Introduction to FreeBSD
(Additional Material)

PacNOG I Workshop

June 20, 2005
Nadi, Fiji

Hervey Allen
Joel Jaeggli

Outline

- Why FreeBSD.
- The World of FreeBSD.
- FreeBSD 5.3 installation.
- Command line vs. GUI.
- Configuration via files.
- FreeBSD disk paritioning.
- FreeBSD directory structure.
- How FreeBSD boots (man boot).
- Commands and programs.
- Create and remove user accounts.
- The vi editor.

Outline continued

- Configuring a network interface.
- Shutdown and restart the server – runlevels.
- Services and what is running.
- How to install software:
  - packages
  - ports
  - source
  - cvs
- File permissions. Commands “chmod” and “chown”.
- Summary
- More resources.

Why FreeBSD?

A question I'm sure most of you are asking...

==> Sparky says, “Take a look at this discussion:”

ws.edu.isoc.org/workshops/2005/pre-SANOG-VI/day1/whyfreebsd.html

Linux != UNIX

The World of FreeBSD

Start here: http://www.freebsd.org/

- RELEASE (5.3 and 4.10 legacy)
- STABLE ('beta' code)
- CURRENT ('alpha' code)
- Ports
- Packages
- Documentation Project
  - FreeBSD Handbook

Installing FreeBSD (5.3)

- How can you install? (FreeBSD Handbook section 2.2.6)
  - A CDROM or DVD
  - Floppy disks (including preconfigued install)
  - An FTP site, going through a firewall, or using an HTTP proxy, as necessary
  - An NFS server
  - A DOS partition on the same computer
  - A SCSI or QIC tape
  - A dedicated parallel or serial connection

Command Line vs. GUI

- To administer a FreeBSD server you can do this entirely from the command line, or “shell”.
- A Graphical User Interface (GUI) is not necessary to provide services (web, email, print, file, database, etc.) using FreeBSD (or Linux/Unix).
- You can run multiple command line windows (shells) at the same time.
- To use a GUI you must install the X Windows system and a desktop environment such as Gnome or KDE. We'll do this later in the week.

Configuration via Files

- In the Windows world most configuration takes place inside the Windows Registry files. These are binary database files.
- Under FreeBSD (and Linux/Unix) almost all configuration is done using text files.
- Graphical tools to configure services under FreeBSD simply write to a configuration file.
- To configure services you usually need to be the system admin account, “root”, and you will often edit text files directly.

FreeBSD Disk Organization

If you wish to understand how FreeBSD organizes and views disks then read section 3.5 of the FreeBSD handbook for an excellent and succinct description.

If you come to disk partitioning from a Windows perspective you will find that UNIX (FreeBSD, Linux, Solaris, etc.) *partitions* data very effectively and easily.

In FreeBSD a “slice” is what you may consider to be a “partition” under Windows.

FreeBSD Partition Schemes

Partition	Usage
a	Root partition (/)
b	swap partition
c	Not used for filesystems.
d	Supposedly not often used.
e/f	/tmp, /usr, etc...

View partition information using "`df -h`" and "`swapinfo`"

FreeBSD Disk Slices

Sample Output to view disk slices from "`fdisk -s`"

```
/dev/ad0: 77520 cyl 16 hd 63 sec
Part          Start          Size Type Flags
   1:             63       8385867 0x0b 0x80
   2:        8385930       8385930 0xa5 0x00
   3:       16771860        208845 0x83 0x00
   4:       16980705      61159455 0x0f 0x00
```

This is a 40GB disk with 3 operating systems spread across four slices. The operating systems include Windows 2000 (1), FreeBSD (2), Linux (3) and the 4th partition is a DOS swap slice for Windows 2000.

FreeBSD Partitions in a Slice

You can see more detailed information about your disk slices by just typing "`fdisk`"

To see the partitions in a FreeBSD slice use "`disklabel /dev/DEV`":

```
# /dev/ad1s1:
8 partitions:
#        size   offset    fstype   [fsize bsize bps/cpg]
  a:   524288        0    4.2BSD     2048 16384 32776
  b:  2045568   524288      swap
  c: 122865057        0    unused        0     0         # "raw" part, don't edit
  d:   524288  2569856    4.2BSD     2048 16384 32776
  e:   524288  3094144    4.2BSD     2048 16384 32776
  f: 119246625  3618432    4.2BSD     2048 16384 28552
```

FreeBSD Partitions in a Slice cont.

To view slice partition information in a more "human" readable format use "`df -h`". This can, however, be misleading. For example:

```
Filesystem     Size    Used   Avail Capacity  Mounted on
/dev/ad1s1a    248M     35M    193M    15%    /
devfs          1.0K    1.0K      0B   100%    /dev
/dev/ad1s1e    248M    526K    227M     0%    /tmp
/dev/ad1s1f     55G    2.7G     48G     5%    /usr
/dev/ad1s1d    248M     42M    186M    18%    /var
/dev/ad1s2      55G     15G     38G    28%    /data
/dev/da0s1     500M    226M    274M    45%    /mnt/flash
```

Use "`swapinfo`" to see the swap partition:

```
Device          1K-blocks     Used    Avail Capacity
/dev/ad1s1b       1022784      124  1022660     0%
```

FreeBSD Directory Structure

Repeat after me:
"The command '`man hier`' is your friend."

So, why is your FreeBSD disk partition split in to "slices"? Largely to separate important file systems from each other. These filesystems are usually represented by specific directories.

Why not just run with everything in one place? That is, everything under root (/).

FreeBSD Directory Structure cont.

Advantages of a single filesystem:

- Easier to resize if you want to make it larger.
- Easier conceptually for some people.

Advantages of multiple filesystems:

- If one system fails other systems can still work:
  - User fills up disk with runaway program.
  - Power failure only damages one file system.
- FreeBSD can optimize layout of files based on the use for the filesystem.
- Logical separation of functionality, thus improving security. I.E. root can be read only.

A Few FreeBSD Directories

- Structure of partitions/directories:
  - / (“root”)
  - /usr
  - /var
  - swap
- Two important directories:
  - /var/tmp
  - /usr/home

“/” Root

The root partition is where critical system files live, including the programs necessary to boot the system in to “single user” mode.

The idea is that this part of the system does not grow or change, but rather stays isolated from the rest of the operating system.

If you give enough room to /usr and /var, then “/” can be quite small (around 512MB should be safe for now).

The one directory that may grow is /tmp, particularly if you run Linux binaries that use /tmp.

/usr

Is used for system software like user tools, compilers, XWindows, and local repositories under the /usr/local hierarchy.

If one has to expand* this partition for additional software, then having it separate makes this possible.

FreeBSD maps user directories to /usr/home.

*We'll discuss this. We don't always install FreeBSD with a separate /usr partition.

/var

This is where files and directories that consistently change are kept. For example, webserver logs, email directories, print spools, temporary files, etc.

On a server it is a good idea to have /var in a separate partition to avoid having it fill your other filesystems by accident.

swap

Swap is where virtual memory lives. Swap is it's own filesystem.

You can run without swap, and your PC may run faster, but this is dangerous if you run out of memory.

There are several opinions about what is the optimal swap size. This can depend on what type of services you run (databases need more swap). The general rule of thumb is that swap size should be somewhere between your RAM and twice your server's RAM.

How FreeBSD Boots

The init process:

- After the kernel boots, which is located in “/” (in Linux it's usually /boot) it hands over control to the program /sbin/init.
- If filesystems look good then init begins reading the resource configuration of the system. These files are read in this order:
  - /etc/defaults/rc.conf
  - /etc/rc.conf (overrides previous)
  - /etc/rc.conf.local (overrides previous)
- Mounts file systems in /etc/fstab

How FreeBSD Boots cont.

The init process cont.:

- Once file systems are mounted then the following starts:
  - Networking services
  - System daemons
  - Locally installed package daemons (/usr/local/etc/rc.d scripts)

Init process and shutdown:

- When `shutdown` is called then init runs the scripts /etc/rc.shutdown.

Commands - Programs – Shell – Path

What's a “command” and a “program”?

Why can't you always run all commands and programs on a system?

How do you “fix” this?

How do you see how things are configured for a user?

- /usr/share/skel
- /etc/profile
- /home/user/.bashrc
- /home/user/.bash_profile
- set, printenv, export

Basic Commands

- cp, cd*, ls, mkdir, mv, rm y man
  - (*built in command shell commands).
- Where are commands located?
- /bin, /usr/bin, /usr/local/bin, /sbin, /usr/sbin
  - The difference between “sbin”, “bin” and “ /usr”
- If you know DOS:
  - cp = copy
  - cd/chdir = cd/chdir
  - ls = dir
  - mkdir = mkdir
  - mv = move (before it was copy and delete/erase)
  - rm = del[ete] and/or erase

Create, Remove, Update User Accounts

(FreeBSD Handbook section 8.6)

User Creation and Maintenance

- passwd, pw, vipw

Some Associated Files

- /etc/passwd, /etc/group, /etc/master.passwd, /etc/sudoers (note visudo)
- /usr/share/skel
- /var/mail

/etc/passwd

The /etc/password file has the following format:

```
hervey:x:500:500:Hervey
   Allen:/home/hervey:/usr/local/bin/bash
```

i.e.:

```
user:pw:UID:GID:name:directory:shell
```

Using /etc/master.passwd the “pw” is represented by an “x”. If the user entry is actually something like a service, then the “shell” is represented with “/sbin/nologin”.

/etc/master.passwd

This file is used to hide encoded user passwords. Only root can (or should) read this file. /etc/pwd.db is a Berkeley db password database that is used by most applications for efficient user authentication.

/etc/master.passwd has the following format:

```
hervey:$1$qvAgYWGD$nLf/LpT1r0XXXXXXjMC/:1001:1001::0:0:Hervey
   Allen:/home/hervey:/usr/local/bin/bash
```

i.e.:

- User's login name.
- Users encoded password. If starts with “1” it's md5 encyrpted.
- User's ID number.
- User's login group ID.
- User's classification (unused).

/etc/master.passwd cont.

```
hervey:$1$qvAgYWGD$nLf/LpT1r0XXXXXXjMC/:1001:1001::0:0:Hervey
   Allen:/home/hervey:/usr/local/bin/bash
```

- Password change time. (0 means never)
- When the account expires (0 means never)
- General user information (like full name...)
- User's home directory.
- User's login shell.

The vi Editor

- Why use vi? Why not emacs, xemacs, joe, pico, ee, etc.? (*Ask me* about “pico -w”)
- vi exists in almost all flavors of Unix and Linux.
- If you have to work on a new machine, then vi will almost always be available to you.
- In reality, you are likely to use a different editor for more complex editing, but we will practice using vi after we install FreeBSD.

Configuring Network Interfaces

During boot if a NIC is recognized then the appropriate code is loaded to support the NIC (a module).

After boot, using "`ifconfig`" you can see if the NIC exists. Look for MAC address.

Initial NIC configuration can be done with `ifconfig`, or try "`dhclient dev`"

If NIC works, edit /etc/rc.conf and put in device specific entries for each boot.

Configuring Network Interfaces cont.

Example lines in /etc/rc.conf for network device:

```
hostname="localhost.localdomain"
ifconfig_wi0="DHCP"
```

Set the hostname and indicate that NIC "`wi0`" will use DHCP to get network information. FreeBSD uses specific names for each network device. "`wi0`" indicates the first "Wireless" card.

Shutdown and Restart a Server

How do you shutdown a FreeBSD box?

- shutdown 1 message
- halt
- init 0

And, to restart?

- reboot
- shutdown -r now
- init 6

Run Levels

FreeBSD has the concept of run levels:

Run-level	Signal	Action
0	SIGUSR2	Halt and turn the power off
1	SIGTERM	Go to single-user mode
6	SIGINT	Reboot the machine

So, in reality, you either run in single-user mode with “everything off” and just root access (run-level 1), or your system is up and fully running in multi-user mode.

Run Levels cont.

Order of what's run in multi-user mode:

- /etc/defaults/rc.conf (scripts in /etc/rc.d correspond).
- Local overrides from /etc/rc.conf.
- Filesystems mounted as described in /etc/fstab.
- Third party services with installed startup scripts run from /usr/local/etc/rc.d.

Most local settings will go in:

/etc/rc.conf

What's Running on a System

- To view all services:
  - `ps -aux | more`
- To view a particular service
  - `ps -aux | grep “name”`

Note the “|” character… This is how we “connect” the results of one command to another command.

Software Install Methods

There are three methods to install software on your FreeBSD system. These are:

1.) FreeBSD packages and the `pkg` utility.

2.) The ports collection /usr/ports.

3.) Installing from source (gcc make).

You are most likely to install from packages, then ports, then from source.

There are advantages and disadvantages to each.

The “pkg” Commands

In general the pkg_add and pkg_delete facilities allow you to install and remove software on your system in an efficient and consistent manner.

The pkg_info command allows you to see what's installed, quickly, and to get detailed information about each software package that is installed.

Package Installation Using pkg_add

- You can get “packages” from local source (a CD), off FreeBSD sites, or your local network.
- To install a package from a CD-ROM:

```
pkg_add /cdrom/dir/package_name
```

- To install from an ftp server you can do:

```
pkg_add ftp://address/dir/package_name
```

Using pkg_info

Find out if something is already installed:

```
pkg_info        (list all installed packages)
pkg_info | grep moz    (find all packages
                        containing “moz”)
```

Get more information about an already installed package:

```
pkg_info name\*

pkg_info -I name\*
```

For example “`pkg -I bash\*`” returns:

```
bash-2.05b.007_2    The GNU Bourne Again Shell
```

Using pkg_delete

If you have a package you wish to remove you can simply type:

```
pkg_delete package_name
```

But, if you want to remove the package and all its dependent packages you would do:

```
pkg_delete -r package_name
```

But, *be careful* about doing this. You might want to check what will happen first by doing:

```
pkg_delete -n package_name
```

Installing from Ports

First you must have installed the /usr/ports collection during system installation. Otherwise, use /stand/sysinstall after installation and then choose Configure, Distributions, then Ports.

Once the “ports collection” is installed you can see the entire tree under /usr/ports. There are several thousand software packages available.

This collection contains minimal information so that you can “make” a software package quickly, and easily from separate CD-ROMs or a network site containing the port source.

See section section 4.5 of the FreeBSD Handbook.

Installing from Ports cont.

To see if a software package exists as a port:

```
cd /usr/ports
make search name=package
make search key=keyword
```

Let's do this for “lsof” (LiSt Open Files):

```
cd /usr/ports
make search name=lsof (or “whereis lsof”)
```

And the output from this is:

```
Port:   lsof-4.69.1
Path:   /usr/ports/sysutils/lsof
Info:   Lists information about open files (similar to fstat
  (1))
Maint:  obrien@FreeBSD.org
Index:  sysutils
B-deps:
R-deps
```

Installing from Ports cont.

From the previous page you'll note that the port is in /usr/ports/sysutils/lsof.

If you have a network connection...

You can simply type "`make install`"

But, you might want to do:

- `make`
- `make install`

To automatically get ports from a local server you can do this by changing a system variable:

- `export MASTER_SITE_OVERRIDE="ftp://local.site/distfiles/ fetch"`

Installing from Ports cont.

You can install from cdrom. If you have a cdrom with the full ports distfiles, then simply mount it. Then you would do:

- `cd /usr/ports/sysutils/lsof`
- `make`
- `make install`

And the port will find the distfile on /cdrom instead of from the internet.

Installing from Source

It's likely you'll want to install software that's either not available as a package or port, or that you need to change or reconfigure before installation.

In such cases, you need to compile the software from source code.

It's very typical that software comes as a single “tar” archive that is compressed (tar.gz or .tgz)

An example of how to install from source -->

Installing from Source cont.

- Download a file fn.tar.gz to /usr/src.
- tar -xvzf /usr/src/fn.tar.gz
- cd /usr/src/fn-version
- ./configure
- make
- make install

This is if everything works, but now you don't have any good way to uninstall the software...

CVS and CVSUP

One issue that arises, “How to keep your ports collection up-to-date?”

CVS, or Concurrent Versions System, can do this for you.

First you must install cvsup, then you can tell this tool to look on a server that has the latest ports collection and update your local collection with a single command like:

```
cvsup -g -L 2 -h cvsup.freebsd.org \
 /usr/share/examples/cvsup/ports-supfile
```

CVS and CVSUP cont.

Later today we'll update the ports collection on your machines using a local CVS server that we have installed.

Rather than cvsup.freebsd.org we'll use a local machine for this.

File Permissions

- There are five categories and three types of permissions that you need to consider.
- The default file permissions are set with the umask command.
- There are two categories of permissions that relate to the user or group that is going to run a file (setuid, setgid).
- Available access permissions are “r” (read), “w” (write), and “x” (execute).
- You can assign permissions to world (a), group (g), and user (u).

File Permissions cont.

- A file belongs to a user. You can assign a file to another user or another group using the chown ("CHange OWNer") command.
- You can change permissions and/or owners for a group of files or for all files and all files in subdirectories using the chmod and chown commands.
- Finally, you can change directory permissions using the chmod command.
- We will practice using file permissions later today.

Summary

- Aimed at stability not user desktops.
- Very, very good track record for stability and security.
- Scales to very large sizes for services.
- Large collection of software, including ability to run Linux packages.
- GUI is not necessary to provide services.
- Software can be installed in several ways.
- Configuration is done using simple text files.

More resources

This presentation is located here:

http://ws.edu.isoc.org/workshops/2005/pre-SANOG-VI/day1/

- http://www.freebsd.org/
- http://www.google.com/
- http://www.freebsd.org/support.html
- O'Reilly books (http://www.oreilly.com/)
- http://www.freshports.org/
- http://www.freebsddiary.org/
- SANOG mailing list (sanog@sanog.org)
- SANOG web site: http://www.sanog.org/