Быстрое руководство по FreeBSD для пользователей Linux®

Аннотация

Этот документ предназначен для быстрого ознакомления с основами FreeBSD пользователей Linux® среднего и продвинутого уровня.

Содержание

1. Введение . . . 1
2. Стандартная оболочка . . . 1
3. Пакеты и порты: установка программ в FreeBSD . . . 2
4. Загрузка системы . . . 3
5. Настройка сети . . . 4
6. Firewall . . . 5
7. Обновление FreeBSD . . . 6
8. procfs: ушёл, но не забыт . . . 6
9. Распространенные команды . . . 7
10. Заключение . . . 7

1. Введение

В этом документе освещаются некоторые технические различия между FreeBSD и Linux®, чтобы пользователи Linux® среднего и продвинутого уровней могли быстро ознакомиться с основами FreeBSD.

Этот документ предполагает, что FreeBSD уже установлена. Для получения помощи по процессу установки обратитесь к главе Установка FreeBSD Руководства FreeBSD.

2. Стандартная оболочка

Пользователи Linux® часто удивляются, обнаружив, что Bash не является оболочкой по умолчанию в FreeBSD. На самом деле, Bash не включён в стандартную установку. Вместо этого используется совместимая с POSIX® shell оболочка sh(1) в качестве оболочки пользователя по умолчанию. Оболочка root — это tcsh(1) по умолчанию в FreeBSD 13 и более ранних версиях и sh(1) в FreeBSD 14 и позднее. Оболочка sh(1) очень похожа на Bash, но имеет гораздо меньший набор возможностей. Как правило, скрипты, написанные для sh(1), будут работать в Bash, но обратное не всегда верно.

Однако Bash и другие оболочки доступны для установки через Коллекцию портов и пакетов FreeBSD.

После установки другой оболочки используйте chsh(1), чтобы изменить оболочку пользователя по умолчанию. Рекомендуется оставить оболочку пользователя `root` без изменений, поскольку оболочки, не входящие в базовую поставку, устанавливаются в /usr/local/bin. В случае возникновения проблемы файловая система, в которой находится /usr/local/bin, может быть не смонтирована. В такой ситуации у `root` не будет доступа к своей оболочке по умолчанию, что помешает ему войти в систему и устранить проблему.

3. Пакеты и порты: установка программ в FreeBSD

FreeBSD предоставляет два метода установки приложений: бинарные пакеты и скомпилированные порты. Каждый метод имеет свои преимущества:

Бинарные пакеты

- Быстрая установка по сравнению с компиляцией крупных приложений.
- Не требует понимания того, как компилировать программное обеспечение.
- Нет необходимости устанавливать компилятор.

Порты

- Возможность настройки параметров установки.
- Могут быть применены пользовательские патчи.

Если установка приложения не требует дополнительной собственной настройки, достаточно установить пакет. Собирайте порт, если приложению требуется изменение параметров, заданных в пакетах по умолчанию. При необходимости собственный пакет можно собрать из портов с помощью `make package`.

Полный список всех доступных портов и пакетов можно найти здесь.

3.1. Пакеты

Пакеты — это предварительно скомпилированные приложения, эквиваленты файлов .deb в системах на основе Debian/Ubuntu и файлов .rpm в системах на основе Red Hat/Fedora. Пакеты устанавливаются с помощью `pkg`. Например, следующая команда устанавливает Apache 2.4:

```
# pkg install apache24
```

Для получения дополнительной информации о пакетах обратитесь к разделу 4.4 Руководства FreeBSD: Использование pkg для управления бинарными пакетами.

3.2. Порты

Коллекция портов FreeBSD — это набор Makefile и патчей, специально адаптированных для установки приложений из исходного кода в FreeBSD. При установке порта система загрузит исходный код, применит необходимые патчи, скомпилирует код и установит приложение вместе с требуемыми зависимостями.

Коллекция портов, иногда называемая деревом портов, может быть установлена в /usr/ports с помощью Git. Подробные инструкции по установке Коллекции портов можно найти в разделе 4.5.1 Руководства FreeBSD.

Для сборки порта перейдите в его каталог и запустите процесс сборки. В следующем примере устанавливается Apache 2.4 из коллекции портов:

```
# cd /usr/ports/www/apache24
# make install clean
```

Преимущество использования портов для установки программного обеспечения заключается в возможности настройки параметров установки. В этом примере указано, что также должен быть установлен модуль mod_ldap:

```
# cd /usr/ports/www/apache24
# make WITH_LDAP="YES" install clean
```

Обратитесь к разделу Использование коллекции портов для получения дополнительной информации.

4. Загрузка системы

Многие дистрибутивы Linux® используют систему инициализации SysV, тогда как FreeBSD использует традиционный BSD-стиль init(8). В BSD-стиле init(8) нет уровней запуска (run-levels), и файл /etc/inittab отсутствует. Вместо этого запуск системы управляется сценариями rc(8). При загрузке системы /etc/rc читает /etc/rc.conf и /etc/defaults/rc.conf, чтобы определить, какие службы должны быть запущены. Указанные службы затем запускаются путем выполнения соответствующих сценариев инициализации, расположенных в /etc/rc.d/ и /usr/local/etc/rc.d/. Эти сценарии аналогичны сценариям, расположенным в /etc/init.d/ в системах Linux®.

Скрипты, расположенные в /etc/rc.d/, предназначены для приложений, входящих в "базовую" систему, таких как cron(8), sshd(8) и syslog(3). Скрипты в /usr/local/etc/rc.d/ предназначены для пользовательских приложений, таких как Apache и Squid.

Поскольку FreeBSD разрабатывается как полноценная операционная система, пользовательские приложения не считаются частью «базовой» системы. Пользовательские приложения обычно устанавливаются с помощью Пакетов или Портов. Чтобы отделить их от базовой системы, пользовательские приложения устанавливаются в /usr/local/. Таким

образом, исполняемые файлы пользовательских приложений находятся в /usr/local/bin/, конфигурационные файлы — в /usr/local/etc/ и так далее.

Сервисы включаются путем добавления соответствующей записи в файл /etc/rc.conf. Системные параметры по умолчанию находятся в файле /etc/defaults/rc.conf, а настройки по умолчанию переопределяются параметрами из файла /etc/rc.conf. Подробнее о доступных параметрах можно узнать в rc.conf(5). При установке дополнительных приложений ознакомьтесь с сообщением установки, чтобы определить, как включить связанные сервисы.

Следующие записи в /etc/rc.conf включают sshd(8), активируют Apache 2.4 и указывают, что Apache должен запускаться с SSL.

```
# enable SSHD
sshd_enable="YES"
# enable Apache with SSL
apache24_enable="YES"
apache24_flags="-DSSL"
```

После включения службы в /etc/rc.conf её можно запустить без перезагрузки системы:

```
# service sshd start
# service apache24 start
```

Если служба не была включена, её можно запустить из командной строки с помощью `onestart`:

```
# service sshd onestart
```

5. Настройка сети

Вместо общего идентификатора *ethX*, который используется в Linux® для обозначения сетевого интерфейса, FreeBSD использует имя драйвера с последующим числом. Следующий вывод из ifconfig(8) показывает два сетевых интерфейса Intel® Pro 1000 (em0 и em1):

```
% ifconfig
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 10.10.10.100 netmask 0xffffff00 broadcast 10.10.10.255
        ether 00:50:56:a7:70:b2
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
```

```
        inet 192.168.10.222 netmask 0xffffff00 broadcast 192.168.10.255
        ether 00:50:56:a7:03:2b
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
```

IP-адрес может быть назначен интерфейсу с помощью ifconfig(8). Чтобы настройки сохранялись после перезагрузки, конфигурация IP должна быть указана в /etc/rc.conf. Следующие записи в /etc/rc.conf задают имя хоста, IP-адрес и шлюз по умолчанию:

```
hostname="server1.example.com"
ifconfig_em0="inet 10.10.10.100 netmask 255.255.255.0"
defaultrouter="10.10.10.1"
```

Используйте следующие записи для настройки интерфейса через DHCP:

```
hostname="server1.example.com"
ifconfig_em0="DHCP"
```

6. Firewall

FreeBSD не использует Linux® IPTABLES для своего межсетевого экрана. Вместо этого FreeBSD предлагает выбор из трёх межсетевых экранов на уровне ядра:

- PF
- IPFILTER
- IPFW

PF разработан проектом OpenBSD и портирован в FreeBSD. PF был создан как замена для IPFILTER, и его синтаксис схож с синтаксисом IPFILTER. PF может использоваться совместно с altq(4) для обеспечения функций QoS.

Этот пример записи PF разрешает входящий SSH:

```
pass in on $ext_if inet proto tcp from any to ($ext_if) port 22
```

IPFILTER — это приложение для межсетевого экрана, разработанное Дарреном Ридом. Оно не является специфичным для FreeBSD и было портировано на несколько операционных систем, включая NetBSD, OpenBSD, SunOS, HP/UX и Solaris.

Синтаксис IPFILTER для разрешения входящего SSH:

```
pass in on $ext_if proto tcp from any to any port = 22
```

IPFW — это межсетевой экран, разработанный и поддерживаемый FreeBSD. Он может быть использован совместно с dummynet(4) для обеспечения возможностей управления трафиком и имитации различных типов сетевых соединений.

Синтаксис IPFW для разрешения входящего SSH будет следующим:

```
ipfw add allow tcp from any to me 22 in via $ext_if
```

7. Обновление FreeBSD

Существует два метода обновления системы FreeBSD: из исходного кода или бинарные обновления.

Обновление из исходного кода — это наиболее сложный метод обновления, но он предоставляет наибольшую гибкость. Процесс включает синхронизацию локальной копии исходного кода FreeBSD с репозиторием FreeBSD Git. После того как локальный исходный код обновлён, можно скомпилировать новую версию ядра и пользовательского пространства.

Двоичные обновления аналогичны использованию `yum` или `apt-get` для обновления системы Linux®. В FreeBSD, freebsd-update(8) может быть использован для получения и установки новых двоичных обновлений. Эти обновления могут быть запланированы с помощью cron(8).



При использовании cron(8) для планирования обновлений используйте `freebsd-update cron` в crontab(1), чтобы снизить вероятность одновременного скачивания обновлений большим количеством машин:

```
0 3 * * * root /usr/sbin/freebsd-update cron
```

Для получения дополнительной информации об обновлениях исходного кода и бинарных обновлениях обратитесь к главе об обновлениях в Руководстве FreeBSD.

8. procfs: ушёл, но не забыт

В некоторых дистрибутивах Linux® можно посмотреть файл /proc/sys/net/ipv4/ip_forward, чтобы определить, включена ли IP-маршрутизация. В FreeBSD вместо этого используется sysctl(8) для просмотра этого и других системных параметров.

Например, используйте следующую команду, чтобы определить, включена ли IP-переадресация в системе FreeBSD:

```
% sysctl net.inet.ip.forwarding
net.inet.ip.forwarding: 0
```

Используйте `-a` для вывода всех системных настроек:

```
% sysctl -a | more
```

Если приложению требуется procfs, добавьте следующую запись в /etc/fstab:

```
proc                /proc           procfs  rw,noauto       0       0
```

Добавление `noauto` предотвратит автоматическое монтирование /proc при загрузке.

Чтобы подключить файловую систему без перезагрузки:

```
# mount /proc
```

9. Распространенные команды

Некоторые эквиваленты распространённых команд приведены ниже:

Команда Linux® (Red Hat/Debian)	**Эквивалент FreeBSD**	**Назначение**
`yum install package` / `apt-get install package`	`pkg install package`	Установить пакет из удалённого репозитория
`rpm -ivh package` / `dpkg -i package`	`pkg add package`	Установить локальный пакет
`rpm -qa` / `dpkg -l`	`pkg info`	Список установленных пакетов
`lspci`	`pciconf`	Перечислить устройства PCI
`lsmod`	`kldstat`	Перечислить загруженные модули ядра
`modprobe`	`kldload` / `kldunload`	Загрузка/выгрузка модулей ядра
`strace`	`truss`	Трассировка системных вызовов

10. Заключение

Этот документ предоставил обзор FreeBSD. Для более детального изучения этих тем, а также многих других, не рассмотренных в данном документе, обратитесь к Руководству FreeBSD.