OBSD.RU



# Управление системой

printable page

## 10.1 - Почему мне сообщается, что я в неправильной группе, при попытке использования su root?

Существующие пользователи должны быть добавлены в группу "wheel" вручную. Это сделано из соображений безопасности и вы должны быть осторожны при предоставлении тому или иному пользователю возможности такой возможности. OpenBSD устроена так, что только пользователи, входящие в группу "wheel" могут использовать утилиту su(1) для того, чтобы стать суперпользователем (root). Пользователи, не включенные в данную группу, не могут использовать su(1). Ниже приводится пример того, как нужно изменить файл /etc/group для добавления пользователя **ericj** в группу "wheel".

Если вы заводите нового пользователя с помощью утилиты adduser(8), вы можете добавить его в группу "wheel" просто указав наименование группы в ответ на вопрос "Invite *user* into other groups:". Это добавит имя пользователя в файл /etc/group, примерно вот так:

```
wheel:*:0:root,ericj
```

Если вы хотите по-другому предоставить пользователям ограниченный доступ к правам суперпользователя без добавления их в группу "wheel", воспользуйтесь утилитой sudo(8).

## 10.2 - Как скопировать файловую систему?

Для копирования файловой системы используйте dump(8) и restore(8). Например, для копирования всего из каталога SRC в каталог DST, сделайте:

```
# cd /SRC; dump 0f - . | (cd /DST; restore -rf - )
```

dump также имеет массу возможностей для резервного копирования и, возможно, является избыточным инструментом, если все что вам нужно, это скопировать файловую систему частично или целиком. Команда tar(1) может быстрее справиться с этим. Её формат очень схож:

```
# cd /SRC; tar cf -  . | (cd /DST; tar xpf - )
```

# 10.3 - Каким образом запустить демон вместе с системой? (Обзор rc(8))

OpenBSD загружается посредством rc(8). При загрузке используются несколько основных файлов:

/etc/rc - Основной скрипт, изменению не подлежит.
/etc/rc.conf - Файл конфигурации для /etc/rc - каких демонов запускать вместе с системой.
/etc/rc.conf.local - Файл конфигурации, который можно использовать для изменения установок /etc/rc.conf - теперь можно не редактировать сам /etc/rc.conf что удобно тем, кто часто обновляет систему (rc сначала читает /etc/rc.conf, а потом /etc/rc.conf.local на предмет изменений. Если в новом релизе системы /etc/rc.conf изменяется, то при обновлении достаточно скопировать /etc/rc.conf.local из старой системы в новую - прим.перев).
/etc/netstart - Скрипт для инициализации сети, изменению не подлежит.
/etc/rc.local - Скрипт для администрирования локальной системы. Здесь хранится информация о новых демонах (нетипичных для типовой установки OpenBSD) и другая хост-специфичная информация.
/etc/rc.securelevel - Скрипт, исполняющий команды, которые должны быть выполнены до изменения уровня безопасности системы (см. Init(8))
/etc/rc.shutdown - Скрипт, исполняемый при выключении системы. Здесь размещаются команды, которые необходимо выполнить перед выключением, см. rc.shutdown(8)

## Как работает rc(8)?

Основными файлами, на которые должен обратить внимание администратор, являются */etc/rc.conf* (или */etc/rc.conf.local*), */etc/rc.local* и */etc/rc.shutdown*. rc(8) выполняется следующим порядком:

*/etc/rc* запускается после загрузки ядра:

Проверяются файловые системы.
Переменные конфигурации считываются из */etc/rc.conf* и, затем, из */etc/rc.conf.local*. Установки из *rc.conf.local* обладают большим приоритетом, чем из *rc.conf*.
Монтируются файловые системы
Чистится */tmp* и сохраняются все файлы редактора
Настраивается сеть посредством */etc/netstart*
- Настраиваются активные интерфейсы.
- Устанавливается имя хоста, домена и т.д.

Запускаются демоны системы
Производятся различные проверки (quotas, savecore и т.д.)
Запускаются местные демоны из */etc/rc.local*

## Запуск служб и демонов, поставляемых с OpenBSD

Большинство демонов и служб, поставляемых с OpenBSD по умолчанию могут запускаться при загрузке посредством простого редактирования */etc/rc.conf* . Для начала взглянем на исходный /etc/rc.conf. В нём есть строки, похожие на:

```
   ftpd_flags=NO
# for non-inetd use: ftpd_flags=" -D "
```

```
#(Для использования не через inetd использовать ftpd_flags=" -D ")
```

Такая строка показывает, что ftpd не запускается вместе с системой (по кр.мере, не через rc(8), см FAQ10 Настройка сервисов анонимного FTP. В любом случае, каждая строка имеет комментарий для **ОБЫЧНОГО** использования этого демона или службы. Это не означает, что запускать эти службы и демонов нужно именно с этими флагами. Из man(1) можно всегда узнать, как запустить нужную сложбу или демона любым удобным способом. Например, ниже приводится строка, соответствующая httpd(8):

```
   httpd_flags=NO
# for normal use: "" (or "-DSSL" after reading ssl(8))
```

Из этого ясно, что при нормальном запуске можно обойтись без флагов - достаточно строки **httpd_flags=""** А чтобы запустить httpd с поддержкой ssl (см FAQ10 Установка защищённого HTTP-сервера с SSL(8) или ssl(8)) понадобится "httpd_flags="-DSSL"".

Хорошим подходом является не изменять */etc/rc.conf*. Вместо этого создаётся файл */etc/rc.conf.local*, в который копируются только те строки из */etc/rc.conf* которые подлежат изменению. Это может облегчить будущее обновление - все изменения хранятся в одном файле. (Логика работы такова - если в строке в качестве значения указано не NO, то это значение передаётся соответствующему демону в качестве параметров. Прим.перев.)

## Загрузка локальных демонов и их конфигурация

Для остальных демонов, устанавливаемых в систему из портов или другим способом, следует использовать файл */etc/rc.local* Например, некоторый демон установлен в /usr/local/sbin/daemonx. Для запуска его при загрузке системы, в */etc/rc.local* нужно добавить следующий фрагмент:

```
   if [ -x /usr/local/sbin/daemonx ]; then
                echo -n ' daemonx';       /usr/local/sbin/daemonx
   fi
```

(Если демон не уходит в фоновый режим автоматически, в конце командной строки нужно добавить " &")

Теперь этот демон будет запускаться при загрузке системы - будут отображаться все сообщения об ошибках, при нормальной загрузке без ошибок будет выдано сообщение, похожее на:

```
   Starting local daemons: daemonx.
```

## rc.shutdown

*/etc/rc.shutdown* - скрипт, испольняемый при выключении системы. Сюда записываются все команды, которые должны быть выполнены перед выключением системы. При использовании apm, сюда можно добавить "powerdown=YES". Это

эквивалентно "shutdown -p".

## 10.4 - Почему пользователи получают "relaying denied" при попытке отправить почту удалённо через мою OpenBSD систему?

Команда:

```
# grep relay-domains /etc/mail/sendmail.cf
```

даёт результат, похожий на:

```
FR-o /etc/mail/relay-domains
```

В этом файле (если его нет, его нужно создать) записана информация о хостах, которым разрешена пересылка почты через данную систему. Синтаксис файла следующий:

```
.domain.com    #Разрешить пересылку от/для любого хоста в domain.com
sub.domain.com #Разрешить пересылку от/для sub.domain.com и любого хоста в этом домене
10.2           #Разрешить пересылку от любых хостов с IP-адресами из подсети 10.2.*.*
```

Не забудьте отправить сигнал 'HangUP' демону sendmail, (сигнал, заставляющих большинство демонов перечитывать файлы конфигурации):

```
# kill -HUP `head -1 /var/run/sendmail.pid`
```

### Более подробно:

http://www.sendmail.org/~ca/email/relayingdenied.html
http://www.sendmail.org/tips/relaying.html
http://www.sendmail.org/antispam.html

## 10.5 - Доступ по протоколу pop разрешён, но пользователи не могут получить свою почту через POP. Что делать?

Большинство проблем в отношении POP вызваны файлами блокировки и временными файлами. Если POP-сервер выдаёт сообщение об ошибке следующего содержания,

```
-ERR Couldn't open temporary file, do you own it?
```

Сначала нужно установить права доступа в /var

```
drwxrwxr-x   2 bin     mail     512 May 26 20:08 mail
```

Права в /var/mail

```
-rw-------   1 username   username        0 May 26 20:08 username
```

Затем, проверить, действительно ли пользователь владеет своим файлом в /var/mail . Например, файлом /var/mail/joe должен владеть пользователь с именем joe - в противном случае, проблемы неизбежны!

Действительно, право записи в каталог /var/mail, данное группе mail может быть угрозой безопасности системы. Но, скорее, это касается крупных систем, провайдеров интернет и т.д. Может быть, установлен pop-сервер не из дерева портов - если это возможно, рекомендуется использовать popa3d, имеющийся в базовой инсталляции системы OpenBSD. Могут быть выбраны неверные параметры запуска (например, параметры блокирования файлов). Может быть, нужно сменить каталог, в котором блокируются файлы (даже если блокирование было бы единственным выжным параметром pop-демона)

Замечание: Кстати, OpenBSD по умолчанию не имеет группы с именем "mail". Если нужно, она может быть создана добавлением строки в файл */etc/group*. Запись вида:

```
mail:*:6:
```

## 10.6 - Почему Sendmail игнорирует файл /etc/hosts?

По умолчанию, Sendmail использует DNS, а не /etc/hosts для разрешения имён. Такой порядок может быть изменён использованием файла /etc/mail/service.switch

Если желательно перед обращением к DNS искать в файле hosts, нужно создать файл /etc/mail/service.switch , содержащий следующую строку:

```
    hosts       files dns
```

Если желательно использовать ТОЛЬКО файл hosts:

```
    hosts       files
```

Изменеия вступят в силу после того, как Sendmail получит сигнал HUP:

```
# kill -HUP `head -1 /var/run/sendmail.pid`
```

## 10.7 - Установка защищённого HTTP-сервера с SSL(8)

OpenBSD поставляется с готовым к использованию SSL httpd-сервером и библиотеками RSA. Для их использования совместно с httpd(8) нужно создать (получить) сертификат. Он будет храниться в */etc/ssl/*, а соответствующий секретный ключ - в */etc/ssl/private/*. Приводимые ниже инструкции взяты из руководства к ssl(8). Там же может быть найдена более подробная информация. Эти ЧаВо только в общих чертах описывают процесс создания сертификата RSA для web-сервера. Для инструкций по созданию сертификата DSA, следует обращаться к руководству по ssl(8).

Для начала, нужно создать ключ сервера и сертификат с помощью OpenSSL:

```
# openssl genrsa -out /etc/ssl/private/server.key 1024
```

Или, если нужно чтобы ключ шифровался идентификационной фразой, которая будет вводиться каждый раз при запуске сервера

```
# openssl genrsa -des3 -out /etc/ssl/private/server.key 1024
```

Следующий шаг - создание запроса на подпись сертификата (Certificate Signing Request), который направляется в Орган Сертификации (Certifying Authority, CA) для подписи сертификата. Делается это следующей командой:

```
# openssl req -new -key /etc/ssl/private/server.key -out /etc/ssl/private/server.csr
```

Полученный файл *server.csr* может быть передан Орган Сертификациидля подписи ключа. Например, в **Thawte Certification** по адресу http://www.thawte.com/.

Если такой вариант не подходит, можно подписать сертификат самому следующей командой:

```
# openssl x509 -req -days 365 -in /etc/ssl/private/server.csr \
      -signkey /etc/ssl/private/server.key -out /etc/ssl/server.crt
```

Когда */etc/ssl/server.crt* и */etc/ssl/private/server.key* будут на месте, можно запустить httpd(8) с флагом **-DSSL** (см. о запуске демонов вместе с системой), который будет обслуживать https-запросы на порту 443.

## 10.8 - Я изменил /etc/passwd , но изменений не заметно. Почему?

Если изменить непосредственно */etc/passwd* изменения будут потеряны. OpenBSD создаёт */etc/passwd* динамически с помощью pwd_mkdb(8). Главный файл паролей OpenBSD - */etc/master.passwd*. Согласно руководству по pwd_mkdb(8), файлы имеют следущие значения

```
FILES
    /etc/master.passwd  текущий файл паролей
```

```
/etc/passwd          файл паролей в стиле 6-й версии (UNIX)
/etc/pwd.db          незащищённый файл базы данных паролей
/etc/pwd.db.tmp      временный файл
/etc/spwd.db         защищённый файл базы данных паролей
/etc/spwd.db.tmp     временный файл
```

В традиционном файле паролей UNIX, таком как /etc/passwd, всё, включая зашифрованные пароли пользователей доступны всем в системе (и являются основной целью программ-взломщиков). 4.4BSD ввела в обращение файл master.passwd, который имеет расширенный формат (с дополнительными, по сравнению с /etc/passwd, параметрами) с правом чтения только пользователем root. Для ускорения доступа к данным, вызовы библиотек читают их обычно из /etc/pwd.db и /etc/spwd.db.

OpenBSD поставляется со специальным редактором для файла паролей. Он называется vipw(8). Vipw использует vi (или выбранный редактор, определённый в $EDITOR) для редактирования /etc/master.passwd. После редактирования, изменения автоматически учитываются в новых /etc/passwd, /etc/pwd.db, и /etc/spwd.db . Vipw также блокирует эти файлы, так что любому, кто в то же время попытается изменить информацию, будет отказано в доступе.

## 10.9 - Как лучше добавлять и удалять пользователей?

OpenBSD предоставляет две команды для лёгкого добавления пользователей в систему:

adduser(8)[/url]
user(8)

Также можно добавить пользователей вручную с использованием vipw(8), но это сложнее в большинстве случаев. Простейший способ добавить пользователя в OpenBSD - использование скрипта adduser(8). Настроить его можно редактированием файла */etc/adduser.conf*. *adduser(8)* проверяет на целостность */etc/passwd*, */etc/group*, и базы данных оболочки. Он создаёт сопутствующие файлы и каталог $HOME. Он может даже отправить приветственное сообщение пользователю. В приводимом ниже примере в систему добавляется пользователь с именем **testuser**, для него выделяется домашний каталог $HOME в */home/testuser*, он включается в группу **guest** и ему предписывается оболочка */bin/ksh*.

```
# adduser
Use option ``-silent if you don't want to see all warnings and questions.
Reading /etc/shells
Reading /etc/login.conf
Check /etc/master.passwd
Check /etc/group
Ok, let's go.
Don't worry about mistakes. I will give you the chance later to correct any input.
Enter username []: testuser
Enter full name []: Test FAQ User
Enter shell csh ksh nologin sh [sh]: ksh
Uid [1002]: Enter
Login group testuser [testuser]: guest
Login group is ``guest. Invite testuser into other groups: guest no
[no]: no
Login class auth-defaults auth-ftp-defaults daemon default staff
[default]: Enter
Enter password []: Type password, then Enter
```

```
Enter password again []: Type password, then Enter
Name:        testuser
Password:    ****
Fullname:    Test FAQ User
Uid:         1002
Gid:         31 (guest)
Groups:      guest
Login Class: default
HOME:        /home/testuser
Shell:       /bin/ksh
OK? (y/n) [y]: y
Added user ``testuser
Copy files from /etc/skel to /home/testuser
Add another user? (y/n) [y]: n
Goodbye!
```

Для удаления пользователя используется команда rmuser(8). Она удаляет пользователя и все следы его пребывания в системе - записи crontab(1), каталог $HOME (если он принаждлежит пользователю), почтовый ящик. Естественно, она удаляет и записи в */etc/passwd* и */etc/group*. Ниже приводится пример удаления пользователя, добавленного в предыдущем примере. При этом запрашивается имя и удалять ли домашний каталог пользователя.

```
# rmuser
Enter login name for user to remove: testuser
Matching password entry:
testuser:$2a$07$ZWnBOsbqMJ.ducQBfsTKUe3PL97Ve1AHWJ0A4uLamniLNXLeYrEie:1002
:31::0:0:Test FAQ User:/home/testuser:/bin/ksh
Is this the entry you wish to remove? y
Remove user's home directory (/home/testuser)? y
Updating password file, updating databases, done.
Updating group file: done.
Removing user's home directory (/home/testuser): done.
```

## Добавление пользователя посредством user(8)

Эти инструменты гораздо менее интерактивны, чем adduser(8), что облегчает их использование в скриптах и программах.

Полный набор состоит из:

- group(8)
- groupadd(8)
- groupdel(8)
- groupinfo(8)
- groupmod(8)
- user(8)
- useradd(8)
- userdel(8)
- userinfo(8)

usermod(8)

## Непосредственное добавление пользователей.

Поскольку user(8) не интерактивна, простейший и рациональнейший способ добавить пользователя - использование команды adduser(8). Сама команда /usr/sbin/user является лишь интерфейсом к командам /usr/sbin/user* . Следовательно, следующая команда может быть вызвана как **user add** или **useradd** - выбор не влияет на выполнение задачи.

В приводимом ниже примере добавляется пользователь с тем же набором параметров, что и в примере с adduser(8) выше. useradd(8) гораздо легче в использовании, если параметры по умолчанию известны заранее. Эти параметры хранятся в */etc/usermgmt.conf* и могут быть выведены командой:

```
$ user add -D
group           users
base_dir        /home
skel_dir        /etc/skel
shell           /bin/csh
inactive        0
expire          Null (unset)
range           1000..60000
```

Приведённые параметры будут использованы если противное не задано параметрами командной строки. Например, в приводимом случае, пользователь должен быть включён в группу **guest**, а не **users**. Ещё одна сложность в добавлении пользователей таким образом заключается в том, что пароль должен быть задан в командной строке. Пароль зашифрован, значит, сначала нужно использовать команду encrypt(1) для его создания. Например, пароли в OpenBSDпо умолчанию используют алгоритм Blowfish с 6 проходами. Ниже приводится пример командной строки для создания зашифрованного пароля для использования с useradd(8).

```
$ encrypt -p -b 6
Enter string:
$2a$06$YOdOZM3.4m6MObBXjeZtBOWArqC2.uRJZXUkOghbieIvSWXVJRzlq
```

Теперь можно создавать пользователя.

```
# user add -p '$2a$06$YOdOZM3.4m6MObBXjeZtBOWArqC2.uRJZXUkOghbieIvSWXVJRzlq' -u 1002 \
-s /bin/ksh -c "Test FAQ User" -m -g guest testuser
```

**Внимание**, строка пароля даётся в одиночных кавычках ' ', а не в двойных (" ") так как оболочка подвергает это обработке перед тем как передать user(8). В добавок, нужно не забыть указать параметр **-m** если нужно создать домашний каталог пользователя и скопировать туда файлы из */etc/skel* .

Для проверки правильности создания пользователя можно использовать множество инструментов. Ниже приводятся несколько команд, которые помогут быстро проверить, всё ли сделано правильно:

```
$ ls -la /home
```

```
total 14
drwxr-xr-x   5 root      wheel   512 May 12 14:29 .
drwxr-xr-x  15 root      wheel   512 Apr 25 20:52 ..
drwxr-xr-x  24 ericj     wheel  2560 May 12 13:38 ericj
drwxr-xr-x   2 testuser  guest   512 May 12 14:28 testuser
$ id testuser
uid=1002(testuser) gid=31(guest) groups=31(guest)
$ finger testuser
Login: testuser                         Name: Test FAQ User
Directory: /home/testuser               Shell: /bin/ksh
Last login Sat Apr 22 16:05 (EDT) on ttyC2
No Mail.
No Plan.
```

Вдобавок к этим командам, user(8) предоставляет интрумент для просмотра свойств пользователя userinfo(8).

```
$ userinfo testuser
login   testuser
passwd  *
uid     1002
groups  guest
change  Wed Dec 31 19:00:00 1969
class
gecos   Test FAQ User
dir     /home/testuser
shell   /bin/ksh
expire  Wed Dec 31 19:00:00 1969
```

### Удаление пользователей

Для удаления пользователя командами из серии user(8), используется команда userdel(8). Это очень простая, но, тем не менее, удобная команда. Для удаления пользователя, добавленного в прошлом примере, используется просто:

```
# userdel -r testuser
```

Кстати, нужно указать параметр -r для удаления домашнего каталога пользователя. Или можно указать параметр -p а не -r и учётная запись пользователя будет заблокирована без удаления информации.

## 10.10 - Как создать учётную запись для доступа только по ftp (не анонимный FTP!)?

Есть несколько способов сделать это, но самым распространённым является добавить "/usr/bin/false" в "/etc/shells". Если затем установить пользователю оболочку "/usr/bin/false", он не сможет войти в интерактивном режиме, но сможет пользоваться доступом по ftp. (это же справедливо и для других типов ограниченных учётных записей - например, только почтовых и т.д.). Может быть уместно ограничить доступ пользователя по ftp его домашним каталогом.

# 10.11 - Установка квот.

Квоты используются для ограничения дискового пространства, доступного пользователю. Это может быть очень полезно в условиях ограниченных ресурсов. Квоты могут быть установлены для пользователей и/или для групп.

Первый шаг в установке квот - убедиться в наличии параметра "option QUOTA" в конфигурации ядра. Этот параметр присутствует в ядре GENERIC. После этого нужно пометить файловые системы, к которым будут применены квоты, в /etc/fstab. По умолчанию, в корне таких файловых систем будут созданы файлы quota.user и quota.group, в которых будет храниться информация о квотах. Это положение вещей можно изменить, указав имя файла с квотами в в /etc/fstab? например, "userquota=/var/quotas/quota.user". Ниже приводится пример /etc/fstab, в котором одна файловая система поддерживает квоты и файл квот находится в нестандартном месте (/var/quotas/quota.user):

```
/dev/wd0a / ffs rw,userquota=/var/quotas/quota.user 1 1
```

Для установки квот пользователям используется инструмент edquota(8). Простой вариант использования - "edquota <ИмяПользователя>". edquota(8) использует vi(1) для редактирования файла квот, если иной редактор не указан переменной окружения EDITOR. Например:

```
# edquota ericj
```

Вывод команды будет похож на следующее:

```
Quotas for user ericj:

/: blocks in use: 62, limits (soft = 0, hard = 0) inodes in use: 25, limits (soft = 0, hard = 0)
```

Для установки пределов, редактировать до получения результата, похожего на следующее:

```
Quotas for user ericj:

/: blocks in use: 62, limits (soft = 1000, hard = 1050) inodes in use: 25, limits (soft = 0, hard = 0)
```

Квота выделяется в блоках по 1 килобайту. В данном случае, нестрогий предел установлен в 1000k, а строгий - в 1050k. Нестрогий предел - та отметка, на которой пользователь получает предупреждение о необходимости привести использование в установленные рамки в определённый срок. Этот срок может быть установлен параметром -t вызова edquota(8). По истечение установленного срока, нестрогий предел расценивается как строгий, что обычно вызывает ошибку выделения места на диске.

Когда квоты установлены, их нужно включить командой quotaon(8). Например:

```
# quotaon -a
```

Команда проанализирует /etc/fstab и включит квоты для файловых систем, которые их поддерживают.Теперь, когда квоты включены, их можно просмотреть при помощи команды quota(1). Использование "quota <ИмяПользователя>" выдаст информацию, касающуюся выбранного пользователя. При вызове без аргументов, выводится статистика квот.

Например:

```
# quota ericj
```

Выдаст информацию, похожую на:

```
Disk quotas for user ericj (uid 1001):

    Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
             /      62    1000    1050              27       0       0
```

По умолчанию, квоты, установленные в /etc/fstab включаются при загрузке системы. Их можно выключить командой

```
# quotaoff -a
```

## 10.12 - Настройка клиента и сервера KerberosV

OpenBSD включает в себя KerberosV как предустановленную часть системы по умолчанию.

Для получения дополнительной информации о KerberosV из вашей системы OpenBSD используйте команду:

```
    # info heimdal
```

## 10.13 - Настройка сервисов анонимного FTP

Анонимный доступ к FTP даёт доступ к локальным файлам по протоколу FTP пользователям без учётной записи. Далее приводится обзор настройки анонимного сервера FTP, просмотр журналов и т.д.

### Создание учётной записи FTP

Для начала необходимо чтобы в системе присутствовала учётная запись с именем *ftp* и с некоторым паролем (см. ниже) Домашним каталогом для этой учётной записи устанавливается /home/ftp - можно задать его где угодно, при анонимном входе демон ftp сделает себе chroot в каталог пользователя *ftp*. Для более подробной информации можно обратиться к руководствам по ftp(8) и chroot(2). Далее - пример создания пользователя *ftp* с помощью adduser(8). Нам также потребуется добавить /usr/bin/false в */etc/shells* чтобы установить шелл по умолчанию для пользователя *ftp*. Пользователь даже в случае использования пустого пароля не сможет зайти в систему.

```
echo /usr/bin/false >> /etc/shells
```

После этого мы готовы добавить пользователя *ftp*.

```
# adduser
Use option ``-silent if you don't want to see all warnings and questions.

Reading /etc/shells
Reading /etc/login.conf
Check /etc/master.passwd
Check /etc/group

Ok, let's go.
Don't worry about mistakes. I will give you the chance later to correct any input.
Enter username []: ftp
Enter full name []: anonymous ftp
Enter shell csh false ksh nologin sh tcsh zsh [sh]: false
Uid [1002]: Enter
Login group ftp [ftp]: Enter
Login group is ``ftp. Invite ftp into other groups: guest no
[no]: no
Login class auth-defaults auth-ftp-defaults daemon default staff
[default]: Enter
Enter password []: Enter
Set the password so that user cannot logon? (y/n) [n]: y

Name:        ftp
Password:    ****
Fullname:    anonymous ftp
Uid:         1002
Gid:         1002 (ftp)
Groups:      ftp
Login Class: default
HOME:        /home/ftp
Shell:       /usr/bin/false
OK? (y/n) [y]: y
Added user ``ftp
Copy files from /etc/skel to /home/ftp
Add another user? (y/n) [y]: n
Goodbye!
```

## Настройка каталога

Вместе с пользователем был создан домашний каталог */home/ftp*. Нужны некоторые изменения для использования его в качестве каталога анонимного ftp - эти изменения объясняются в руководстве по ftp(8).

Создавать /home/ftp/usr и /home/ftp/bin **не нужно**.

*/home/ftp* - Главный каталог, принадлежит пользователю root и имеет права доступа 555.

*/home/ftp/etc* - Это можно, но не рекомендуется. Этот каталог используется только для того, чтобы сервер мог показать, кому из реальных пользователей системы какие файлы принадлежат. Для этого туда нужно скопировать /etc/pwd.db и /etc/group. Папка должна иметь права доступа 551, а два последних файла - 444. Они используются только для сопоставления имён числовым идентификаторам. Пароли хранятся в spwd.db - вот его-то и НЕ нужно копировать

*/home/ftp/pub* - Это стандартное место для расположения предоставляемых файлов. Права доступа к этому каталогу должны быть 555.

Все эти каталоги должны принадлежать пользователю root. Ниже - как всё должно выглядеть:

```
# pwd
/home
# ls -laR ftp
total 5
dr-xr-xr-x  5 root  ftp    512 Jul  6 11:33 .
drwxr-xr-x  7 root  wheel  512 Jul  6 10:58 ..
dr-x--x--x  2 root  ftp    512 Jul  6 11:34 etc
dr-xr-xr-x  2 root  ftp    512 Jul  6 11:33 pub

ftp/etc:
total 43
dr-x--x--x  2 root  ftp    512 Jul  6 11:34 .
dr-xr-xr-x  5 root  ftp    512 Jul  6 11:33 ..
-r--r--r--  1 root  ftp    316 Jul  6 11:34 group
-r--r--r--  1 root  ftp  40960 Jul  6 11:34 pwd.db

ftp/pub:
total 2
dr-xr-xr-x  2 root  ftp  512 Jul  6 11:33 .
dr-xr-xr-x  5 root  ftp  512 Jul  6 11:33 ..
```

## Запуск сервера и регистрация активности.

Есть возможность запуска ftpd как с помощью inetd(8), так и сценариев rc. Примеры ниже показывают, как запускать демона из inetd.conf. Сначала - поподробнее о некоторых параметрах ftpd. По умолчанию, соответствующая строка */etc/inetd.conf* имеет вид:

```
ftp             stream  tcp     nowait  root    /usr/libexec/ftpd       ftpd -US
```

Здесь ftpd вызывается с флагом -US. Анонимные соединения будут регистрироваться в */var/log/ftpd*, а соответствующие сеансы - в */var/run/utmp*. Таким образом, информацию о сеансах можно будет просматривать командой who(1). В некоторых случаях уместно предоставить только анонимный доступ, запретив ftp-доступ пользователям. Для этого ftpd вызывается с параметром -A. Ниже приводится строка позволяющая только анонимный доступ. Здесь также используется *-ll* для записи всех соединений в syslog, вместе с такими ftp-командами как get, retrieve и т.д.

```
ftp             stream  tcp     nowait  root    /usr/libexec/tcpd       ftpd -llUSA
```

**Внимание**, для СИЛЬНО загруженных ftp-серверов запуск демона через inetd не рекомендуется. Лучше закомментировать соответствующую строку в inetd.conf и запускать ftpd из rc.conf с параметром -D - такой способ запуска демона требует меньше системных ресурсов. Пример строки для запуска ftpd из rc.conf.local.

```
ftpd_flags="-DllUSA"
 # for non-inetd use: ftpd_flags="-D"
```

Это сработает только в том случае, если строка ftpd закомментирована/удалена из */etc/inetd.conf* и inetd перечитал свой файл конфигурации.

Другие файлы, имеющие отношение к вопросу:

*/etc/ftpwelcome* - Содержит приветствие для вновь подключившихся к серверу.
*/etc/motd* - Сообщение для тех, кто вошёл в ftp-сервер.
*.message* - Этот файл можно размещать в любом каталоге - он будет отображён при входе в этот каталог.

## 10.14 - Как ограничить доступ пользователю ftpd(8) его домашним каталогом?

По умолчанию, пользователи ftp после входа могут перемещаться в любое место файловой системы, куда им предоставлен доступ. В некоторых случаях это нежелательно и можно ограничить доступ к файловой системе домашним каталогом.

Использовать параметр -A при запуске демона ftpd(8), если вы хототе разрешить пользоваться только в чруте.

Если вы хотите произвести более тонкую настройку, используйте login capability infrastructure и настройку ftpd(8).

Пользователи с указаниями в login class с переменной ftp-chroot автоматически оказывются в чруте. Также вы можете воспользоватья добавлением username в файл **/etc/ftpchroot**. Пользователь должен быть упомянут в одном из указанных мест.

## 10.15 - Применение исправлений (патчей) в OpenBSD

Ошибки встречаются даже в OpenBSD. Некоторые ошибки могут привести к проблемам устойчивости (например, нечто может заставить систему прекратить функционировать желаемым образом). Другие ошибки могут привести к уязвимостям безопасности (которые могут позволить другим "использовать" компьютер для своих целей). Когда находят критическую ошибку, обновление направляется в дерево *-current* исходных текстов и выпускаются исправления для поддерживаемых версий OpenBSD. Эти исправления выкладываются на странице errata и разделены на "общие" - относящиеся ко всем платформам и те, что относятся к одной или более, но не всем платформам.

Однако, эти исправления не дополняют OpenBSD - они предназначены только для устранения проблем устойчивости и безопасности и должны немедленно применяться к подверженным системам (обычно, это НЕ ВСЕ системы, в зависимости от их назначения).

Существуют три пути обновления системы посредством исправлений:

**Обновить систему до** *-current*. Так как все исправления применяются к исходному коду -current, обновление системы до последнего среза является очень хорошим способом использования исправленного кода. Однако, работа с -current не для всех.

**Обновить систему до** *-stable*. Это можно сделать, получив новое или обновив имеющееся дерево исходных кодов до соответствующей -stable ветви и пересобрав ядро и файлы пользовательского уровня. Вобщем, это, может быть, самый лёгкий путь, но он занимает больше времени (пересобирается вся система) и полная сверка исходного кода может занять много времени при ограниченной пропукной способности канала.

**Применить исправление, собрать и установить отдельные затронутые файлы.** Именно этот способ будет описан в примере ниже. Это требует передачи меньшего количества данных и, обычно, меньше времени чем полная сверка/обновление исходного кода через cvs(1), но часто это наиболее сложный способ потому, что нет определённого универсального алгоритма. Иногда приходится исправлять, собирать и устанавливать один файл, иногда целую ветку (когда проблема в библиотеке).

Опять же, исправление отдельных файлов не всегда просто, что даёт серьёзный повод для использования *-stable* (или исправленной "patch") ветви OpenBSD. Смешивание и сравнение решений для исправления системы возможны лишь при хорошем понимании механизма, новым пользователям стоит выбрать один из методов и строго его придерживаться.

## Как исправления "errata" отличаются от использования дерева CVS?

Все исправления, выложенные на странице errata предназначены именно для исходного кода релиза конкретной указанной версии. Исправления для последнего CVS могут содержать изменения, нежелательные в релизе. Важно то, что если установлен срез (snapshot), в тот же момент проведена сверка дерева исходных кодов, попытка применить опубликованное исправление может не увенчаться успехом по причине возможных изменений, произошедших за это (между сверкой и публикацией) время.

## Применение исправлений.

Исправления для опреационной системы OpenBSD распространяются как унифицированные изменения (Unified diffs), которые являются текстовыми файлами, содержащими изменения в изначальном исходном коде. Они **НЕ** распространяются в двоичной форме. Это означает, что для исправления системы необходимо иметь в наличии исходный код соответствующей версии **РЕЛИЗА**. Вобщем, понадобится всё дерево исходных кодов. Для тех, кто пользуется оффициальным набором CD, исходные коды находятся на диске 3, они также доступны в виде файлов на серверах FTP. Предполагается, что всё дерево прошло сверку.

Например, рассмотрим исправление 001 для OpenBSD 3.6, относящееся к драйверу st(4) ленточных накопителей. Без этого исправления, восстановление данных с резервных копий достаточно сложно. Использующим накопители на магнитной ленте это обновление необходимо, тогда как остальным оно не нужно. Рассмотрим исправление:

```
# '''more 001_st.patch'''
Apply by doing:
        cd /usr/src
        patch -p0 < 001_st.patch

Rebuild your kernel.

Index: sys/scsi/st.c
===================================================================
RCS file: /cvs/src/sys/scsi/st.c,v
```

```
retrieving revision 1.41
retrieving revision 1.41.2.1
diff -u -p -r1.41 -r1.41.2.1
--- sys/scsi/st.c       1 Aug 2004 23:01:06 -0000       1.41
+++ sys/scsi/st.c       2 Nov 2004 01:05:50 -0000       1.41.2.1
@@ -1815,7 +1815,7 @@ st_interpret_sense(xs)
        u_int8_t skey = sense->flags & SSD_KEY;
        int32_t info;

-       if (((sense->flags & SDEV_OPEN) == 0) ||
+       if (((sc_link->flags & SDEV_OPEN) == 0) ||
            (serr != 0x70 && serr != 0x71))
                return (EJUSTRETURN); /* let the generic code handle it */
```

Видно, что в начале исправления содержится краткая инструкция по его применению. Предположим, что это исправление находится в каталоге /usr/src - в таком случае, проделаем следующее:

```
# '''cd /usr/src'''
# '''patch -p0 < 001_st.patch'''
Hmm...  Looks like a unified diff to me...
The text leading up to this was:
--------------------------
|Apply by doing:
|        cd /usr/src
|        patch -p0 < 001_st.patch
|
|Rebuild your kernel.
|
|Index: sys/scsi/st.c
|===================================================================
|RCS file: /cvs/src/sys/scsi/st.c,v
|retrieving revision 1.41
|retrieving revision 1.41.2.1
|diff -u -p -r1.41 -r1.41.2.1
|--- sys/scsi/st.c      1 Aug 2004 23:01:06 -0000       1.41
|+++ sys/scsi/st.c      2 Nov 2004 01:05:50 -0000       1.41.2.1
--------------------------
Patching file sys/scsi/st.c using Plan A...
Hunk #1 succeeded at 1815.              <-- Look for this message!
done
```

Обратим внимание на сообщение "Hunk #1 succeeded" выше. Это означает, что исправление было успешно применено. Многие исправления намного сложнее данного и используют несколько фрагментов в нескольких файлах. В этом случае нужно следить за тем чтобы все фрагменты были успешно заменены во всех файлах. Если нет, обычно, это означает проблемы с деревом исходных кодов, отклонение от инструкций или повреждение исправления. Исправления очень чувствительны к пробельным символам - копирование и вставка из окна браузера часто заменяет символы табуляции на пробелы. Любые такие изменения делают исправление непригодным.

После этого можно собрать ядро обычным способом, установить его и перезагрузить систему.

Не все исправления относятся к ядру. В некоторых случаях, пересобираются отдельные утилиты. Иногда приходится пересобирать все утилиты, статически связанные с исправленной библиотекой. Нужно следовать указаниям в заголовке

исправления и, в случае неуверенности, пересобирать всю систему.

Обычно, исправления, не имеющие отношения к конкретной системе применять не нужно. Например, если в данной системе отсутствует накопитель на магнитной ленте, пользы от применения представленного выше исправления не будет. Однако, исправления предполагается применять по порядку - возможно, что более позднее исправление зависист от предыдущего. Об этом нужно помнить при выборе, какие исправления применять, а какие нет - в случае неуверенности, следует применить все по порядку.

## 10.16 - Расскажите подробнее об Apache в chroot(2)

В OpenBSD, Apache httpd(8) по умолчанию имеет ограниченный доступ к файловой системе - корень файловой системы изменён посредством chroot(2). Это значительно повышает безопасность, но может вызвать проблемы у неподготовленного пользователя.

### Что такое chroot?

Приложение, к которому применён chroot(2), получает доступ только к определённому каталогу и не может перемещаться по остальному дереву файловой системы - этот каталог приложение видит как корень (/)В случае httpd(8), программа запускается, открывает файлы журналов, подключается к TCP-порту (но пока не принимает запросов) и читает файл конфигурации. Затем, она заменяет корень файловой системы на */var/www*, снижает привилегии и начинает принимать запросы. Это означает, что все обслуживаемые файлы должны располагаться внутри */var/www* . Это значительно повышает безопасность - в случае нарушений безопасности Apache, повреждены могут оказаться только файлы в одном каталоге, доступные только для чтения без возможности повредить другим ресурсам.

### Что это означает для администратора?

Прямое ограничение доступа к файловой системе посредством chroot(2) достаточно необычно и по умолчанию не используется в большинстве других операционных систем. Многие приложения и настройки системы не будут работать про смене корня файловой системы без некоторой доработки. Следует понимать, что безопасность и удобство могут оказаться целями чсовсем противоположными. Имплементация Apache в OpenBSD не пытается найти компромисс между простотой и безопасностью.

**Унаследованные исторически шаблоны файловой системы:** Серверы, обновлённые с предыдущих версий OpenBSD могут содержать web-файлы в каталогах пользователей, что точно не будет работать после смены корня chroot(2), поскольку httpd(8) не имеет достум в домашние директории вида */home*. Администраторы также могут заметить, что существующий раздел /var/www мал для того чтобы вместить все нужные файлы. Можно либо всё переделать, либо не использовать chroot(2). Конечно, можно использовать символические ссылки из /var/www на домашние каталоги, однако вы НЕ сможете /var/www создать ссылки в другие места файловой системы -- это предотвратит chroot(2). Стоит также заметить, что и в случае chroot(2) доступа FTP FTP chroot также не позволит совершить переход по символическим ссылкам. A solution to this is to not use /home as your home directories for these users, rather use something similar to /var/www/users. Symbolic links can be used completely within the chroot(2), but they have to be relative, not absolute.

**Ротация лог-файлов**: обычно, журналы чередуются переименованием старых файлов и посылкой сигнала SIGUSR демону httpd(8) чтобы Apache закрыл свои старые журналы и открыл новые. Теперь это невозможно, так как httpd(8) не может открыть файлы журналов для чтения после снижения привелегий. httpd(8) должен быть остановлен и перезапущен:

```
# apachectl stop && apachectl start
```

Yes, the last line attempts to restart Apache immediately, and in case that fails it waits a few seconds and tries again. And yes, that does mean that for a few seconds every time you do your log rotation, your web server will be unavailable. While this could be annoying, any attempt to permit httpd(8) to reopen files after chroot(2)ing would defeat the very purpose of the chroot! There are also other strategies available, including logging to a pipe(2), and using an external log rotator at the other end of the pipe(2).

**Имеющиеся модули Apache**: Virtually all will load, however some may not work properly in chroot(2), and many have issues on "apachectl restart", generating an error, which causes httpd(8) to exit.

**Имеющиеся CGI**: большинство НЕ БУДУТ работать без дополнительных ухищрений. Им могут понадобиться программы и библиотека за пределами */var/www* - некоторые могут быть исправлены статическим связыванием (теперь библиотеки не нужны). Большинству достаточно добавить в /var/www необходимые файлы, однако, этот процесс нетривиален и требует глубокого знания основ программирования.

**File system mount options**: By default in OpenBSD, your /var partition will be mounted with the nosuid and nodev options. If you attempt to use an application within the chroot, you may need to change those options. You may need to do that even if you don't use the chroot option, of course.

В некоторых случаях, приложение или его конфигурация могут быть адаптированы для работы с изменённым корнем, в остальных случаях, придётся просто отключить возможность смены корня ключом -u в строке httpd в файле /etc/rc.conf

## Пример замены корня chroot(2) приложения: wwwcount

Для примера процесса адаптации приложения для работы с изменённым корнем, рассмотрим wwwcount, - простой web-счётчик, имеющийся в пакетах. Являясь эффективной программой, он не реагирует на замену корня apache и не будет работать в такой среде в конфигурации по умолчанию.

Сначала, установим wwwcount. После конфигурирования и установки выясняется, что приложение на работает, вызывая внутреннюю ошибку сервера (сообщение "Internal Server error"). Первый шаг - перезапустить apache с флагом -u чтобы убедиться, что проблема вызвана именно заменой корня, а не настройками системы.

```
# apachectl stop
/usr/sbin/apachectl stop: httpd stopped
# httpd -u
```

После этого выясняется, что счётчик работает правильно после смены принадлежности каталога, так чтобы apache (и исполняемые им CGI) имел право на запись. Значит, проблема действительно в замене корня - перезапустим apache с использованием замены корня:

```
# apachectl stop
/usr/sbin/apachectl stop: httpd stopped
# httpd
```

Хорошей отправной точкой является предположение, что wwwcount использует некоторые библиотеки и другие файлы, недоступные после смены корня. Можно воспользоваться командой ldd(1) для выяснения динамических зависимостей CGI:

```
# cd /var/www/cgi-bin/
# ldd Count.cgi
Count.cgi:
        Start    End      Type Ref Name
        00000000 00000000 exe   1  Count.cgi
        03791000 237ca000 rlib  1  /usr/lib/libc.so.30.3
        03db4000 03db4000 rtld  1  /usr/libexec/ld.so
```

Проблема локализована - два файла недоступны после смены корня - скопируем их в /var/www

```
# mkdir -p /var/www/usr/lib /var/www/usr/libexec
# cp /usr/lib/libc.so.30.3 /var/www/usr/lib
# cp /usr/libexec/ld.so /var/www/usr/libexec
```

и попробуем счётчик снова.

Теперь программа хотя бы запускается, но выдаёт сообщение об ошибке "Unable to open config file for reading" ("Невозможно открыть файл конфигурации для чтения"). Конфигурационный файл по умолчанию расположен в /var/www/wwwcount/conf, но после смены корня это будет в /wwwcount/conf. Можно либо пересобрать программу с учётом изменения местоположения файлов, или переместить файл данных. Так как программа установлена из пакета, сделаем файл доступным после смены корня. Для получения результата, пригодного для использования как со сменой корня, так и без неё, воспользуемся символической ссылкой:

```
# mkdir -p /var/www/var/www
# cd /var/www/var/www
# ln -s ../../wwwcount wwwcount
```

Данная ссылка создана для того, чтобы работать после смены корня. Проверим ещё раз - ещё проблема. Теперь wwwcount жалуется, что не может найти файлов "strip image", которые он использует для отображения сообщений. После недолгих поисков, они обнаруживаются в /usr/local/lib/wwwcount - скопируем их таким образом, чтобы они были доступны и после смены корня:

```
# tar cf - /usr/local/lib/wwwcount | (cd /var/www; tar xpf - )
```

Проверяем - работает!

Скопированы были только те файлы, которые были действительно нужны для работы. Вообще, только действительно необходимые файлы подлежат копированию.

## Стоит ли использовать chroot?

В расмотренном примере, программа была достаточно простой, однако, мы столкнулись с разнообразными проблемами.

*Не каждое приложение может и должно быть адаптировано к работе со сменённым корнем chroot(2).*

Целью является безопасный web-сервер, а смена корня есть средство для её достижения, а никак не цель. Смысл смены

корня Apache в OpenBSD в том, что в новой файловой системе пользователь, от имени которого работает Apache не сможет запустить других программ, изменить какие-нибудь файлы или притвориться другим пользователем. Ослабление требований ведёт к ослаблению безопасности. Chroot иль не chroot...

Некоторые приложение просты и адаптация их к работе в среде с изменённым корнем имеет смысл. Другие слишком сложны и, либо их адаптация не стоит затраченных усилий, либо слишком большая часть системы будет скопирована в новую подсистему - теряются преимущества использования chroot(2). Например, OpenWebMail, требует прав записи в почтовый каталог и домашние каталоги пользователей и возможности работать от имени любого пользователя системы. Попытка запихнуть её в chroot бессмысленна и имеет логическим концом потерю всех преимуществ chroot. Даже в случае настолько простого приложения, как рассмотренный выше счётчик, необходима запись на диск (для отслеживания счётчиков), что несколько уменьшает преимущества от использования chroot(2).

Менять корень любому приложению, требующему привелегий суперпользователя бессмысленно, так как суперпользователь может всегда преодолеть эти ограничения.

Не стоит забывать, что если процесс адаптации к chroot слишком сложен, усложняется также и процесс обновления системы, что, в конечном итоге, может привести к гораздо большему ослаблению безопасности, чем отказ от chroot.

## 10.17 - Могу ли я сменить командную оболочку пользователя root?

Иногда советуют никогда не менять интерпретатор командной строки для учетной записи root, но в OpenBSD это не так.

Для пользователя *root* командной оболочкой по-умолчанию является ksh.

В Unix, по традиции, используются только статические слинкованные интерпретаторы командной строки для учетной записи root, потому что если система загрузится в однопользовательском режиме, некорневые файловые системы не будут смотированы и динамически слинкованные интерпретаторы не смогут получить доступ к разделу /usr. В случае с OpenBSD это несущественно, так как, при входе в однопользовательком режиме, после авторизации, OpenBSD попросит указать интерпретатор, или предложит sh по-умолчанию. В OpenBSD три стандартных интерпретатора (csh, sh и ksh) слинкованы статически и могут быть использованы в однопользовательском режиме.

## 10.18 - Что ещё можно сказать о *ksh*?

ksh В OpenBSD - в действительности, pdksh, Public Domain Korn Shell, и является тем же самым исполняемым файлом, что и sh.

Пользователи, привыкшие к *bash*, часто используемому в Linux найдут для себя много знакомого в ksh. Ksh(1) много возможностей *bash*, включая дополнение по tab, редактирование командной строки и использование истории стрелкой вверх, CTRL-A/CTRL-E для перехода в начало/конец строки. Если необходимо использование других возможностей bash, bash может быть установлен из пакетов или портов.

Приглашение командной строки *ksh* может быть легко изменено для вывода более подробной информации вместо стандартного "$ " установкой переменной PS1. К примеру, добавлением следующей строки:

```
export PS1='$PWD $ '
```

в файле /etc/profile изменит приглашение системы на:

```
/home/nick $
```

См. файл /etc/ksh.kshrc, в котором находится множество удобных функций и примеров, которые могут быть использованы в пользовательском файле .profile.

Ksh(1) OpenBSD была усовершенствована введением нескольких специальных символов для использования в PS1 (изначальном приглашении системы) наподобие используемого в bash. Например,

```
\e - Вставляет ASCII-символ escape.
\h - Имя машины (без имени домена).
\H - Полное имя машины (с именем домена).
\n - Вставляет символ перевода строки.
\t - Текущее время в 24-часовом формате ЧЧ:ММ:СС.
\u - Имя текущего пользователя.
\w - Текущий рабочий каталог. $HOME сокращается как `~'.
\W - Полный путь к текущему каталогу.
\$ - Показывает "#" для root users, "$" других пользователей.
```

(См. руководство ksh(1) для более подробных описаний!)

Можно использовать следующую команду:

```
export PS1="\n\u@\H\n\w $ "
```

для получения излишне информативного, но иногда полезного приглашения системы.

Перевод соответствует $OpenBSD: faq10.html,v 1.137 2007/11/01 02:11:01 nick Exp $