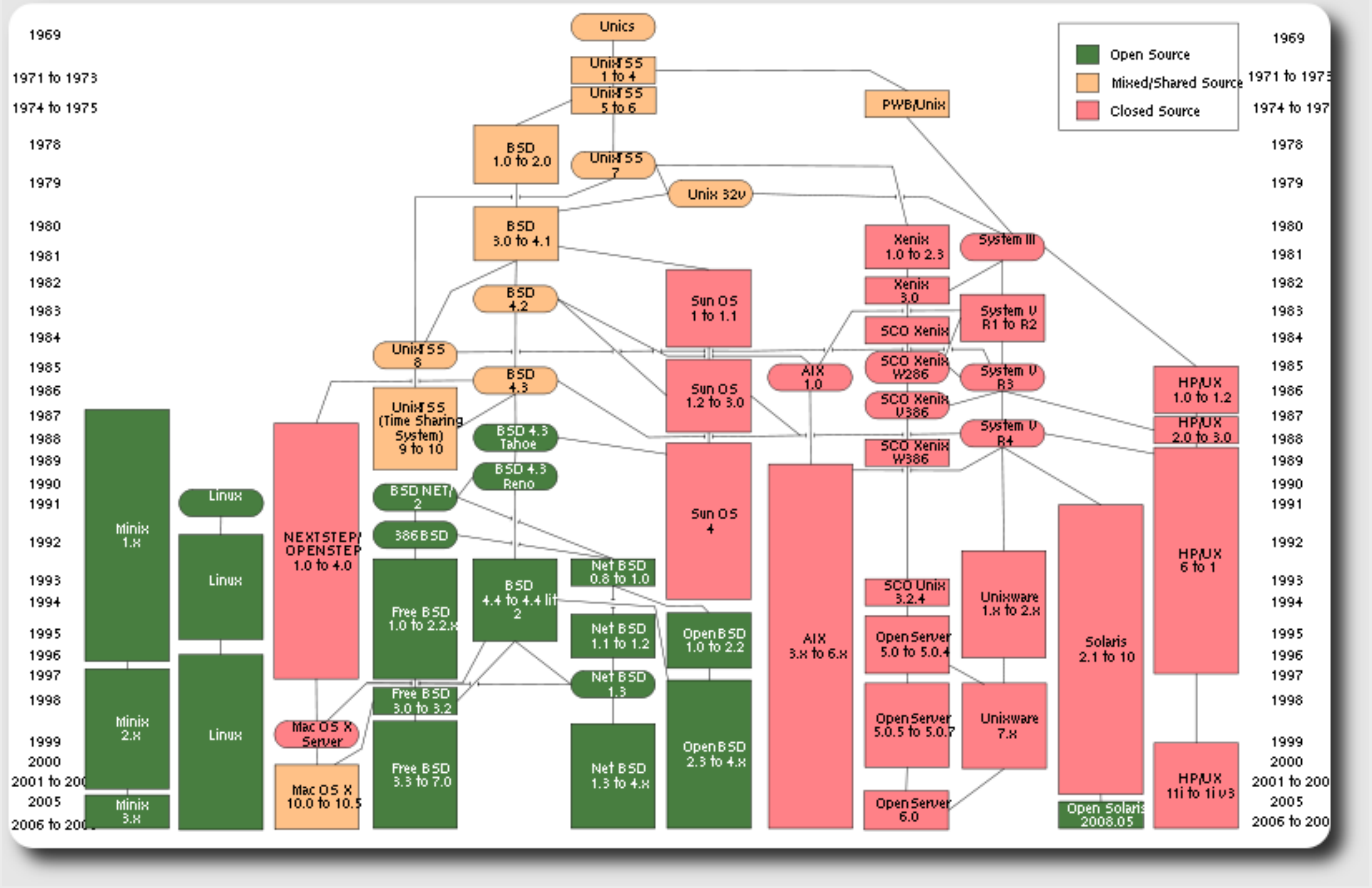
UNIX™/Linux Overview

Unix/IP Preparation Course
May 23, 2010
Kigali, Rwanda

UNIX History



FreeBSD Timeline

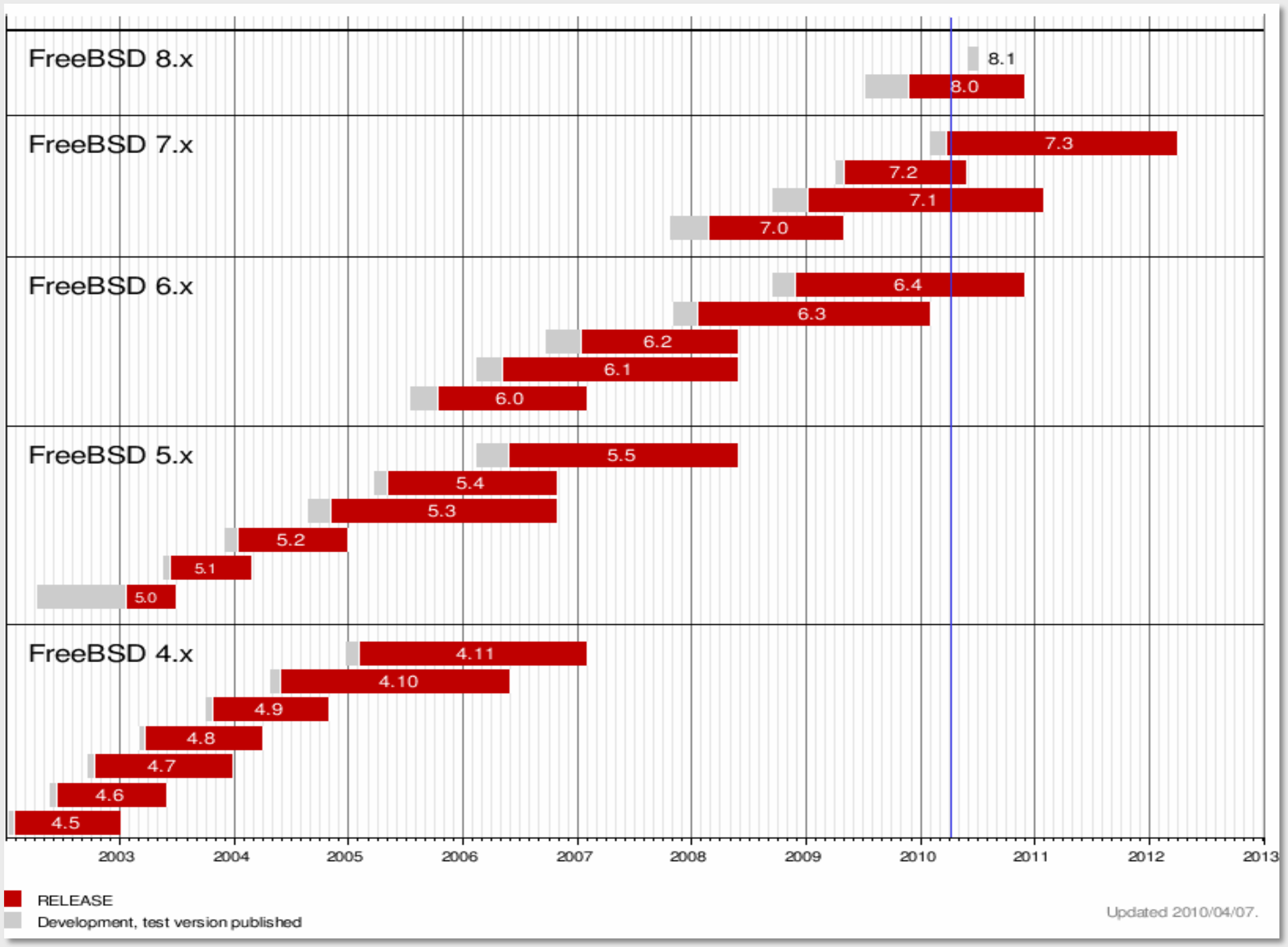
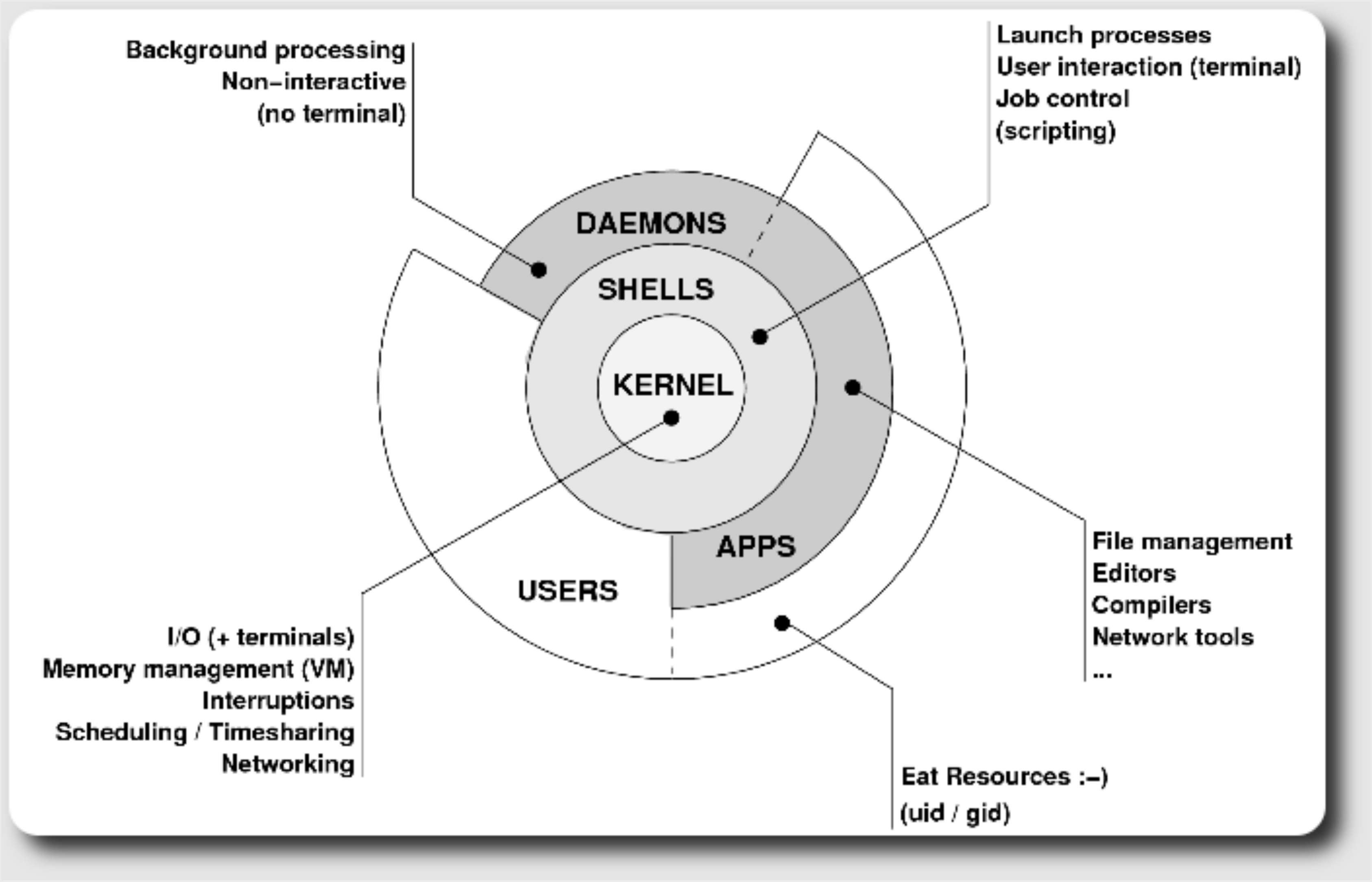


Image courtesy of Wikipedia

Unix vs. Linux

- Are they the same?
  - Yes, at least in terms of operating system interfaces
  - Linux was developed independently from Unix
  - Unix is much older (1969 vs. 1991)
- 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
  - Scripting is integral
- Security
  - Due to modular design has a reasonable security model
  - Linux and its applications are not without blame

The Unix System



Kernel

The "core" of the operating system

Device drivers

communicate with your hardware

block devices, character devices, network devices, pseudo devices

Filesystems

organise block devices into files and directories

Memory management

Timeslicing (multitasking)

Networking stacks - esp. TCP/IP

Enforces security model

Shells

Command line interface for executing programs

DOS/Windows equivalent: command.com or command.exe

Also programming languages for scripting

DOS/Windows equivalent: batch files

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

User processes

- The programs that you choose to run
- Frequently-used programs tend to have short cryptic names
  - "`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 too
  - Using the Debian/Ubuntu repositories

System processes

Programs that run in the background; also known as "daemons" ==>

Examples:

cron: executes programs at certain times of day

syslog<u>d</u>: takes log messages and writes them to files

inet<u>d</u>: accepts incoming TCP/IP connections and starts programs for each one

ssh<u>d</u>: accepts incoming logins

sendmail (other MTA daemon like Exim): accepts incoming mail

Security model

Numeric IDs

- user id (uid 0 = "*root*", the superuser)
- group id
- supplementary groups

Mapped to names

- /etc/passwd, /etc/group (plain text files)

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.

d

‘Auto Defaults’ 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.

Partitioning Issues

/var may not be big enough

/usr contains OS utilites, 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?

Note...

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

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.*

System Startup

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

Administration

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.

Important Reads

- man builtin
- man hier
- man man
- man ports
- man rc.conf

And, "`man any_unknown_command`" when you are in doubt.

There's More

The FreeBSD Handbook

http://www.freebsd.org/handbook/

Some Web Resources

http://www.freebsd.org
http://forums.freebsd.org
http://distrowatch.com/table.php?distribution=freebsd
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...