

Coming to FreeBSD from Linux

This tutorial is a short (and I hope useful) introduction on FreeBSD for people who come from Linux world.

Many things are similar in both FreeBSD and Linux operating systems. Some things might look weird, first time for somebody who come from Linux and do not have knowledge of other UNIX-like operating systems.

For a detailed handbook on FreeBSD visit: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/

To keep this intro short, here are some things you must know:

General things:

After installing FreeBSD and activating SSH, you must create a user account in group wheel. By default FreeBSD installation of SSHd do not allow you to remotely connect via SSH on root, you must SSH as a user and then su to root. Also, FreeBSD will not allow any users to su to root. Your created user must be in group wheel to be allowed to su to root.

Networking:

In Linux you have eth0 for first network card, eth1 for second and so on. In FreeBSD things are a little different, you will have different names for network interfaces, name given by manufacturer of that network cards. For example rl0 is first network Realtek 8139 interface, rl1 second and so on. Intel 100 mbps network cards are fxp (fxp0, fxp1 and so on), Intel Gigabit network cards are em0, em1 and so on. Here is a link with drivers (and naming) for ethernet cards: <http://www.freebsd.org/releases/6.2R/hardware-i386.html#ETHERNET>

Hard Drives:

In Linux you have hda for first IDE hard drive, hdb for second and so on, also you have sda for first scsi/sata, sdb for second scsi/sata hard drive. In FreeBSD you have ad0 for first IDE drive, ad1 for second IDE drive, da0 for first scsi drive, da1 for second and so on. For SATA drives you have ad, usually numbering starts from ad4 (yes, it is ad like IDE) if you have IDE controller activated in bios, well it depends on motherboard model, and configuration of bios.

In FreeBSD CD/DVD drives are acd (acd0 first drive, acd1 second drive and so on).

Drive Partitions

In Linux you have hda1 for first partition, hda2 for second and so on.

In FreeBSD things are a little bit different. You have slices and partitions. Partitions are created within a slice. To understand better I'll give you an example:

An IDE (or SATA) drive of 200 GB size you have: ad0 (drive name)

We assume you only have one slice on total size of hard drive. That slice is ad0s1

Ok, then we assume on that slice we have 5 partitions: 1. root (mounted as /), 2. swap, 3. tmp (mounted as /tmp), 4. var (mounted as /var) and 5. usr, (mounted as /usr).

Then we will have

| | | |
|--------|----------------|-----------------|
| ad0s1a | root partition | mounted in / |
| ad0s1b | swap partition | |
| ad0s1c | raw partition | |
| ad0s1d | tmp partition | mounted in /tmp |
| ad0s1e | var partition | mounted in /var |
| ad0s1f | usr partition | mounted in /usr |

Configuring network

To configure nameservers: both Linux and FreeBSD use the same /etc/resolv.conf file, with the same content: "nameserver IP" (for example, IP is 10.0.0.1 and is the dns server that will be used). Multiple name servers can be used.

To configure network cards in Linux, it depends on the Linux distribution. For example in Debian you must edit

`/etc/network/interfaces` file, and setup there either static IP or DHCP, then restart network service by using `/etc/init.d/networking restart` command.

Following our example in Debian Linux you must have the following lines in `/etc/network/interfaces`:

```
# ----- /etc/network/interfaces -----
# Loopback device:
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.0.0.2
    broadcast 10.0.0.255
    netmask 255.255.255.0
    gateway 10.0.0.1
# ----- eof -----
```

In FreeBSD network interface can be setup from `/etc/rc.conf` file which is main FreeBSD configuration files. Here can also be defined other things like what services to start on boot, parameters for that service, screen saver, language for terminal, and other FreeBSD options.

Following our example in FreeBSD you must have the following lines in `/etc/rc.conf`

```
# ----- /etc/rc.conf -----
defaultrouter="10.0.0.1
ifconfig_fxp0="inet 10.0.0.1 netmask 255.255.255.0"
...
...
# ----- eof -----
```

To manually add a default route in Linux use: `route add default gw 10.0.0.1`

To manually add a default route in FreeBSD use: `route add default 10.0.0.1`
(we assume 10.0.0.1 is IP of the gateway).

Configuration files

In Linux most of configuration files are located in `/etc`, and in other subdirectories of `/etc` directory.

In FreeBSD only system configuration files are located in `/etc`. All application (userland) configuration files are located in `/usr/local/etc`. This approach keeps the system cleaner.

Services are started in FreeBSD from `/usr/local/etc/rc.d`. In that directory are copied scripts that will start userland applications. FreeBSD base services must be started from `/etc/rc.d` directory.

Editors:

To edit a file in FreeBSD use either `edit` or `vi` (installed by default). You can also install `nano` or `joe` from packages or ports. Also `mcedit` from Midnight Commander is available if you install `mc` package / port.

Firewalls

Linux: `iptables` / `netfilter`.

FreeBSD uses `ipfw` (native FreeBSD firewall), `ipfilter` (native on FreeBSD but it seems not to be used anymore, the interesting thing is that was ported to other unices, like Solaris, HP/UX, OpenBSD, NetBSD) and `PF` (OpenBSD's PF, packet filter, ported to FreeBSD). Usually either `ipfw` or `PF` is used, both have similar features.

More info can be found here:

IPFW: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/firewalls-ipfw.html

IPFW man page: `man ipfw`

PF: <http://www.openbsd.org/faq/pf/>
 PF man page: `man pf.conf`

Userland Applications, ports and packages

For this tutorial we will still use our example, comparing FreeBSD with Debian Linux.

In Debian you have `dpkg`, `apt-get` and `aptitude` for package management, packages have extension `.deb`.

In FreeBSD you have packages and ports. Packages are precompiled FreeBSD applications that can be installed using `pkg_install` command. FreeBSD's packages extension is `.bz2`, but those packages are not regular bz2 archives, it also contain an index recognized by `pkg_install` application, and script files that will be run at installation time. Port system on the other hand make possible fetch and compilation of applications before installation.

Examples of package management in Debian

```
apt-get install ftpd      # will install ftpd application under Debian Linux
apt-get remove ftpd     # will remove ftpd application from Debian Linux system
dpkg -i ftpd            # install ftpd application under Debian using dpkg
dpkg -r ftpd           # remove ftpd application under Debian using dpkg
dpkg -L ftpd           # find files installed by ftpd package under Debian
dpkg -l | grep ftpd    # find ftpd package from Debian installed packages
                        # (find if ftpd package is installed or not)
dpkg -s ftpd           # get info about ftpd package
dpkg -S /bin/netstat    # find to which package belongs netstat binary
aptitude install ftpd  # will install ftpd package
aptitude remove ftpd  # will remove ftpd package
aptitude seach ftpd    # will search for ftpd package
```

Examples of package management in FreeBSD

```
pkg_add -r apache       # install apache package
pkg_delete apache-2.2.4_2 # delete apache package
pkg_info               # list installed packages
pkg_info | grep apache  # find if apache package is installed
pkg_info -L apache-2.2.4_2 # list files installed by apache package (you must use the exact
                        # name of the package, if your version differs use previous
                        # command to find exact name of package.
pkg_info -r apache-2.2.4_2 # will show on which packages depend apache package
```

For more info consult man pages for `pkg_add`, `pkg_delete`, `pkg_info`, `pkg_create`, `pkg_version`.

Examples of managing ports in FreeBSD

If you do not want do install an application from packages, for example you need to recompile a package with a specific option, you have the possibility to install it from ports. For example we want to install `apache22`:

```
whereis apache22      # we will get apache22: /usr/ports/www/apache22
cd /usr/ports/www/apache22
make install          # this will fetch source files for apache and will compile and then
                        # install apache. If other ports will be needed by apache,
                        # those ports will be installed too.
make                  # only fetch and compile but not install port
make clean            # remove fetched/compiled port files
make deinstall        # deinstall port, but not remove from ports
make reinstall        # deinstall and reinstall port
```

Note1: After you've installed a port, it will be seen as a package too, using `pkg_info` command.

Note2: In order to be able to install ports, you must choose to install ports collection when you first install FreeBSD.

Configuring and starting a FreeBSD application after it was installed from packages or ports

As I've told you before, all configuration files for userland applications under FreeBSD are kepted in `/usr/local/etc`

(with few minor exceptions)

Configuring an application under FreeBSD consist of following steps:

- we asume you've installed application using pkg_install, or make install from ports
- find and edit config file (or files) that comes with that application
- edit /usr/local/etc/rc.d/servicename file and change servicename_enable from "NO" to "YES"
- edit /etc/rc.conf file and add servicename_enable="YES" if you want that service to start on boot
- run service: /usr/local/etc/rc.d/servicename start

FreeBSD Kernel

By default at installation a GENERIC kernel is used by FreeBSD. If you want to add some features to FreeBSD kernel (support for different options that are not in GENERIC) you have two possibilities:

- Load a kernel module for that option/feature
- Recompile the kernel

Load a kernel module:

```
kldload module.ko      # load a kernel module
kldstat                # list loaded modules
kldunload module.ko   # unload a module
```

Modules can be found in /boot/kernel/.

Compiling FreeBSD kernel:

(we asume you have installed i386 version of FreeBSD)

a) Copy and edit Kernel config file

```
cp /usr/src/sys/i386/conf/GENERIC /usr/src/sys/i386/conf/SERVER
```

```
edit /usr/src/sys/i386/conf/SERVER
```

Add your option to SERVER file, then save file.

b) Compile Kernel

```
cd /usr/src
```

```
make -j4 buildkernel KERNCONF=SERVER
```

```
make installkernel KERNCONF=SERVER
```

Other useful things in FreeBSD

Logs are generated using syslog facility. To configure syslog service, edit /etc/syslog.conf file. To restart syslog daemon, run /etc/rc.d/syslogd restart.

Cron service is managed by crond daemon. To configure you can use /etc/crontab file (not recommended by some sysadmins because if you cvsup and recompile your source tree you might loose your change, but to use instead a crontab file in /var/cron/tabs. To restart cron service /etc/rc.d/cron restart.

Sysctl can be use to tune/change many FreeBSD sysctl variables.

```
sysctl -a                # will show all sysctl variables
```

```
sysctl -w variable=value # will change a sysctl variable
```

To make sysctl to be setup at boot time put the variable=value line on a single line in /etc/sysctl.conf. Some sysctl values can only be changed before /etc scripts are processed, in that case you must put the sysctl variable=value in /etc/loader.conf file.