Linux Systems Administration
Getting Started with Linux

Network Startup Resource Center

UNIVERSITY OF OREGON

NSRC
Network Startup Resource Center

Who Are We Teaching?

A class that has already experienced or used Linux or UNIX in the real world.

We're assuming a beginning to Intermediate level of knowledge.

Are we right?

You Need To Know:

1. **System Access**
2. **User privileges**
3. **Setting up Network**
4. **File System Layout**
5. **Editing config files**
6. **Software Management**
7. **Managing services & Processes**
8. **Checking system & memory load**

The Superuser

By default, one account has elevated privileges to issue any command, access any file, and perform every function:

<u>Superuser, a.k.a. *root*:</u>

- Technically, can change to anything – but don't

<u>Limit use of root:</u>

- Inexperienced users can cause serious harm
- Use of root for non-privileged tasks unnecessary and can be open to attack
- Security and privacy violations – root can look at anyone's files

• Limit what root can do remotely
• Ensure a strong password or no password access at all

System Access

Log in

Graphical or Command Line (GUI or CLI):

Remote Access

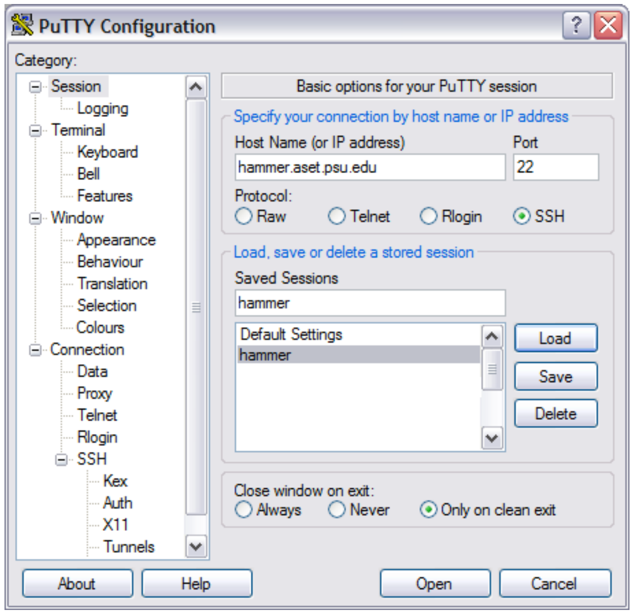
- PuTTY (Windows)
- ssh client

Physical host Access

Requirements

Username

password


PuTTY Configuration

Basic options for your PuTTY session

Specify your connection by host name or IP address

Host Name (or IP address): hammer.aset.psu.edu

Port: 22

Protocol: Raw Telnet Rlogin [✓ SSH]

Load, save or delete a stored session

Saved Sessions: hammer

Default Settings
hammer

Close window on exit: Always Never [✓ Only on clean exit]

System Access

Users

Superuser → *root*

General user → *given in class*

UNIVERSITY OF OREGON

NSRC
Network Startup Resource Center

System Access

Superuser Privileges

What usually works best is short periods of superuser privilege, only when necessary:

1) Obtain privileges,

2) complete task,

3) relinquish privileges

Most common ways are `su` and `sudo`

System Access

`su`

Short for *su*bstitute or *s*witch *u*ser

Syntax: `su [options] [username]`

- If `username` is omitted, `root` is assumed.
  Example of `su` use:

```
$ su -
```

1) Prompted for user's password. In above example *root* login scripts will execute ("`-`" option)
2) A new shell opens with the privileges of *username*, if specified, otherwise will be *root* user.
3) Once done issuing commands, type `exit`

System Access

`sudo`

Allows you to issue a single command as another user (some think of this as "*super user do*")

Syntax is:

```
sudo [options] [-u user] command
```

Again, if no user specified, root assumed

1) New shell opened with user's privileges
2) Specified command executed
3) Shell exited

Works for almost all commands

System Access

Once access is granted

```
nsrc@host1:~$
```

What the above-prompt means:

User → *`nsrc`*
Host → *`host1`*
~ → home directory
$ → general user. User *root* denoted by “`#`”

System Access: 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, Power Shell
- Linux/Unix: Shell scripting, Perl, php, python, C, etc.

Choice of similar but slightly different shells

- **bash:** "Bourne-Again Shell". POSIX standard + command history.
- **sh:** the "Bourne Shell". Standardised in POSIX
- Others: **ksh**, **tcsh**, **zsh, csh**
- bash is very common

```
#!/bin/bash
```

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

Some Useful Commands

- `ls`
- `pwd`
- `cd`
- `cat/more/less`
- `mkdir`
- `cp`
- `mv`
- `rm`
- `w/who`
- `man`
- ....

The Format of a Command

```
command [options] parameters
```

"Traditionally, UNIX command-line options consist of a dash, followed by one or more lowercase letters. The GNU utilities added a double-dash, followed by a complete word or compound word."

Two very typical examples are:

```
-h
--help
```

and

```
-v
--version
```

Command Parameters

- The ***parameter*** is what a command ***acts upon***.
- Often there are multiple parameters.
- In Unix UPPERCASE and lowercase for both options and parameters matter.
- **Spaces** ___ are ___ critical ___ !!

“`--  help`” is wrong.

“`--help`” is right.

Command Examples

Let's start simple – *Follow along as we go*:

Display a **li****s**t of files:

```
ls
```

Display a **li****s**t of files in a **l**ong listing format:

```
ls -l
```

Display a **li****s**t of **a**ll files in a **l**ong listing format with **h**uman-readable file sizes:

```
ls -alh
```

Command Examples

Some equivalent ways to do “`ls -alh`”:

```
ls -lah
ls -l -a -h
ls -l -all --human-readable
```

Note that there is no double-dash option for “`-l`”. You can figure this out by typing:

```
man ls
```

Or by typing:

```
ls --help
```

Single “-” vs Double “--”

Why would you ever use?

```
ls -l -all --human-readable
```

And not just?:

```
ls -lah
```

??

Network Setup

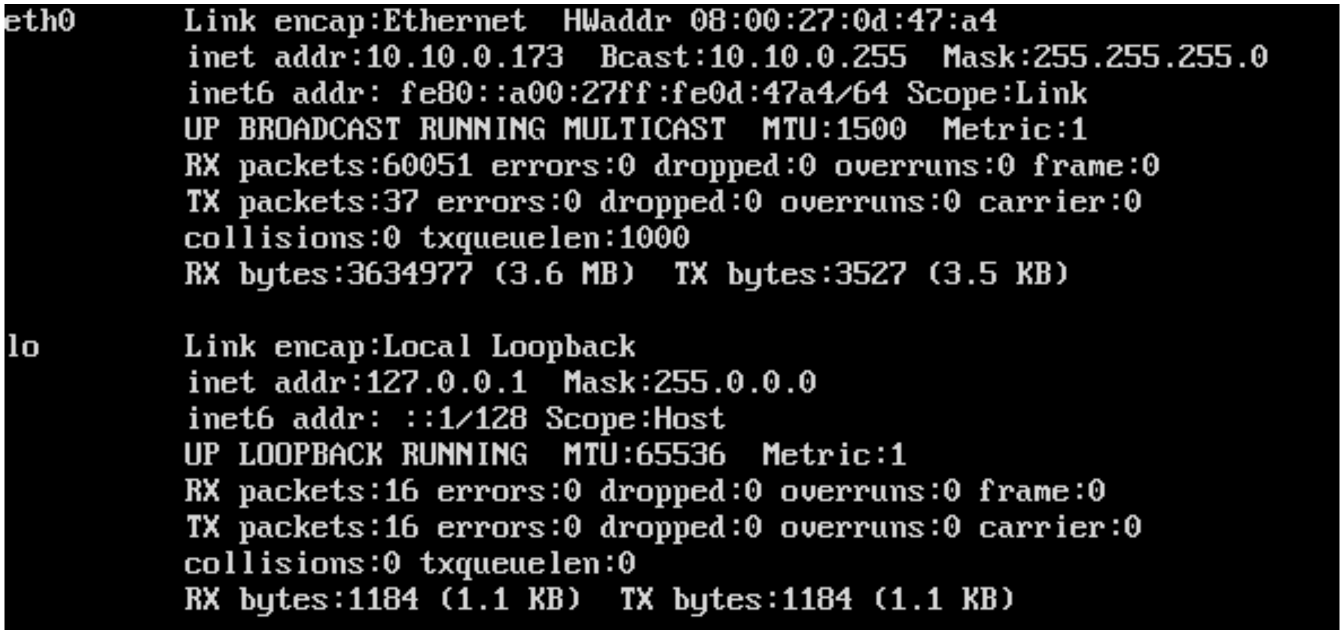
Wired Ethernet

- Displayed as `ethX` or `intX` with `X` starting at 0
- Aliases to physical adapter and driver
- Ifconfig shows network interface(s) status

Wireless Interface

- Use `iwconfig` to manage these and display info

Network Setup



```
eth0      Link encap:Ethernet  HWaddr 08:00:27:0d:47:a4
          inet addr:10.10.0.173  Bcast:10.10.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe0d:47a4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:60051 errors:0 dropped:0 overruns:0 frame:0
          TX packets:37 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3634977 (3.6 MB)  TX bytes:3527 (3.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1184 (1.1 KB)  TX bytes:1184 (1.1 KB)
```

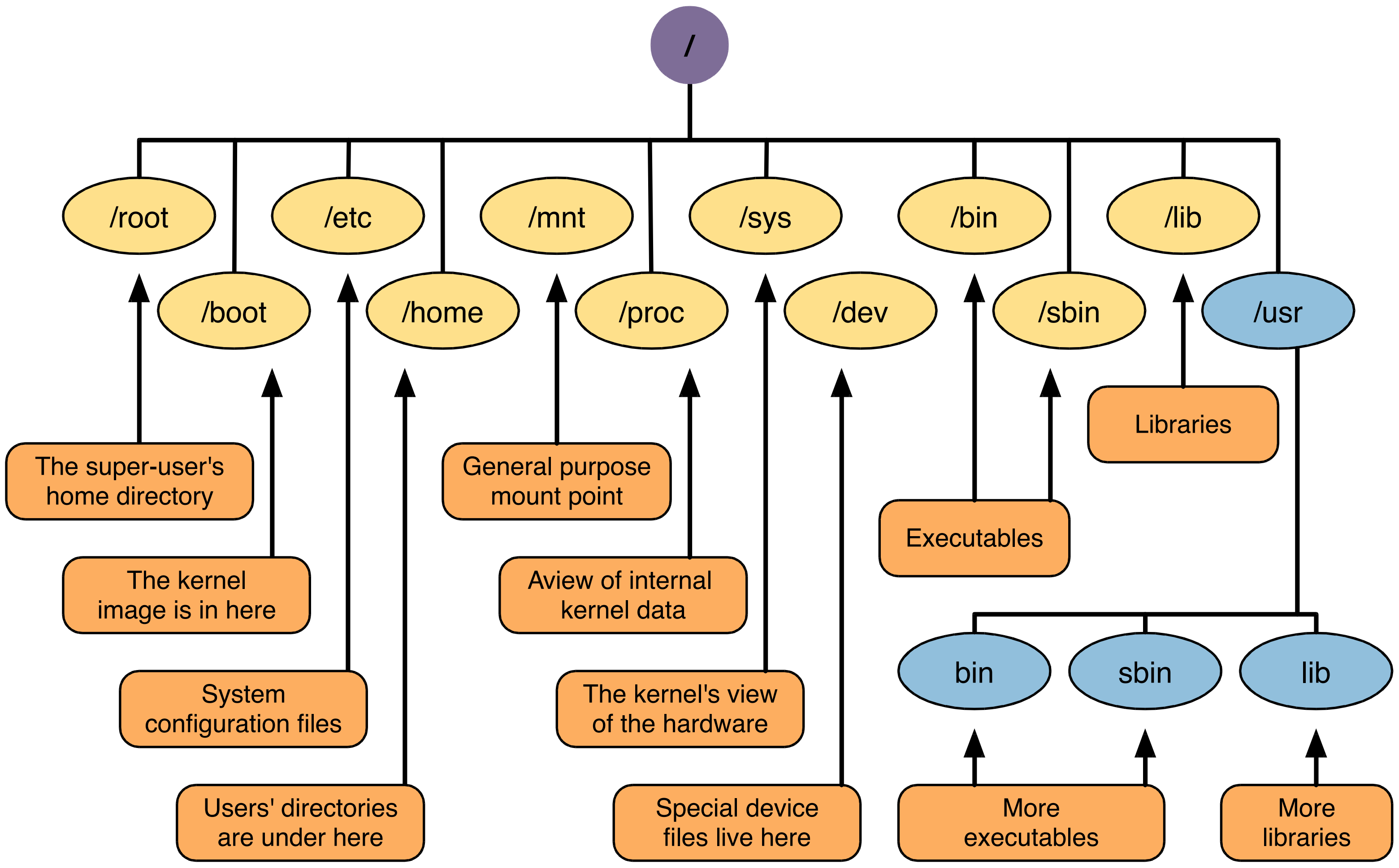
Manually setup IP Address

```
nsrc@host1:~$ sudo ifconfig eth0 10.10.0.173 netmask 255.255.255.0
```

If you need to setup IP Address permanently

```
nsrc@host1:~$ sudo vi /etc/network/interfaces
```

Linux File System Layout

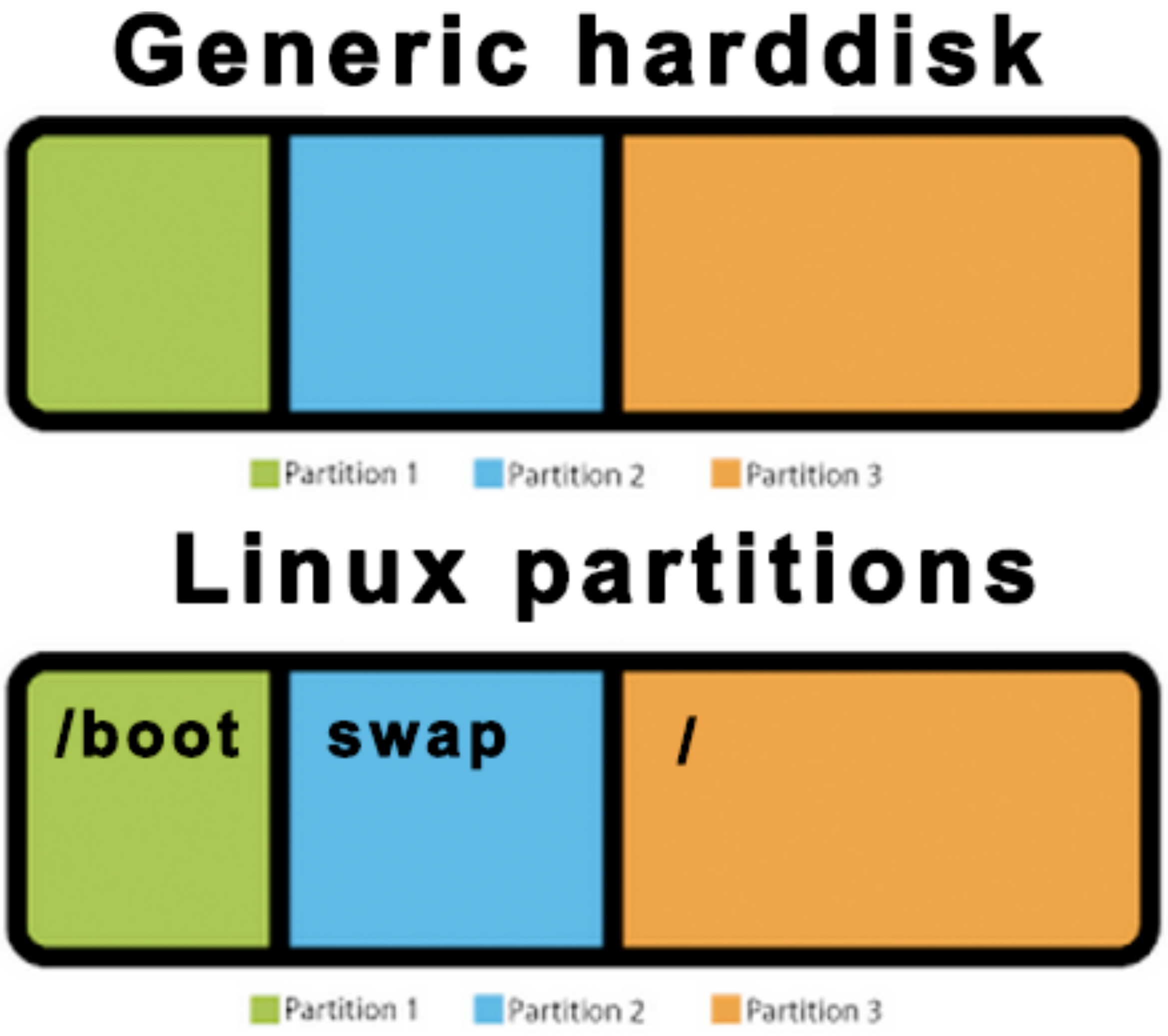

/
/root
/boot
/etc
/home
/mnt
/proc
/sys
/dev
/bin
/sbin
/lib
/usr
The super-user's home directory
The kernel image is in here
System configuration files
Users' directories are under here
General purpose mount point
Aview of internal kernel data
The kernel's view of the hardware
Special device files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

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

Partitioning Considerations

- Single large partition or multiple?
- A single partition is flexible, but a rogue program can fill it up…
- Multiple partitions provide protection, but you may need to resize later, on older filesystems, or without a “Volume Manager”
- Is /var big enough? /tmp?
- How much *swap* should you define?

Partitioning Considerations



Notes

- 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 or RAID with a separate drive.

Remember, "`rm -rf /`" on a mirror will erase everything on both disks ☺

Data Security <==> Backup

Editing Linux Files

- Be able to edit a file using vi
- Begin to understand the “language” of configuration files
- Use alternate editors: ee, joe, pico, nano, emacs, xemacs, gedit, etc.

vi Philosophy

- It's available!
- Default installed in almost all Linux/UNIX distros
- It's ubiquitous in UNIX and Linux (`visudo`, `vipw`, `vigr`, etc.)
- Excellent and fast *search* and *replace* functionality
- Not hard after initial learning curve.
- If you wanna look geeky... Use vi...

Why is vi "so hard to use"?

Like all things it's not really – once you are used to how it works.

The ***critical*** vi concept:

1. vi has two modes
2. These modes are ***insert*** and ***command***

Let's see how we use these...

vi Command and Insert Modes

<u>Swapping modes</u>

- *command mode* at file open
- To edit, switch to *insert mode*
- ESCape key exits *insert mode*
  - Returns to *command mode*

Get used to this <==> vi user!

vi Insert Mode

Enter *insert mode* (two ways):

- Press "**i**" key
  - Enter text directly after your cursor.
- Press "**o**" key
  - Adds new line *below* you cursor. Start editing.
- Press ESCape key to exit *insert mode*

vi Command Mode

Many, *many* commands in vi. Some common ones include:

- Press "**x**" to delete a character at a time.
- Press "**dd**" to delete the line you are on.
- Press "*/*", text to search for, then press <ENTER>.
    - Press "n" to find the next occurrence of text.
    - Press "N" to find previous occurrences of text.

Saving Files, or How To Exit vi

1. In vi press the *ESC*ape key to verify you are in *command mode*.
2. Type "**:**" in *command mode* to do:

:w	→ write the file to disk
:wq	→ write the file to disk, then quit
:q	→ quit the file (only works if no changes)
:q!	→ quit and lose any changes made
:w!	→ override r/o file permission if you are owner or *root* and write the file to disk.
:w!q	→ override r/o file permission if you are owner or *root* & write the file & quit.

Speed Up Editing Config Files

Press the *ESC*ape key for *command mode*, then...

1. Search for the first occurrence of a string:
   - **`/string`** → press <ENTER>
   - **“n”** → press “n” for each following occurrence
   - **“N”** → press “N” for each previous occurrence
2. Replace *all* occurrences of a string in a file:
   - **`:%s/old_string/new_string/g`**
3. Replace occurrences of string in file with confirmation:
   - **`:%s/old_string/new_string/gc`**

UNIVERSITY OF OREGON

NSRC Network Startup Resource Center

Speed Things Up Some More!

Press the *ESCape* key for *command mode,* then...

1. Go directly to a specific line number
   - **`:NN`** → press <ENTER>. If NN=100, go to line 100
2. Go to start/end of a line
   - **0** / **$** – or – **^** / **A**
3. Go to top/bottom of a file:
   - **1G** / **G** – or – *ctrl-Home* / *ctrl-End* if available
4. Copy/paste a line:
   - “yy” to yank a line, then “**p**” to place below cursor
5. Undo the last change you made (in command mode)
   - press **“u”**

Configuration File Patterns

- The most common comment character is “#”.
- After that you'll see “/* ... */” or “//”.
- There are a few others that are less common:
  : ; <!-- ... -->

Editing Configuration Files

- Some configuration files have many comments and few directives
- Others are the opposite.
- Blocks of configuration may be indicated in a programmatic manner, i.e. (Apache):

```
<VirtualHost *>

<SubSection>

directive

</SubSection>

</VirtualHost>
```

Editing Configuration Files

Another standard is to do the following:

```
## comment
## comment
# default setting=off
```

To change the default do:

```
default setting=on
```

Editing Configuration Files

Things to watch out for:

- Spaces
- Quotes and single quotes: “directive” or 'directive'
- Caps or CamelCase syntax
  - `Localhost=“myhost”`
  - `LocalHost=“myhost”`
- Line end indicator (**:** or **;**)
- New-line or continuation character “\”.

UNIVERSITY OF OREGON

NSRC Network Startup Resource Center

Debian/Ubuntu Software Management

Repositories (collections of software)

- Controlled by: `/etc/apt/sources.list`

`dpkg`

- `dpkg --get-selections, dpkg-reconfigure, dpkg-query`

`apt`

- `apt-cache, apt-cache policy, apt-cache search, apt-get, apt-get install, apt-get remove, apt-get purge, apt-get clean`
- Meta-packages: `(build-essentials, ubuntu-desktop)`

`aptitude`

- `aptitude search, aptitude clean, aptitude remove, aptitude purge`

Services Management

Startup scripts

In `/etc/init.d/` (System V)

In `/etc/init/` (Ubuntu Current / Upstart)

NOTE!

When you install packages in Debian/Ubuntu services run as soon as they are installed!

Services Management Cont.

Controlling services

Permanently set service state:

- Use `update-rc.d` command:

Stop/Start/Restart/Reload/Status Services

– `# service <Service> <Action>`

– `# /etc/init.d/<service> <action>`

Examples

```
    # update-rc.d -n apache2 enable
    # update-rc.d -f apache2 remove
```

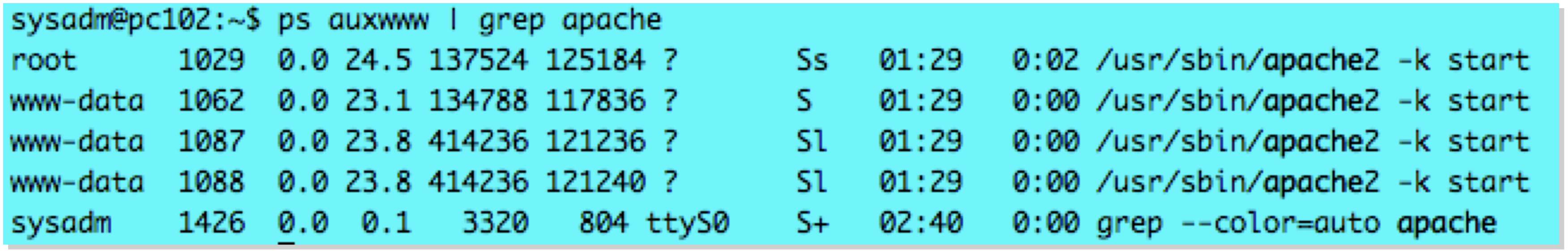
Review "`man update-rc.d`" for details

```
    # service apache2 restart
 # service apache2 status
```

Process Management

Check for a process by name

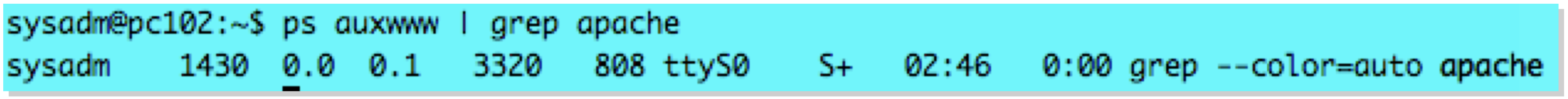
- `ps auxwww | grep apache`



```
sysadm@pc102:~$ ps auxwww | grep apache
root      1029  0.0 24.5 137524 125184 ?       Ss   01:29   0:02 /usr/sbin/apache2 -k start
www-data  1062  0.0 23.1 134788 117836 ?       S    01:29   0:00 /usr/sbin/apache2 -k start
www-data  1087  0.0 23.8 414236 121236 ?       Sl   01:29   0:00 /usr/sbin/apache2 -k start
www-data  1088  0.0 23.8 414236 121240 ?       Sl   01:29   0:00 /usr/sbin/apache2 -k start
sysadm    1426  0.0  0.1   3320    804 ttyS0   S+   02:40   0:00 grep --color=auto apache
```

Stop the process by PID (Process ID). From above listing:

- `sudo kill 1029` (why this one?)
- `Sudo kill -9 1029` (force stop if hung)



```
sysadm@pc102:~$ ps auxwww | grep apache
sysadm    1430  0.0  0.1   3320    808 ttyS0   S+   02:46   0:00 grep --color=auto apache
```

More Commands

Use these command and see what happens

- `top` (press ‘q’ to exit)
- `free -g`, then just `free`
- `df -h`
- `netstat -anp | more`

To understand what each command is doing:

- `man top`
- `man free`
- `man df`
- `man netstat`

Any Questions?