Unix/IP Preparation Course
May 29, 2011
Dar es Salaam, Tanzania
FreeBSD Timeline

Image courtesy of Wikipedia
### Unix and Linux

<table>
<thead>
<tr>
<th>Are they the same?</th>
<th>Yes, at least in terms of operating system interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linux was developed independently from Unix</td>
</tr>
<tr>
<td></td>
<td>Unix is much older (1969 vs. 1991)</td>
</tr>
</tbody>
</table>

**Scalability and reliability**

- Both scale very well and work well under heavy load  
  (this is an understatement 😛)

**Flexibility**

- Both emphasize small, interchangeable components

**Manageability**

- Remote logins rather than GUI (ssh)  
  Configuration is done with *plain-old-text™*  
  Scripting is integral

**Security**

- Due to modular design has a reasonable security model  
  All software has security holes
The Unix System Simplified

- Hardware
- The UNIX Kernel
- The Shell Program
- Command Processes
The Unix System More Detail

- **Background processing**
  - Non-interactive (no terminal)

- **Launch processes**
  - User interaction (terminal)
  - Job control (scripting)

- **I/O (+ terminals)**
  - Memory management (VM)
  - Interruptions
  - Scheduling / Timesharing

- **File management**
  - Editors
  - Compilers
  - Network tools

- **Users**
  - (uid / gid)

- **DAEMONS**

- **SHELLS**

- **KERNEL**

- **APPS**
The Kernel

• The "core" of the operating system
• Contains device drivers
  - Communicate with your hardware
  - Block devices (physical media – hard drive, CD, RAM)
  - Character devices (keyboards, mice, terminals, modems)
  - Network devices (network cards)
  - Pseudo devices (/dev/null, /dev/random)

• Filesystems
  - Organise block devices into files and directories
  - ufs2, ext2, ext3, ext4, reiserfs, jfs, zfs
Memory management
- Real, Virtual and paging algorithms

Timeslicing (multitasking)
- Resource allocation to processes

Networking stacks - esp. TCP/IP
- Packets traverse the kernel

Enforces security model
- Does this user have privileges
- Numeric userid identifiers (“uid”)
- id 0 is “special” - root
Shells

• Command line interface for executing programs
• Also programming languages for scripting
  – Windows equivalent: batch files or newer windows scripting language
• Choice of similar but slightly different shells
  – sh: the "Bourne Shell". Standardised in POSIX
  – csh: the "C Shell". Not standard, but includes command history
  – bash: the "Bourne-Again Shell". Combines POSIX standard with command history.
  – Others: ksh, tcsh, zsh
The programs that you choose to run

Frequently-used programs tend to have short cryptic names (why?)

"ls" = list files
"cp" = copy file
"rm" = remove (delete) file

Lots of stuff included in most base systems
Editors, compilers, system admin tools

Lots more stuff available to install as well
Thousands and thousands of packages
System processes

Programs that run in the background; also known as "daemons" ➔ “sparky”

Examples:

- **cron**: Executes programs at certain times of day
- **syslogd**: Takes log messages and writes them to files
- **inetd**: Accepts incoming TCP/IP connections and starts programs for each one
- **sshd**: Accepts incoming logins
- **sendmail** (other MTA daemons like Exim, Postifx): accepts incoming mail (smtp)
Security model

Numeric IDs
  user id (uid 0 = "root", the superuser)
  group id
  supplementary groups

Mapped to names in plain text files
  /etc/passwd
  /etc/group

Suitable security rules enforced
  e.g. you cannot kill a process running as a different user, unless you are "root"
Any questions?
Core directory refresher

/ (boot, /bin, /sbin, /etc, maybe /tmp)
/var (Log files, spool, maybe user mail)
/usr (Installed software packages)
/tmp (May reside under “/”)

Don't confuse the the “root account” (/root) with the “root” (“/”) partition.
During FreeBSD installation you can choose this option. It creates the following:

- "/" Small Root partition
  - this will contain everything not in another partition
    /bin, /sbin, /usr etc.
- A *swap partition* for virtual memory
- /var
  - For “variable” files, such as logs, mail spools, etc.
- /tmp
  - Where temporary files are located
- /usr
  - /usr/home contains user directories. This is the largest partition created. In Linux this is just /home.
Many/most Linux distributions will default to:

- **“/”** Almost every here. Very large partition.
  - this will contain everything not in another partition
    /bin, /sbin, /usr etc.

- **A swap partition** for virtual memory

- **/boot**
  - Contains the linux kernel image(s) and associated configuration and
  bootstrapping files.
Partitioning Issues

/var may not be big enough
/usr contains OS utilities, third-party software
/usr/home contains your own important data
If you reinstall from scratch and erase /home, you will lose your own data
  • Everything in “/” is now more common due to RAID. Why? Valid?
  • /tmp?
  • Others?
  • How much swap should you define?
Partitioning is just a logical division

If your hard drive dies, most likely everything will be lost.

If you want data security, then you need to set up mirroring with a separate drive.

Another reason to keep your data on a separate partition, e.g. /u
Remember, “rm -rf” on a mirror works very well.

Or, as always “Data Security” <==> Backup
Any questions?
Software Installation FreeBSD

Software management in FreeBSD

• Install from source
• Install from binary
• Compile from source using a port
• Use a wrapper tool, such as portinstall.
• Install pre-built FreeBSD packages using pkg_*

You can keep the source tree local and up-to-date. This is known as the ports collections. A number of tools to do this, including portsnap.
Two major packaging systems:
  - Redhat Package Manager ➔ RPM
  - Debian Packages ➔ DPKG

Both have wrapper tools to make them easier to use:
  - rpm wrapped with “yum”
  - dpkg wrapped with “apt” and “aptitude”

Both use repositories.

Linux has the other usual suspects as well:
  - Install from source
  - Install from binary

Most Linux users consider the Debian Packaging system, dpkg, to be superior to the Redhat Package Manager, rpm.
System Startup FreeBSD

Startup scripts in FreeBSD

- `/etc/rc.d` – system startup scripts
- `/usr/local/etc/rc.d` – third-party startup scripts

Controlling services

- In `/etc/defaults/rc.conf` – initial defaults
- `/etc/rc.conf` – override settings here
Several variants and new systems, such as *upstart*, are used. But, commonly:

- `/etc/rc.d` or
- `/etc/init.d/rc.d` – system startup scripts
- `/etc/rc.local` – customize startup

Controlling services

- In `/etc/` – and
- `/etc/default`
The use of the *root* account is discouraged and the *sudo* program should be used to access root privileges from your own account instead.

You can do a “*buildworld*” to move between major and minor releases (FreeBSD).

You can use *apt* and/or *yum* to move between many major and minor Linux releases.
Important Reads

• man builtin
• man hier
• man man
• man ports
• man rc.conf

And, “man any_unknown_command” when you are in doubt.
<table>
<thead>
<tr>
<th>There's More</th>
</tr>
</thead>
</table>

**The FreeBSD Handbook**  
http://www.freebsd.org/handbook/

**Some Web Resources**  
http://www.freebsd.org  
http://forums.freebsd.org  
http://www.freshports.org/  
http://wiki.freebsd.org  
http://en.wikipedia.org/wiki/FreeBSD

*GIYF (Google Is Your Friend)*
Packages & Exercises

We'll reinforce some of these concepts using exercises...