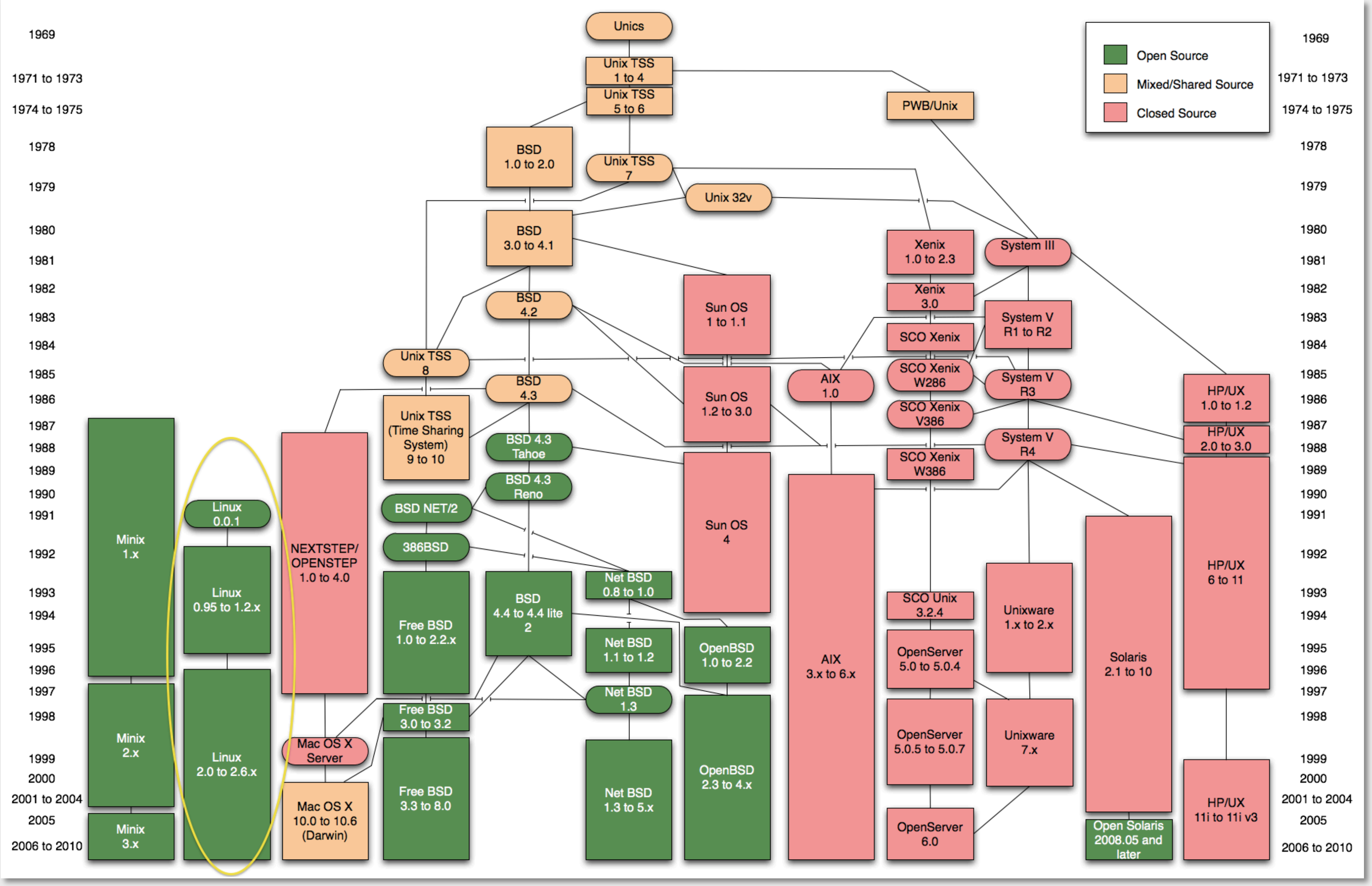
UNIX™/Linux Overview

Unix/IP Preparation Course
May 6, 2012
Serrekunda, The Gambia

UNIX/Linux History



🟩 Open Source

🟧 Mixed/Shared Source

🟥 Closed Source

1969

1971 to 1973

1974 to 1975

1978

1979

1980

1981

1982

1983

1984

1985

1986

1987

1988

1989

1990

1991

1992

1993

1994

1995

1996

1997

1998

1999

2000

2001 to 2004

2005

2006 to 2010

Unics

Unix TSS 1 to 4

Unix TSS 5 to 6

PWB/Unix

BSD 1.0 to 2.0

Unix TSS 7

Unix 32v

BSD 3.0 to 4.1

Xenix 1.0 to 2.3

System III

Sun OS 1 to 1.1

BSD 4.2

Xenix 3.0

System V R1 to R2

SCO Xenix

Unix TSS 8

BSD 4.3

AIX 1.0

SCO Xenix W286

System V R3

HP/UX 1.0 to 1.2

Unix TSS (Time Sharing System) 9 to 10

Sun OS 1.2 to 3.0

SCO Xenix V386

HP/UX 2.0 to 3.0

BSD 4.3 Tahoe

System V R4

SCO Xenix W386

Minix 1.x

NEXTSTEP/ OPENSTEP 1.0 to 4.0

BSD 4.3 Reno

Sun OS 4

Linux 0.0.1

BSD NET/2

386BSD

Net BSD 0.8 to 1.0

Linux 0.95 to 1.2.x

Free BSD 1.0 to 2.2.x

BSD 4.4 to 4.4 lite 2

SCO Unix 3.2.4

Unixware 1.x to 2.x

HP/UX 6 to 11

Net BSD 1.1 to 1.2

OpenBSD 1.0 to 2.2

AIX 3.x to 6.x

OpenServer 5.0 to 5.0.4

Solaris 2.1 to 10

Minix 2.x

Linux 2.0 to 2.6.x

Net BSD 1.3

Free BSD 3.0 to 3.2

Mac OS X Server

OpenServer 5.0.5 to 5.0.7

Unixware 7.x

Free BSD 3.3 to 8.0

Net BSD 1.3 to 5.x

OpenBSD 2.3 to 4.x

Mac OS X 10.0 to 10.6 (Darwin)

HP/UX 11i to 11i v3

OpenServer 6.0

Minix 3.x

Open Solaris 2008.05 and later

1969

1971 to 1973

1974 to 1975

1978

1979

1980

1981

1982

1983

1984

1985

1986

1987

1988

1989

1990

1991

1992

1993

1994

1995

1996

1997

1998

1999

2000

2001 to 2004

2005

2006 to 2010

FreeBSD Timeline

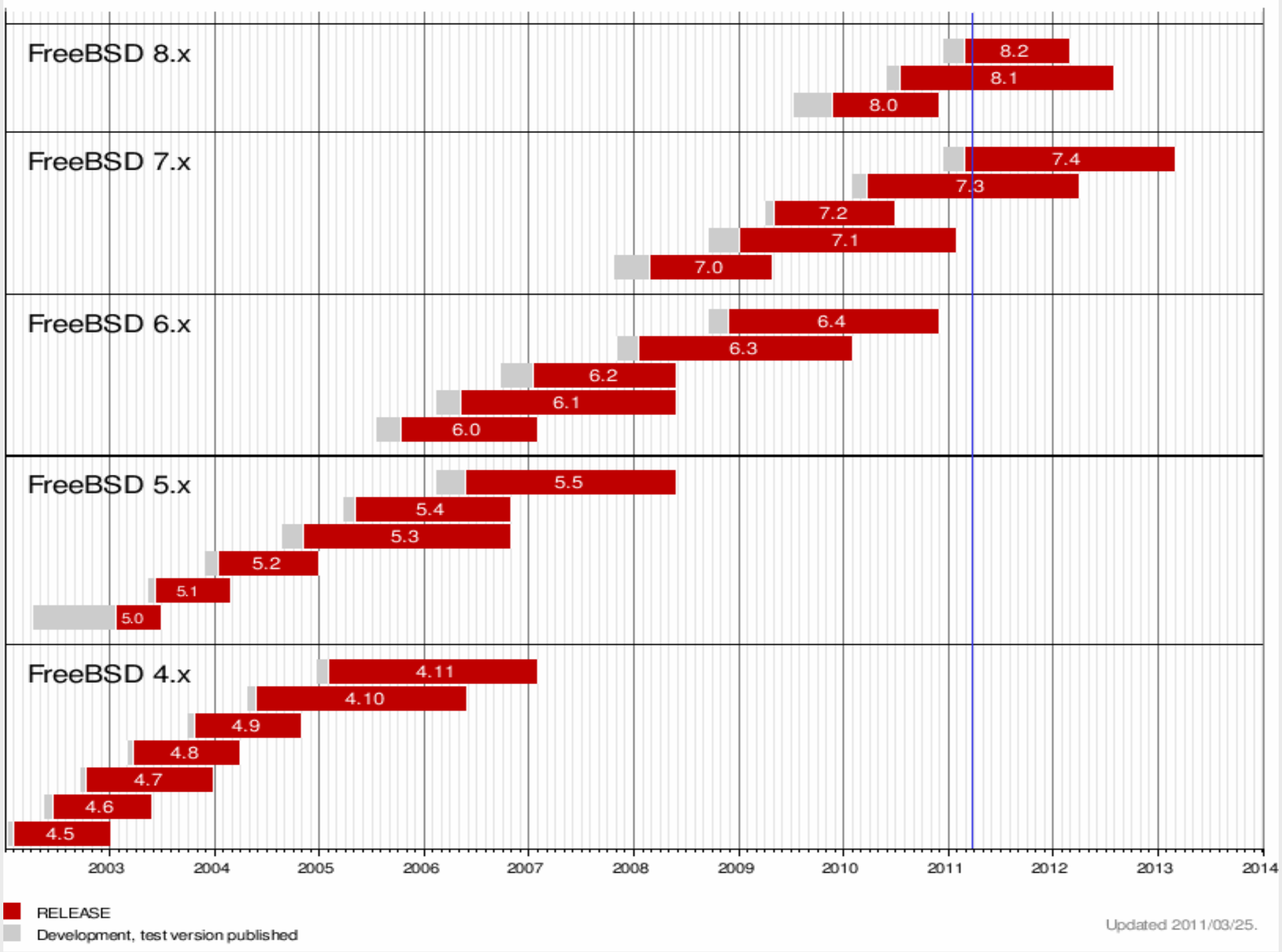
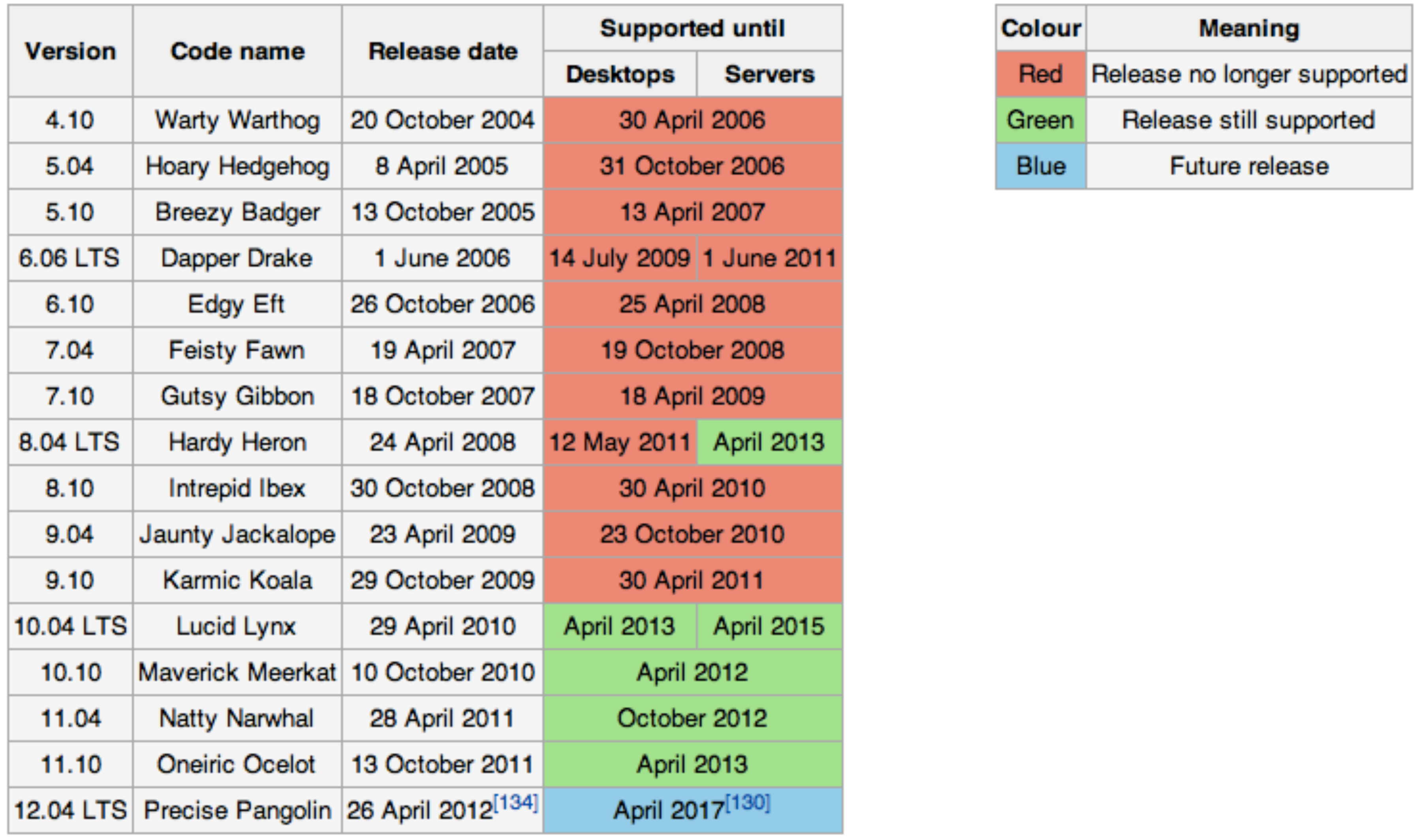


Image courtesy of Wikipedia

Ubuntu Timeline



Version	Code name	Release date	Supported until	
			Desktops	Servers
4.10	Warty Warthog	20 October 2004	30 April 2006	
5.04	Hoary Hedgehog	8 April 2005	31 October 2006	
5.10	Breezy Badger	13 October 2005	13 April 2007	
6.06 LTS	Dapper Drake	1 June 2006	14 July 2009	1 June 2011
6.10	Edgy Eft	26 October 2006	25 April 2008	
7.04	Feisty Fawn	19 April 2007	19 October 2008	
7.10	Gutsy Gibbon	18 October 2007	18 April 2009	
8.04 LTS	Hardy Heron	24 April 2008	12 May 2011	April 2013
8.10	Intrepid Ibex	30 October 2008	30 April 2010	
9.04	Jaunty Jackalope	23 April 2009	23 October 2010	
9.10	Karmic Koala	29 October 2009	30 April 2011	
10.04 LTS	Lucid Lynx	29 April 2010	April 2013	April 2015
10.10	Maverick Meerkat	10 October 2010	April 2012	
11.04	Natty Narwhal	28 April 2011	October 2012	
11.10	Oneiric Ocelot	13 October 2011	April 2013	
12.04 LTS	Precise Pangolin	26 April 2012[134]	April 2017[130]	

Colour	Meaning
Red	Release no longer supported
Green	Release still supported
Blue	Future release

Image courtesy of Wikipedia

Unix and 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

UNIX / Linux and Windows

Why does AfNOG use UNIX / Linux?

- It’s what the Internet uses to provide core services
  - 60% of all web servers are running Apache
- Much of Enterprise class computing built around UNIX / Linux
- Open Source network monitoring and management solutions
  - - Widely used
  - - Generally not available for Windows
- Router OSes are command-line and some, even, Linux

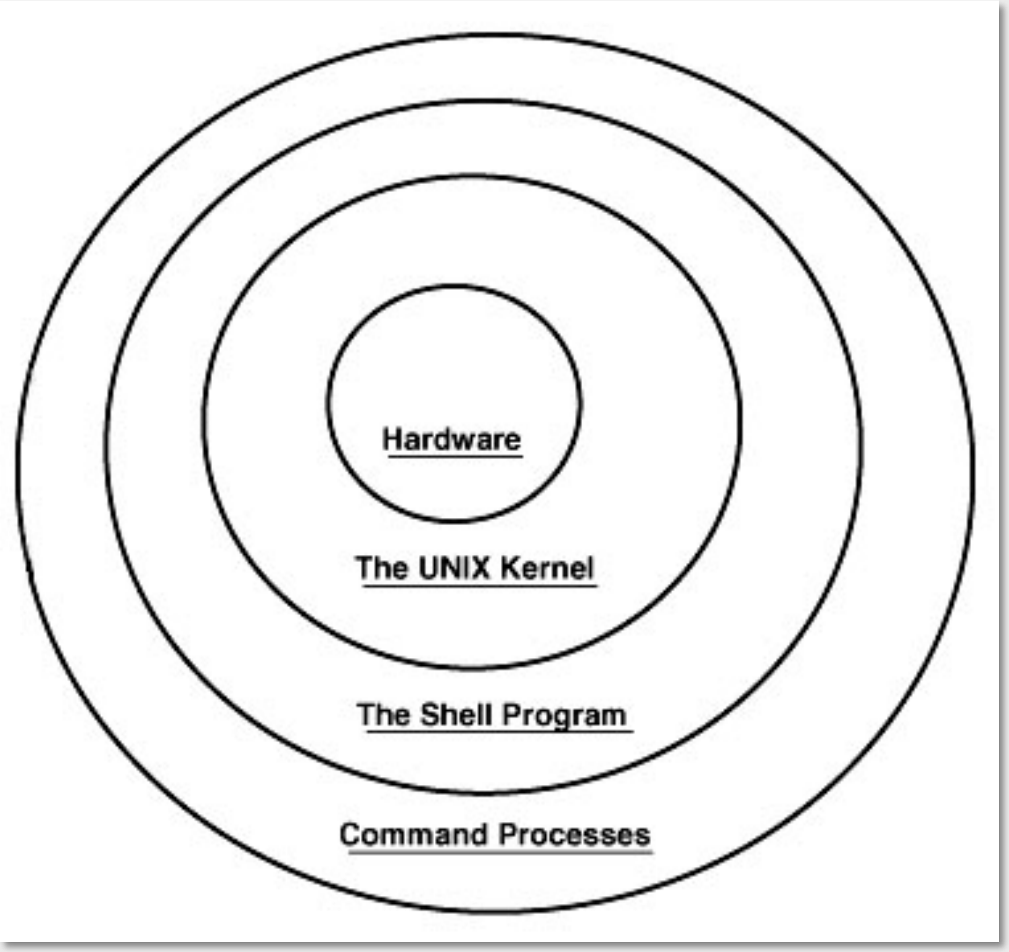
We assume

- End users are on Windows (some places Macs, too)
- Don’t expect end-users to use UNIX or Linux
- We do expect that you are likely to use Linux or UNIX

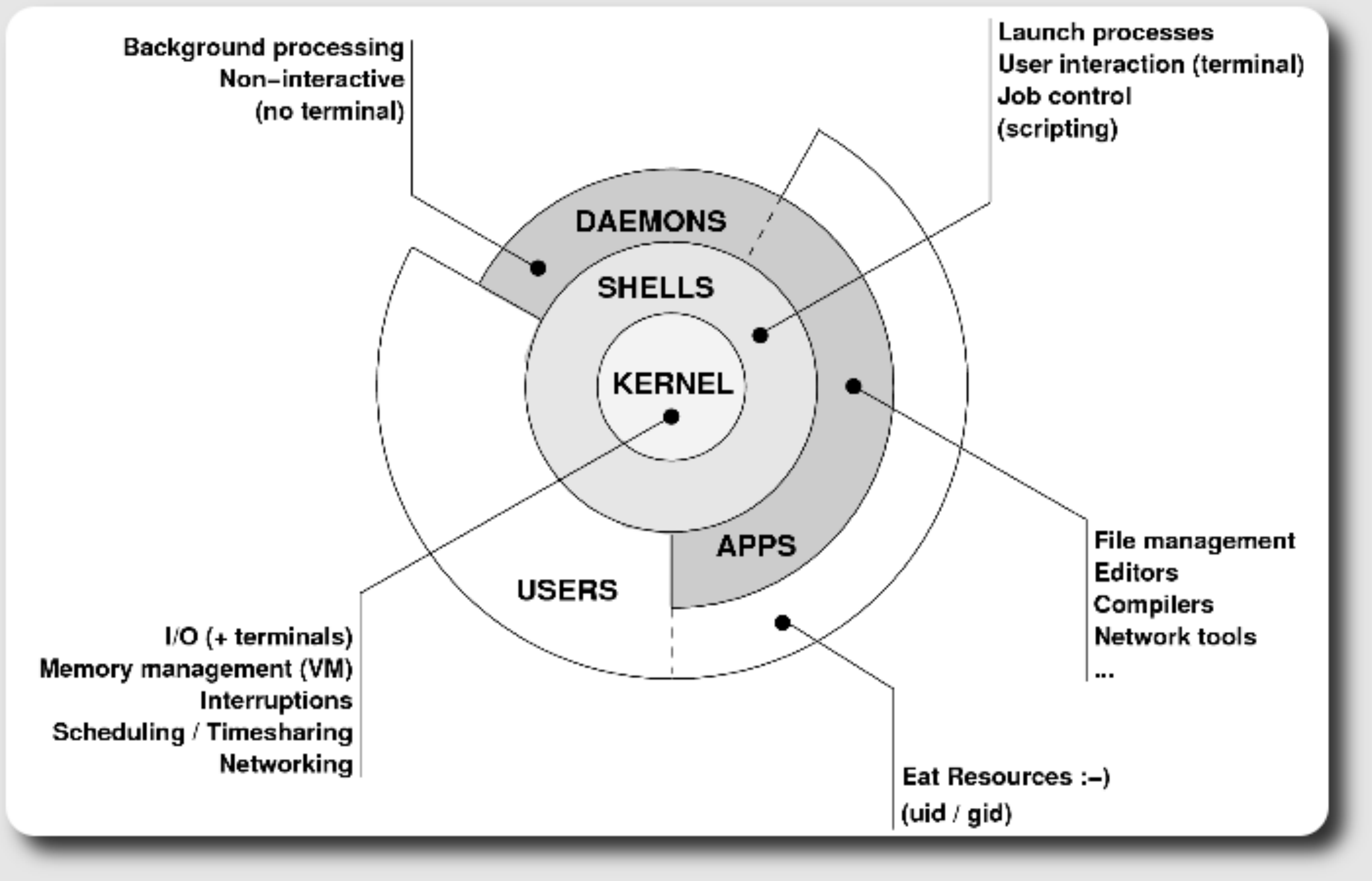
Licensing

- Windows products cost $$
- Open Source software is “free” (as in beer)
- Actual costs to implement vary widely

The Unix System Simplified



The Unix System More Detail



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

The Kernel continued

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

- Windows equivalent: command.com or command.exe

Also programming languages for scripting

- DOS/Windows equivalent: batch files, WSF, VBScript

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 (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?

?

Connect to your Virtual Linux Machine

Now you will use ssh to log in on your virtual Linux machine:

1. Windows users download putty.exe from:
   http://noc.ws.nsrc.org/downloads
2. Save putty.exe to your desktop and double-click the icon
3. Connect to pcN.ws.nsrc.org as user "*sysadm*"
   We'll do this now and instructors will help

- Mac / Linux users open a terminal window and do

```
$ ssh sysadm@pcN.ws.nsrc.org
```

Password for *sysadm* user will be given in class

Core directory refresher

/	*(/boot, /bin, /sbin, /etc, maybe /tmp)*
/var	*(Log files, spool, maybe user mail)*
/usr	*(Installed software packages)*
/tmp	*(May reside under "/")*
/home	*(user's home directories reside here)*

d

Don't confuse the the "root account" (/root) with the "root" ("/") partition.

‘Auto Defaults’ Partition FreeBSD

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.

‘Auto Defaults’ Partition Linux

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

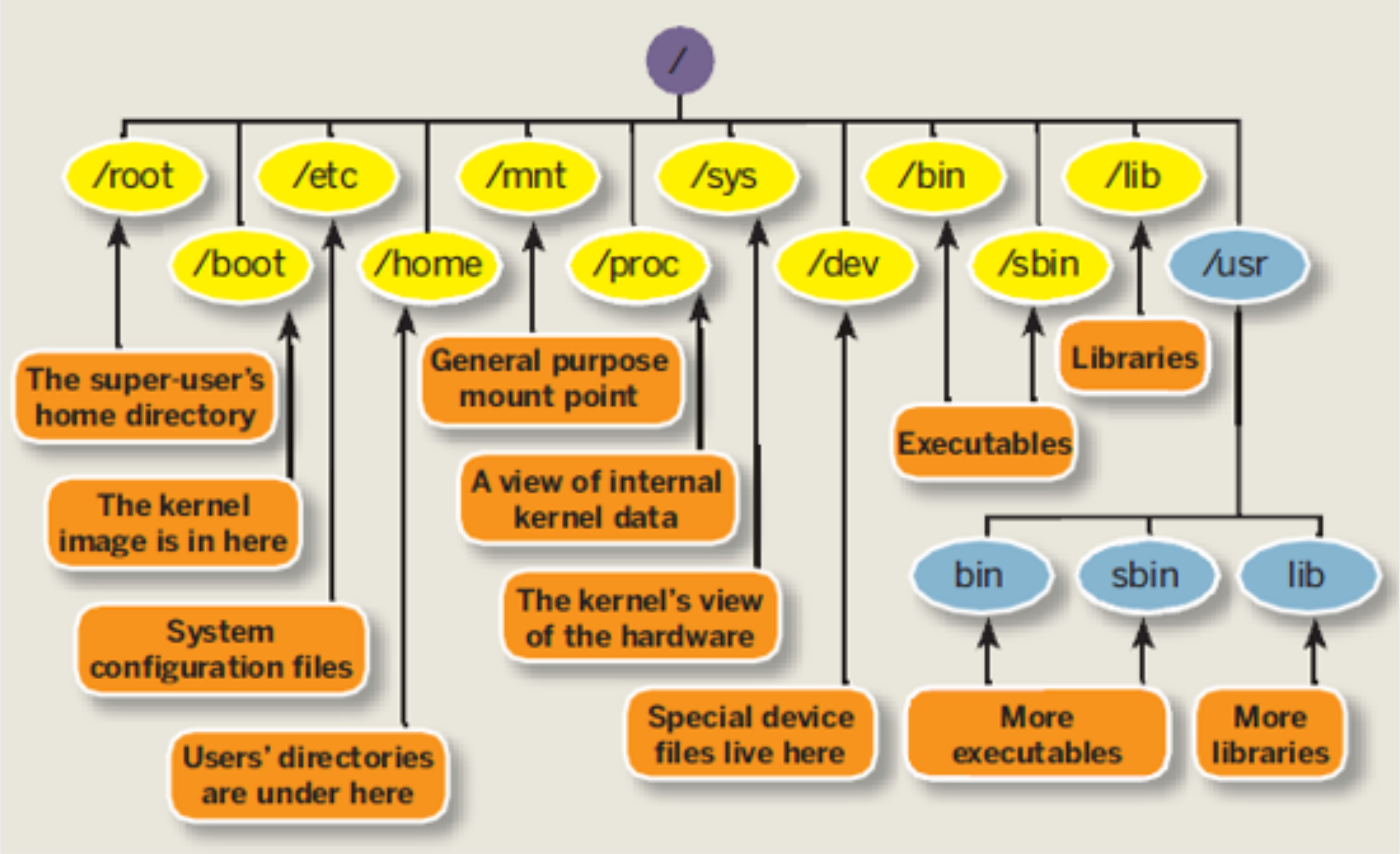
/usr contains OS utilites, third-party software

/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?

Sample file layout Linux



/
/root
/boot
/etc
/home
/mnt
/proc
/sys
/dev
/bin
/sbin
/lib
/usr
bin
sbin
lib
The super-user's home directory
The kernel image is in here
System configuration files
Users' directories are under here
General purpose mount point
A view of internal kernel data
The kernel's view of the hardware
Special device files live here
Executables
Libraries
More executables
More libraries

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 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.*

Software Installation Linux

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

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

System Startup Linux

Startup scripts

In /etc/init.d/ (System V)
In /etc/init/ (Ubuntu 12.04 LTS and Upstart)

NOTE! Upon install services run!

Controlling services

- `update-rc.d` (default method)
- Stop/Start/Restart/Reload/Status Services

```
# service <Service> <Action>
```

or, “old school”

```
# /etc/init.d/<service> <action>
```

Administration

- The use of the *root* account is discouraged. The *`sudo`* program is used 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.
- Ubuntu does `do-release-upgrade` to move to a new version.

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/

FreeBSD Resources

http://www.freebsd.org
http://forums.freebsd.org
http://www.freshports.org/
http://wiki.freebsd.org
http://en.wikipedia.org/wiki/FreeBSD

Ubuntu Resources

http://www.ubuntu.com
http://ubuntuforums.org
http://www.debian.org
http://ubuntuguide.org
http://en.wikipedia.org/wiki/Debian
http://en.wikipedia.org/wiki/Ubuntu_(Linux_distribution)

Packages & Exercises

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