотя выбранный командный интерпретатор предотвратит возможность входа, но дополнительная защита не помешает.

Теперь, после создания административного пользователя и группы можно назначить этого пользователя владельцем требуемых файлов. Каждому файлу поставлены в соответствие пользователь и группа, владеющие им. Увидеть права доступа к файлу и владельца можно с помощью команды `ls -l`. (Если вы не помните, как используются права доступа в UNIX, прочитайте страницы руководства ls(1) и chmod(1).) Многие системные администраторы уделяют самое пристальное внимание правам доступа для владельца файла, чуть меньше – для всех остальных и весьма поверхностное – правам доступа для группы.

```
# ls -l
total 3166
-rw-r----- 1 mwlucas mwlucas 79552 Nov 11 17:58 rndc.key
-rw-rw-r-- 1 mwlucas mwlucas 3131606 Nov 11 17:58 absolutefreebsd.com.db
```

Для этого примера были созданы два файла. Первый файл, *rdnc.key*, доступен для чтения и записи пользователю mwlucas, для чтения – всем, кто входит в состав группы mwlucas, все остальные не имеют никаких прав доступа к этому файлу. Файл *absolutefreebsd.com.db* доступен для чтения и записи пользователю mwlucas и всем, кто входит в состав группы mwlucas, все остальные имеют только право на чтение. Если вы входите в состав группы mwlucas, вы сможете редактировать файл *absolutefreebsd.com.db* и вам для этого совершенно не нужны привилегии пользователя root.

Изменить владельца и группу файла можно с помощью команды chown(1). При этом необходимо знать имя пользователя и группы, которые будут владеть файлом. В данном случае нам требуется, чтобы файлами владели пользователь dns и группа dns.

```
# chown dns:dns rndc.key
# chown dns:dns absolutefreebsd.com.db
# ls -l
total 3166
-rw-r----- 1 dns dns 79552 Nov 11 17:58 rndc.key
-rw-rw-r-- 1 dns dns 3131606 Nov 11 17:58 absolutefreebsd.com.db
```

Теперь этими файлами владеют пользователь dns и группа dns. Любой, кто входит в состав группы dns, сможет редактировать файл *absolutefreebsd.com.db*, не используя пароль root. Наконец, этот файл доступен для чтения серверу имен, работающему с привилегиями пользователя bind. Теперь стоит добавить администраторов DNS в группу dns, в файле */etc/group*, и они тут же смогут приступить к выполнению своих обязанностей.

Администраторы могут думать, что для перезапуска сервера имен им необходимо знать пароль root. Однако это легко решается с помощью rndc(8). Другие задачи могут решаться с помощью заданий в cron или с помощью программы sudo(8).

Интересные группы, создаваемые по умолчанию

В состав FreeBSD входит несколько групп, создаваемых по умолчанию. Большая часть из них используется системой и не требует к себе внимания системного администратора – вам достаточно знать об их существовании. Тем не менее здесь представлены самые полезные, интересные и необычные группы по умолчанию (табл. 7.2). Добавление собственных новых групп упрощает администрирование, однако примите к сведению, что в каждой системе FreeBSD есть группы, приведенные в этом списке.

Таблица 7.2. Интересные группы FreeBSD

Имя группы	Назначение
audit	Группа, обладающая правом доступа к информации audit(8)
authpf	Группа аутентификации фильтра пакетов PF
bin	Группа, владеющая основными двоичными файлами и программами в системе
bind	Группа для встроенного сервера DNS (глава 14)
daemon	Группа, используемая различными системными сервисами, например системой печати
_dhcp	Группа для выполнения операций с клиентами DHCP
dialer	Группа пользователей, имеющих доступ к последовательным портам; удобна для работы с модемами и программой tip(1)
games	Группа для игровых программ и файлов
guest	Группа для гостей системы (практически никогда не используется)
kmem	Группа для программ, обращающихся к памяти ядра, например fstat(1), netstat(1) и т. д.
mail	Группа для системы электронной почты (глава 16)
mailnull	Группа по умолчанию для Sendmail (глава 16)
man	Группа, владеющая страницами руководства
network	Группа, владеющая программами для работы с сетью, такими как ppp(8)
news	Группа для программ, предоставляющих доступ к Usenet (если установлено)
nobody	Основная группа для пользователя nobody, не имеющего никаких привилегий
nogroup	Группа без привилегий
operator	Группа, имеющая доступ к приводам; обычно используется для выполнения резервного копирования
_pflogd	Группа для ведения протокола PF
proxy	Группа для FTP-прокси в фильтре пакетов PF

Таблица 7.2 (продолжение)

Имя группы	Назначение
smmsp	Группа для Sendmail Submission User (глава 16)
sshd	Владелец сервера SSH (глава 15)
staff	Администраторы системы
sys	Системная группа для прочих нужд
tty	Группа для программ, имеющих право вывода на терминалы, таких как wall(1)
uucp	Группа для программ, использующих протокол UNIX-to-UNIX Copy Protocol
wheel	Пользователи, которые имеют право использовать пароль root
www	Группа для программ веб-сервера (но не для веб-файлов)

Настройка безопасности пользователей

Вы можете ограничить активность пользователей с целью предотвращения значительного потребления системных ресурсов, таких как память или процессорное время, каким-то одним пользователем. В наше время это не так важно, потому что сейчас даже небольшие компьютеры оснащаются быстрыми процессорами и большими объемами памяти, но это по-прежнему полезно в системах с десятками и сотнями пользователей. Кроме того, можно ограничить места, откуда пользователи смогут входить в систему.

Ограничение возможности входа

Всякий раз, когда пользователь пытается войти в систему, FreeBSD проверяет содержимое файла */etc/login.access*. Если там есть правила, запрещающие вход в систему того или иного пользователя, то его попытка зарегистрироваться немедленно провалится. По умолчанию этот файл пуст – нет никаких ограничений для пользователей, имеющих имя и пароль.

В файле */etc/login.access* есть три поля, разделенных двоеточиями. Первое поле предоставляет (+) или отнимает (-) право на вход в систему; второе поле – список пользователей или групп; третье – список источников подключений. Можно также использовать выражения `ALL` (все) и `ALL EXCEPT` (все, за исключением), позволяя администратору задавать простые, но выразительные правила. Программа входа в систему проверяет правила, руководствуясь первым совпадением, а не лучшим. Когда программа login(1) находит правило, которому соответствуют и группа, и источник подключения, соединение немедленно разрешается или отклоняется. Значит, порядок правил очень важен. Например, чтобы разрешить регистрацию с системной консоли только членам группы wheel, можно попробовать следующее:

```
+:wheel: console
```

Однако с этим правилом не все в порядке: на самом деле оно не отменяет полномочий пользователей на вход в систему. По умолчанию регистрация разрешена, а все, что делает это правило, – явно предоставляет право регистрироваться пользователям из группы wheel, поэтому данное правило не будет препятствовать регистрации других пользователей. Если я не вхожу в эту группу и попытаюсь зарегистрироваться с системной консоли, это правило не запретит мне доступ.

Можно было бы попробовать создать два правила:

```
+:wheel: console
-:ALL:console
```

Это даст желаемый эффект, но этот вариант можно упростить, если использовать выражение `ALL EXCEPT`.

```
-:ALL EXCEPT wheel: console
```

Это правило отклоняет подключения быстрее. Кроме того, снижается вероятность ошибки администратора. Как правило, при создании списков в *login.access* лучше запрещать регистрации, а не разрешать их. Как только система дойдет до этого правила, она немедленно откажет в регистрации с консоли пользователям, не входящим в группу wheel.

Последнее поле в файле *login.access*, источник подключения, может содержать имена хостов, адреса хостов, номера сетей, имена доменов или специальные значения `LOCAL` и `ALL`.

Имена хостов

Имена хостов опираются на DNS или файл *hosts*. Если есть подозрение, что сервер имен подвергается атакам взломщиков, не стоит использовать доступ по имени соединяющегося хоста; злоумышленники могут назначить имени хоста любой IP-адрес и обмануть вашу систему при подключении. Тем не менее можно предпринять следующее:

```
-:ALL EXCEPT wheel:fileserver.mycompany.com
```

Пользователи из группы wheel могут входить в систему с fileserver, а все остальные – нет.

Адреса хостов и сети

Адреса хостов похожи на имена хостов, но они невосприимчивы к манипуляциям с DNS.

```
-:ALL EXCEPT wheel:192.168.1.5
```

Номер сети – это усеченный IP-адрес:

```
-:ALL EXCEPT wheel:192.168.1.
```

Такое правило позволит каждому члену группы wheel регистрироваться с машины, IP-адрес которой начинается с 192.168.1, а также запретит доступ всем остальным.

LOCAL

Самое сложное местоположение – `LOCAL`. Ему соответствует любое имя хоста без точки (обычно это хосты в локальном домене). Например, *www.absolutefreebsd.com* предполагает, что любой хост в домене *absolutefreebsd.com* соответствует `LOCAL`. Такая функциональность реализована на основе метода обратного преобразования адресов в DNS (глава 14), который более уязвим для мистификаций. Мой ноутбук может заявить, что его имя – *humvee.blackhelicopters.org*, но его IP-адрес соответствует записи обратного DNS, утверждающей, что он находится где-то в сети моего поставщика услуг Интернета. Машина в домене *absolutefreebsd.com* подумает, что моему ноутбуку присвоено имя хоста, принадлежащее другому домену, а значит, он не является локальным. Таким образом, я не могу применять `LOCAL` в качестве метода проверки.

Аналогичным образом, любой, кто владеет блоком IP-адресов, сможет дать им любые имена в записях обратного DNS. Поэтому ограничение `LOCAL` не особенно полезно в реальных условиях.

ALL и ALL EXCEPT

Выражение `ALL` соответствует всем хостам, а `ALL EXCEPT` – всем, за исключением указанных далее. На мой взгляд – это наиболее полезные выражения при построении правил, ограничивающих источники подключения. Например, если необходимо обеспечить доступ к системе только с двух рабочих станций, можно записать такое правило:

```
-:ALL EXCEPT wheel:ALL EXCEPT 192.168.89.128 192.168.170.44
```

Собрать все вместе

Назначение этих правил заключается в обеспечении политики входа в систему, которая соответствует вашей политике безопасности. Если вы оказываете услуги общего характера, но удаленный доступ к системе разрешается только для системных администраторов, однострочный *login.access* позволит предотвратить попытки входа посторонних пользователей:

```
-:ALL EXCEPT wheel:ALL
```

Это замечательно, если с подобными ограничениями можно жить и работать. Однако мне случалось работать с несколькими поставщиками услуг Интернета, которые использовали FreeBSD. Обычным клиентам не разрешалось регистрироваться на серверах, если в их учетных записях не был указан командный интерпретатор. Системные администраторы, а также администраторы DNS и веб-сервера (члены групп dns и webmasters) могли выполнять вход с удаленных рабочих мест. Но с консоли могли выполнять вход только системные администраторы.

```
-:ALL EXCEPT wheel:console
-:ALL EXCEPT wheel dns webmasters:ALL
```

При наличии этой записи пользователи не смогут войти в систему, если их не добавить в группу, которой разрешен доступ.

Ограничение на использование ресурсов системы

Существует возможность организовать более точное управление с помощью классов доступа. Описания классов доступа находятся в файле */etc/login.conf* и определяют, какие данные и какие ресурсы могут предоставляться пользователям. Каждый пользователь закрепляется за определенным классом, и у каждого класса есть свои ограничения на доступ к системным ресурсам. Изменения ограничений для класса затрагивают всех пользователей, которые к нему принадлежат. Класс пользователя задается при создании его учетной записи. Изменить класс можно с помощью команды chpass(1).

Определения классов

По умолчанию *login.conf* начинается с класса default, который применяется для учетных записей, не относящихся к другим классам. По сути, этот класс предоставляет пользователям неограниченный доступ к системным ресурсам и больше подходит для приложений серверов с ограниченным числом пользователей. Если это соответствует вашим потребностям, не трогайте этот файл вообще.

Каждое определение класса состоит из последовательности операций присваивания значений переменным, которые определяют среду окружения пользователя, порядок учета и ограничения на системные ресурсы. Каждая запись в определении класса начинается и заканчивается двоеточием. Символ обратного слэша – это символ продолжения, он означает, что определение класса продолжается на следующей строке. За счет этого файл можно представить в удобочитаемом формате. Вот как может начинаться описание класса:

```
❶default:\
❷:passwd_format=❸md5:\
:copyright=/etc/COPYRIGHT:\
:welcome=/etc/motd:\
...
```

Этот класс называется `default` ❶. Я показал три из нескольких десятков переменных в этом классе. Например, переменной `passwd_format` ❷ присвоено значение `md5` ❸. Такие операции присваивания значений переменным и само имя класса составляют описание класса, и вы можете влиять на работу пользователя в системе, причисляя его к тому или иному классу.

Некоторые переменные *login.conf* не имеют значений; они изменяют свойства учетной записи своим наличием. Например, воздействие переменной `requirehome` определяется упоминанием ее в классе. Если эта переменная присутствует в определении класса, то для входа в систему пользователь обязан будет иметь домашний каталог.

```
:requirehome:\
```

После редактирования *login.conf* необходимо обновить базу данных login, чтобы изменения вступили в силу:

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

Эта команда перестроит файл базы данных */etc/login.conf.db*, который используется для ускорения поиска, как описанный ранее файл */etc/spwd.db*. По умолчанию файл */etc/login.conf* включает в себя несколько примеров классов пользователей. Чтобы понять, какие ограничения можно накладывать на пользователей в различных ситуациях, справляйтесь с этими примерами. В следующем разделе рассказывается о том, что можно устанавливать в классе доступа. Полный перечень доступных параметров в вашей версии FreeBSD можно найти на странице руководства login.conf(5).

Ограничения на ресурсы

Ограничения на ресурсы позволяют управлять объемом системных ресурсов, которые одновременно может задействовать один пользователь. Если к машине подключаются несколько сотен пользователей, а один из них решает скомпилировать OpenOffice.org, то этот пользователь может потреблять намного больше памяти и времени процессора, чем ему положено по справедливости. Если ограничить ресурсы, которые один пользователь может монополизировать одновременно, то система будет более отзывчивой к остальным пользователям.

В табл. 7.3 описаны переменные файла *login.conf*, отвечающие за ограничения ресурсов.

Таблица 7.3. Переменные login.conf для ограничения ресурсов

Переменная	Описание
cputime	Максимальное время процессора, которое может использовать любой процесс
filesize	Максимальный размер одного файла
datasize	Максимальный объем памяти, который может потреблять один процесс для хранения данных
stacksize	Максимальный объем стека, доступный одному процессу
coredumpsize	Максимальный размер дампа памяти
memoryuse	Максимальный объем памяти, который процесс может заблокировать
maxproc	Максимальное количество процессов, которые могут быть одновременно запущены одним пользователем
openfiles	Максимальное количество открытых файлов на один процесс
sbsize	Максимальный размер буфера сокета, который может задействовать приложение пользователя

Обратите внимание: зачастую ограничения на ресурсы привязаны к процессам по отдельности. Если каждый процесс получает 20 Мбайт памяти, а каждый пользователь может запускать 40 процессов, то тем самым вы разрешаете каждому пользователю получить 800 Мбайт памяти. Возможно, в вашей системе очень много памяти, но действительно ли так много?

Текущие и максимальные ограничения на ресурсы

Помимо ограничений, перечисленных выше, имеется возможность задать текущие (current) и максимальные ограничения на ресурсы. *Текущие* ограничения обычно носят рекомендательный характер, и пользователь может менять их по своему желанию. Это удобно в окружении, основанном на совместной работе, когда пользователи легко разделяют ресурсы, но вам необходимо предупреждать пользователей, превышающих стандартные ограничения. *Максимальные* ограничения абсолютны, и пользователь не может их раздвинуть.

Если не указывать тип ограничения, текущее или максимальное, система будет воспринимать его как максимальное.

Для задания текущего ограничения добавьте `-cur` к имени переменной. Для установления жесткого лимита добавьте `-max`. Например, для установления текущего ограничения на количество процессов, которые могут быть у одного пользователя, выполните следующее:

```
...
:maxproc-cur: 30:\
:maxproc-max: 60:\
...
```

Помимо наложения ограничений можно использовать функции учета ресурсов. Учет ресурсов не так уж важен сегодня по сравнению с временами, когда недорогой компьютер стоил десятки тысяч долларов, поэтому учет ресурсов в этой книге не рассматривается. Гораздо важнее иметь возможность накладывать ограничения на отдельного пользователя, чтобы не позволить ему потреблять больше процессорного времени, чем положено. Тем не менее полезно знать, что такая возможность существует.

Задание параметров среды по умолчанию

Существует возможность определять параметры среды в файле */etc/login.conf*. Это лучше, чем устанавливать их в пользовательских файлах *.cshrc* или *.profile*, поскольку настройки в *login.conf* окажут влияние на все учетные записи пользователей, подключающихся после задания этих параметров. Некоторые командные интерпретаторы, такие как zsh(1), не просматривают упомянутые конфигурационные файлы, поэтому описание среды в классе позволит правильно установить значения переменных окружения для пользователей, работающих с такими интерпретаторами.

Каждое поле окружения распознает два специальных символа. Тильда (~) замещается домашним каталогом пользователя, а знак доллара ($) – именем пользователя. Следующие несколько примеров из класса default иллюстрируют их использование:

```
:setenv=MAIL=❶/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 ❷~/bin:\
```

Например, переменной окружения MAIL присваивается значение */var/mail/<username>* ❶. Так же последний каталог в определении переменной PATH – это каталог *bin* в домашнем каталоге пользователя.

В табл. 7.4 перечислены типичные параметры настройки среды, которые используются в файле *login.conf*.

Таблица 7.4. Типичные параметры настройки среды в login.conf

Переменная	Описание
hushlogin	Если присутствует в определении класса, системная информация не выдается при входе в систему.
ignorenologin	Если присутствует в определении класса, пользователь может войти в систему, даже когда файл */var/run/nologin* существует.
ftp-chroot	Если присутствует в определении класса, пользователи при работе с FTP помещаются chroot-окружение (глава 15).
manpath	Список каталогов для переменной окружения $MANPATH.
nologin	Если присутствует, пользователь не может войти в систему.
path	Список каталогов для переменной окружения $PATH.
priority	Приоритет (nice) пользовательских процессов по умолчанию (глава 19).
setenv	Список переменных окружения, разделенных запятыми, с их значениями.
umask	Значение umask по умолчанию (см. builtin(1)). Это значение всегда должно начинаться с 0.
welcome	Полный путь к файлу, содержащему приветственное сообщение.
shell	Полный путь к командному процессору, который будет запущен после входа в систему. Эта запись подменяет командный процессор, указанный в */etc/master.passwd*. Однако переменная окружения $SHELL будет указывать на командный процессор, заданный в файле паролей, поэтому окружение будет противоречивым. Изменение значения этой переменной – замечательный способ раздосадовать пользователей.
term	Тип терминала по умолчанию. Чуть ли не каждая программа, пытающаяся установить тип терминала, подменяет эту запись.
timezone	Значение по умолчанию переменной окружения $TZ.

Управление свойствами пароля и входа в систему

В отличие от параметров среды окружения, которые можно настроить не только с помощью классов доступа, большая часть параметров настройки входа в систему и аутентификации могут выполняться только в определении класса доступа. Ниже приводятся некоторые параметры настройки аутентификации:

`minpasswordlength`

Задает минимальную длину пароля. Это значение возымеет эффект лишь при следующем изменении пароля пользователем; никакой проверки на соответствие уже установленных паролей этой длине не проводится. В следующем примере минимальная длина пароля устанавливается равной 28 символам (это лучший способ побудить пользователей использовать ключи SSH).

```
\:minpasswordlen=28:\
```

`passwd_format`

Задает алгоритм, применяемый для шифрования паролей в */etc/master.passwd*. Значение по умолчанию – `md5`. Другие допустимые значения – `des` (DES), `blf` (Blowfish) и `nthash` (Windows NT). DES наиболее полезен, когда необходимо иметь одинаковые пароли на компьютерах с различными операционными системами. Blowfish – очень ресурсоемкий алгоритм, но в современных системах процессорное время стоит дешевле грязи, поэтому его тоже вполне можно использовать. Алгоритм `nthash` очень удобен, когда *возникает потребность*, чтобы кто-то взломал вашу систему, иного предназначения этого алгоритма я не вижу.

`mixpasswordcase`

Если это свойство задано, то при следующем изменении паролей пользователи не смогут задавать в них только строчные буквы.

`host.allow`

Пользователи в классе с этим значением могут применять rlogin и rsh. Такая установка настоятельно не рекомендуется.

`host.deny`

Это значение используется при работе с rlogin и rsh. Избегайте их как несвежего мяса.

`times.allow`

Определяет промежуток времени, когда пользователь может входить в систему. Для задания времени необходимо в полях, разделенных двоеточиями, указать дни и периоды времени. Дни обозначаются двумя первыми буквами названия дня недели (Su, Mo, Tu, We, Th, Fr и Sa). Время указывается в стандартном 24-часовом формате. Например, если пользователю разрешено входить в систему только по средам между 8.00 и 17.00, подойдет такая запись:

```
        :times.allow=We8-17:\
```

`times.deny`

Определяет промежуток времени, когда пользователю нельзя входить в систему. Отметим, что эта запись не «выкинет» пользователя, если он уже вошел в систему. Формат записи такой же, как и в `times.allow`. Если `times.allow` и `times.deny` перекрываются, то `times.deny` имеет преимущество.

Флаги файлов

Все UNIX-подобные операционные системы обладают одними и теми же наборами прав доступа к файлам в файловой системе, такими как права на запись, на чтение и на исполнение файла для владельца файла, для группы и для всех остальных. Система FreeBSD расширяет схему прав, вводя *флаги файлов*. Вместе с правами доступа эти флаги усиливают защиту системы. Некоторые флаги используются для функций, которые не связаны с безопасностью, однако здесь будут рассматриваться флаги, имеющие к ней отношение. Полный список флагов файлов вы найдете в странице руководства chflags(1).

Многие флаги оказывают различное влияние в зависимости от уровня безопасности системы, кратко рассмотренного в следующем разделе. Для понимания уровней безопасности необходимо понимать флаги файлов, и при этом флаги файлов опираются на уровни безопасности. Пока кивайте головой и улыбайтесь при упоминании об этих уровнях; все станет ясно через несколько страниц.

Ниже представлены флаги, относящиеся к безопасности:

`sappnd`

Системный флаг «только добавление», который может устанавливать только root. В файл с этим флагом можно добавлять записи, однако удалять или редактировать этот файл нельзя (это особенно удобно для файлов протоколов). Если учетная запись каким-то образом «скомпрометирована», то установка флага `sappnd` файла *.history* может принести пользу. Тактика типичного злоумышленника состоит в том, чтобы удалить *.history* или сделать из него символическую ссылку на */dev/null*, чтобы администратор не узнал о предпринятых действиях. Благодаря установке флага `sappnd` малыши со скриптами не смогут замести следы таким образом. Забавно наблюдать за кем-либо, кто пытается удалить файл с флагом `sappnd`. Можно увидеть, как растет разочарование взломщика с каждой новой попыткой.[1] Этот флаг нельзя изменить, когда система работает на уровне безопасности 1 и выше.

[1] На самом деле это не так забавно, потому что злоумышленник все-таки проник в систему, но это хотя бы доставит краткий миг радости в тяжелый день.

`schg`

Системный флаг «неизменяемости», который может устанавливать только root. Файлы с установленным флагом `schg` нельзя изменять вообще: редактировать, перемещать, заменять. Сама файловая система пресекает все попытки коснуться этого файла каким-либо образом. Этот флаг нельзя изменить, когда система работает на уровне безопасности 1 и выше.

`sunlnk`

Системный флаг «запрет на удаление», который может устанавливать только root. Файл можно редактировать или изменять, но нельзя удалить. Этот флаг не настолько безопасен, как предыдущие два: если файл можно редактировать, то его можно очистить. Впрочем, это удобно при определенных обстоятельствах. Я использовал эту возможность, когда программа настаивала на удалении собственного файла протокола в случае аварии. Обычно установка любых стандартных флагов не приносит пользы. Этот флаг нельзя изменить, когда система работает на уровне безопасности 1 и выше.

`uappnd`

Пользовательский флаг «только добавление», который может устанавливать только владелец файла или root. Как в случае с установленным системным флагом `sappnd`, в файл с флагом `uappnd` можно добавлять записи, однако этот файл нельзя удалять или редактировать. Такая возможность наиболее полезна для протоколов, которые ведут личные программы пользователя. Установка такого файла предохраняет пользователей от их собственных ошибок. Владелец файла и root могут удалить этот флаг в любое время.

`uchg`

Пользовательский флаг «неизменяемости», который может устанавливать только владелец файла или root. Как в случае с флагом `schg`, описанным ранее, флаг «неизменяемости» защищает пользователя от изменения файла. Как и в предыдущих случаях, root может преодолеть действие этого флага. Кроме того, сам пользователь может убрать его, на каком бы уровне безопасности ни работала система. Этот флаг помогает предотвратить ошибки, но не обеспечивает безопасность системы.

`uunlnk`

Пользовательский флаг «запрет на удаление», который может установить только владелец файла или root. Файл с установленным флагом `uunlnk` владелец удалить не может, а root может. Пользователь может снять этот флаг в любой нужный момент.

Просмотр и установка флагов файлов

Флаги можно устанавливать командой chflags(1). Например, чтобы защитить ядро от подмены, можно сделать следующее:

```
# chflags schg /boot/kernel/kernel
```

Теперь никто не заменит ядро: ни злоумышленник, ни вы.

С помощью ключа `-R` можно рекурсивно изменять флаги, продвигаясь по дереву каталогов. Например, чтобы сделать каталог */bin* неизменяемым, воспользуйтесь такой командой:

```
# chflags -R schg /bin
```

Ура! Базовые исполняемые файлы теперь нельзя изменить.

Флаги файла можно увидеть с помощью команды `ls -lo`:

```
# ls -lo log
-rw-r--r--   1 mwlucas   mwlucas   sappnd 0 Nov 12 12:37 log
```

Символы `sappnd` здесь говорят о том, что флаг «только добавление» установлен на этот файл протокола. Для сравнения приведем строку в случае, когда у файла не установлены флаги:

```
# ls -lo log
-rw-r--r--   1 mwlucas   mwlucas   - 0 Nov 12 12:37 log
```

Дефис вместо имени флага говорит о том, что флаги файловой системы не были установлены.

В свежеустановленных системах FreeBSD лишь немногие файлы отмечены таким образом. Впрочем, флаги можно установить как угодно. На одной системе, считавшейся взломанной, я неистово запускал `chflags -R schg` в различных системных каталогах, чтобы не позволить никому заменить системные исполняемые файлы на троянские версии. Это не остановило бы взломщика, но я почувствовал себя лучше, представив, как бы он был расстроен, если бы добрался до командной строки.

Для снятия флага надо в команде `chflags` предварить имя флага словом `no`. Например, чтобы снять флаг `schg`, установленный на ядро, введите следующую команду:

```
# chflags noschg /boot/kernel/kernel
```

Надо сказать, что снятие этого флага подразумевает работу системы на уровне безопасности -1. Поэтому, не откладывая в долгий ящик, обсудим уровни безопасности.

Уровни безопасности

Уровни безопасности – это параметры настройки ядра, которые изменяют поведение базовой системы, запрещая те или иные действия. С повышением уровня безопасности ядро будет вести себя немного иначе. Например, на низких уровнях безопасности флаги файлов, рассмотренные выше, могут быть сняты. Например, флаг «неизменяемости» можно снять, отредактировать файл, а затем снова установить

флаг. На более высоких уровнях безопасности флаг файла не может быть снят. Подобные изменения имеют место в других частях системы. В целом, изменение поведения системы, вызванное повышением уровня безопасности, либо сорвет планы злоумышленника, либо остановит его. Уровень безопасности системы, вступающий в силу при начальной загрузке, можно задать с помощью параметра `kern_securelevel_enable="YES"` в файле *rc.conf*.

Уровни безопасности усложняют эксплуатацию системы, поскольку накладывают определенные ограничения на ее поведение. В конце концов, многие операции, выполняемые при обычном администрировании, может применить и злоумышленник, заметающий следы. Например, на некоторых уровнях безопасности невозможно отформатировать или смонтировать новый жесткий диск. С другой стороны, уровни безопасности препятствуют злоумышленнику больше, чем вам.

Установка уровней безопасности

Существует пять уровней безопасности: -1, 0, 1, 2 и 3, где -1 является низшим уровнем, а 3 – наивысшим. Как уже говорилось, разрешить использование уровней безопасности можно с помощью параметра `kern_securelevel_enable` в файле *rc.conf*. После этого уровень безопасности можно автоматически задавать при загрузке, указав его в переменной `kern_securelevel` файла *rc.conf*. Когда бы вы ни увеличили уровень безопасности системы, его нельзя снизить без перезагрузки в однопользовательском режиме. Если бы снизить уровень можно было без перезагрузки, этим воспользовался бы злоумышленник!

Уровень безопасности -1

Уровень безопасности -1, устанавливаемый по умолчанию, не подразумевает применения дополнительных средств защиты, предоставляемых ядром. Если вы изучаете FreeBSD и часто изменяете конфигурацию, оставьте уровень безопасности -1 и применяйте права доступа к файлам, встроенные в систему FreeBSD, а также другие меры безопасности UNIX, достаточные в большинстве ситуаций.

Уровень безопасности 0

Уровень безопасности 0 используется только в начале загрузки системы. Он не предлагает никаких специальных функций. Когда система переходит в многопользовательский режим, уровень безопасности автоматически увеличивается до 1. Установка `kern_securelevel=0` в */etc/rc.conf* эквивалентна установке `kern_securelevel=1`. Однако это может быть полезно, если во время загрузки системы запускаются сценарии, которые не могут выполнять необходимые действия на более высоких уровнях безопасности.

Уровень безопасности 1

На уровне безопасности 1 ситуация интереснее:

- Системные флаги файлов не могут быть сняты.
- Нельзя загружать и выгружать модули ядра (см. главу 5).
- Программы не могут записывать данные напрямую в системную память через устройства */dev/mem* или */dev/kmem*.
- Закрыт доступ к */dev/io*.
- На монтированные диски нельзя записывать данные напрямую, а значит, нельзя форматировать разделы. (Файлы можно записывать на диск через стандартный интерфейс ядра, нельзя лишь обращаться к диску, как к физическому устройству.)

Самое наглядное следствие уровня безопасности 1 в том, что нельзя изменять флаги файловой системы, специфичные для BSD. Если файл помечен системным флагом «неизменяемый», то заменить его не удастся.

Уровень безопасности 2

Уровень безопасности 2 предоставляет все преимущества уровня безопасности 1 с двумя добавлениями:

- Нельзя записывать данные напрямую в монтированные и немонтированные файловые системы.
- Системное время за раз можно изменить не более чем на 1 секунду.

Начинающий системный администратор может посчитать эти особенности несущественными, однако это важные приемы обеспечения безопасности. Хотя UNIX предоставляет удобные инструменты, например текстовые редакторы для записи данных в файл, без них можно обойтись. Точно так же можно обойтись и без файловой системы и получить прямой доступ к нулям и единицам, которые записаны на диске. В таком случае можно изменить любой файл вне зависимости от прав доступа. На практике такая возможность применяется лишь при установке нового жесткого диска. Обычно напрямую записывать данные на диск может только пользователь root. При уровне безопасности 2 это не позволяется даже ему.

Другой хакерский прием состоит в том, чтобы изменить системное время, отредактировать файл и установить прежнее время. Когда администратор ищет файлы, которые могли вызвать неполадки, искаженный файл выглядит так, словно к нему годами не притрагивались, а значит, не привлечет к себе пристального внимания.

Уровень безопасности 3

Уровень безопасности 3 называют *сетевым режимом безопасности*. Он проявляет себя точно так же, как уровень безопасности 2, но не допускает изменений правил пакетного фильтра. Правила брандмауэра невозможно изменить на этом уровне безопасности. Если в системе

включена фильтрация пакетов или управление полосой пропускания, а их правила хорошо настроены и вряд ли будут изменяться, то можно установить уровень безопасности 3.

Какой уровень безопасности выбрать?

Уровень безопасности, необходимый для той или иной операционной среды, зависит от ситуации. Например, если машина FreeBSD только что введена в эксплуатацию и ее требуется точно настроить, следует оставить уровень безопасности -1. По окончании настройки уровень безопасности можно увеличить. Для большинства систем вполне подходит уровень безопасности 2.

Если применяется одна из программ FreeBSD, предназначенных для фильтрации пакетов и организации брандмауэра, то стоит задуматься об уровне безопасности 3. Однако убедитесь в надежности правил брандмауэра, прежде чем выбрать уровень 3! Уровень безопасности 3 не позволит изменять конфигурацию брандмауэра без разрыва соединения с Интернетом. Вы должны быть уверены на все 100%, что ни один из ваших клиентов не скажет: «Я доплатил, удвойте мою полосу пропускания».

Когда уровни безопасности и флаги файлов не помогают?

Рассмотрим случай, когда взломщик находит «дыру» в сценарии CGI на веб-сервере Apache, получает через него доступ к командному интерпретатору (shell), а затем с его помощью получает права root.

Если уровень безопасности выбран правильно, планы взломщика, вероятно, сорвутся, так как он не сможет заменить ядро системы на свое, специально скомпилированное ядро. Тем не менее он сможет заменить множество системных исполняемых файлов на «троянских коней» с тем, чтобы при следующем входе в систему ваш пароль был отправлен на анонимный почтовый ящик с веб-интерфейсом или в сетевую телеконференцию.

Итак, для защиты своих ключевых файлов вы запускаете `chflags schg -R /bin/*, chflags schg -R /usr/lib` и т. д. Отлично. Если вы забыли про один файл, скажем, какой-нибудь непонятный */etc/rc.bsdextended*, то хакер сможет отредактировать его, добавив `chflags -R noschg /`. Позднее он может перезагрузить систему – ночью, когда вы этого не заметите. Как часто вы садитесь и основательно проверяете файлы */etc/rc*?

Вы думаете, что все файлы полностью защищены и система в безопасности. А что у вас в каталоге */usr/local/etc/rc.d*, где находятся локальные программы, запускаемые при начальной загрузке? В процессе загрузки система будет выполнять все файлы с расширением *.sh*, найденные в этом каталоге. Значит, гипотетический хакер может причинить немало вреда, разместив здесь простенький сценарий командного интерпретатора. В конце концов, */etc/rc* увеличивает уровень безопасно-

сти в последнюю очередь, когда все программы уже запущены. А что если хакер создаст сценарий, который убьет запущенный */etc/rc* перед тем, как он увеличит уровень безопасности, а затем запустит свой собственный */var/.hidden/rc.rootkit*, чтобы завершить подключение к сети?

Конечно, это только один путь, есть и другие, количество которых ограничивается только способностями взломщика. Важно помнить, что организация защиты системы – это нелегкая задача, для которой не существует простого решения. Как только взломщик получил доступ к командной строке, начинается схватка. И, если он имеет достаточно хорошую подготовку, о его появлении вы узнаете слишком поздно. Следуйте надежным рекомендациям по обеспечению безопасности и постоянно обновляйте систему, это самое первое средство предотвратить вторжения. Не позволяйте себе обленившись, положившись на уровни безопасности!

Жизнь с уровнями безопасности

Если вы использовали флаг `schg` направо и налево, вы скоро обнаружите, что установка обновлений или «заплат» стала неудобной. На самом деле, те условия, которые усложняют жизнь хакерам, могут сделать и вашу жизнь невыносимой, если не знать обходных путей. Итак, каковы же обходные пути?

Если файл */etc/rc.conf* защищен флагом `schg`, то для редактирования системных файлов сначала надо понизить уровень безопасности. Конечно, параметр, задающий уровень безопасности, находится в */etc/rc.conf*, поэтому для редактирования этого файла необходимо получить контроль над системой до запуска */etc/rc*. Для этого загрузите систему в однопользовательском режиме (как рассказывалось в главе 3), смонтируйте требуемые файловые системы, запустите `chflags noschg` на задействованных файлах и продолжите загрузку. Можно даже полностью отключить уровни безопасности в */etc/rc.conf* и потом спокойно работать. В этом случае процесс обновлений пойдет быстрее, но защита с помощью флагов файлов будет потеряна.

После внесения необходимых изменений уровень безопасности можно увеличить (но не уменьшить), не прибегая к перезагрузке. Для этого достаточно изменить параметр sysctl `kern.securelevel`, установив желаемый уровень безопасности.

```
# sysctl kern.securelevel=3
```

Теперь, когда вы можете управлять изменениями файлов, рассмотрим управление доступом к системе из сети.

Цели нападения из сети

Как правило, взламывается не операционная система, а программы, которые предоставляют возможность подключения из сети. Операци-

онная система может способствовать или не способствовать защите программ от сетевых атак, но вторжение всегда начинается с приложений. Один из способов сократить количество нападений на сервер состоит в том, чтобы составить полный перечень работающих программ, которые ожидают подключения из сети, и запретить запуск тех из них, которые не являются необходимыми. Операционная система FreeBSD содержит программу sockstat(1), которая предоставляет наиболее простой способ узнать, какие программы «прослушивают» сеть.

Программу sockstat мы уже рассматривали достаточно подробно в главе 6 – команда `sockstat -4` продемонстрирует все открытые порты TCP/IP IPv4. Каждый открытый сетевой порт можно рассматривать как потенциальную брешь и цель для нападения. Закройте все сетевые порты, ненужные для работы сервера, и обеспечьте защиту для тех, которые необходимы.

Список открытых портов следует просматривать постоянно, это поможет вам узнать кое-что, что может удивить вас. Например, вы можете с удивлением обнаружить, что некоторое программное обеспечение, установленное вами, имеет сетевой компонент, о существовании которого вы даже не подозревали, который преспокойно «прослушивал» сеть.

Но как отключить ненужные службы после того, как вы узнаете, что таковые действительно имеются? Лучший способ закрыть ненужные порты заключается в том, чтобы не запускать программы, которые их открывают. Обычно сетевые демоны запускаются из двух мест: либо из */etc/rc.conf*, либо из сценария запуска в */usr/local/etc/rc.d*. Программы, интегрированные в основную систему FreeBSD, такие как sendmail(8), sshd(8) и rpcbind(8), имеют флаги в */etc/rc.conf*, с помощью которых можно разрешать или запрещать запуск этих программ, а также многих других дополнительных приложений. Некоторые дополнительные программы, такие как веб-серверы, запускаются из сценариев

Безопасность серверов и рабочих станций

Во многих компаниях, которые мне приходилось видеть, уделяют очень много внимания безопасности серверов и практически не уделяют внимания безопасности рабочих станций. Однако для злоумышленника совершенно неважно, что взламывать, – сервер или рабочую станцию. На многих серверах и в системах сетевой защиты имеются специальные правила для рабочих станций системных администраторов. Злоумышленник с радостью взломает рабочую станцию и воспользуется ею для проникновения на сервер. Безусловно, безопасность сервера имеет очень большое значение, но не пренебрегайте защитой рабочих станций, особенно если это ваша рабочая станция!

в */usr/local/etc/rc.d*. Более подробно вопрос разрешения или запрета запуска программ во время загрузки рассматривается в главе 3.

Собрать все вместе

Когда в системе открыты только необходимые порты и ясно, какие программы их используют, то о защите этих программ и следует позаботиться. Если разработчики, обеспечивающие безопасность FreeBSD, отправляют извещение о неполадках сервиса, который вы не применяете, это сообщение можно спокойно проигнорировать. Если же речь идет о «дыре» в используемой программе, на это необходимо обратить внимание и как можно скорее внести необходимые исправления. Способность быстро реагировать на реальные опасности поможет вам защититься от большинства нападений. Такие инструменты, как флаги файлов или уровни безопасности, позволят минимизировать ущерб, который могут причинить взломщики. Наконец, использование групп для ограничения действий самих системных администраторов может защитить компьютеры от случайного или преднамеренного причинения вреда.