Linux System Administration

Getting started with Linux

SANOG XXV Kandy, Sri Lanka

Who We Think We Are Teaching ?

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

We're assuming an primary to Intermediate level of knowledg

Are we right?

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**

System Access

Log in

- GUI
- CLI
  - Remote Access
    - puTTY
    - ssh client
    - Etc..
  - Host Access

Requirement

- Username
- password

System Access

User

- root user (the superuser, privileged user)
  - Like “Administrator” on Windows systems
- normal user (non-privileged user)

System Access

The Superuser

- By default, one account has elevated privileges to issue any command, access any file, and perform every function
- Superuser, a.k.a. root
  - Technically, can change to anything – but don't
- Must limit use of root
  - 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

System Access

Superuser Privileges

- What usually works best is short periods of superuser privilege, only when necessary
- Obtain privileges, complete task, relinquish privileges
- Most common ways are su and sudo

System Access

su

- Short for *substitute* or *switch user*
- Syntax: `su [options] [username]`
  - If `username` is omitted, `root` is assumed
- After issuing command, prompted for that user's password
- A new shell opened with the privileges of that user
- Once done issuing commands, must type exit

System Access

sudo

- Allows you to issue a single command as another user
- Syntax:

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

- Again, if no user specified, root assumed
- New shell opened with user's privileges
- Specified command executed
- Shell exited

System Access

Access granted

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

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
- Linux/Unix: Perl, shell, php, python, C, etc.

Choice of similar but slightly different shells

- **bash:** the "Bourne-Again Shell". Combines POSIX standard with command history.
- **sh:** the "Bourne Shell". Standardised in POSIX
- Others: **ksh**, **tcsh**, **zsh, csh**

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 Example

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

Display a list of files:

```
ls
```

Display a list of files in a long listing format:

```
ls -l
```

Display a list of all files in a long listing format with human-readable file sizes:

```
ls -alh
```

Command Example

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`

Some commands accept options without the "-": e.g. `ps aux`, `tar zvxf`

Setting up Network

- By default, wired ethernet interfaces are found as ethX, with X starting at 0
- These are aliases to the actual physical adapter and driver
- Use *`ifconfig`* to know the status of your Network Interfaces
  - Alternatively use `ip link` and `ip addr`
- Wireless interfaces a bit different
  - Use iwconfig to manage these and display info

Setting up Network

- By default, wired ethernet interfaces are found as ethX, with X starting at 0
- These are aliases to the actual physical adapter and driver
- Use *ifconfig* to know the status of your Network Interfaces
- Wireless interfaces a bit different
  - Use iwconfig to manage these and display info

Setting up Network

```
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)
```

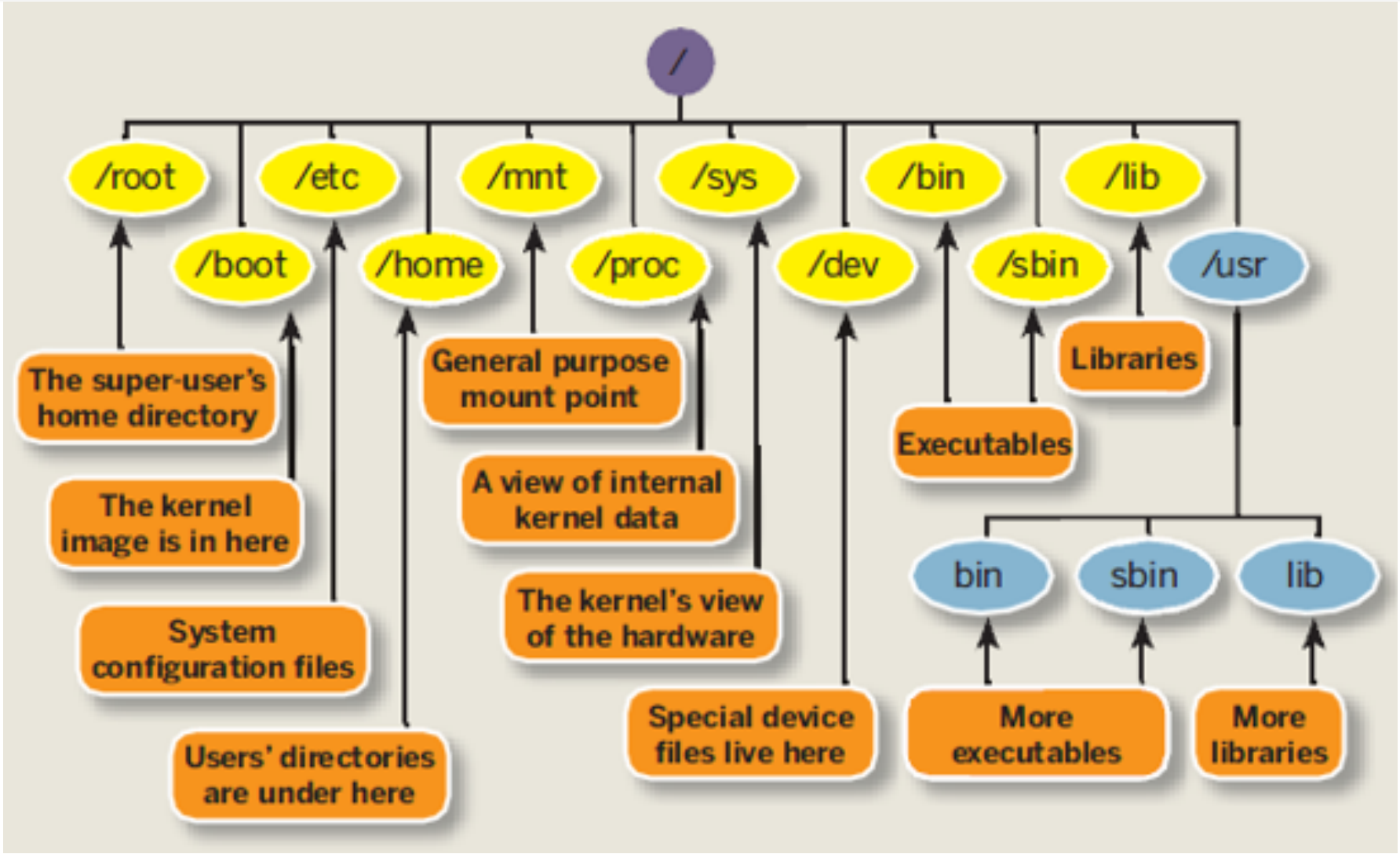
Manually setup IP Address

nsrc@host1:~$ ifconfig eth0 10.10.0.173 netmask 255.255.255.0

If you need to setup IP Address permanently

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

Linux File System Layout



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 provides a more “protected” approach, 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?

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

Commands ...

- du
- df
- mount

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!
- Wait, what was that? Oh yeah, it's available!
- It's has some very powerful features.
- It's ubiquitous in UNIX and Linux (`visudo`, `vipw`, `vigr`, etc.)
- Not that hard to learn after initial learning curve.
- Impress your friends and family with your arcane knowledge of computers.

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

Swapping modes

- When you open a file in vi you are in *command mode* by default.
- If you wish to edit the file *you need to switch to insert mode first*.
- To exit *insert mode* press the ESCape key.
- If you get used to this concept you are halfway done to becoming a competent vi user.

vi insert mode

Two common ways to enter insert mode upon opening a file include:

- Press the “i” key to start entering text directly after your cursor.
- Press the “o” key to add a new line *below* you cursor and to start adding text on the new line.
- Remember, to exit *insert mode* press the ESCape key at any time.

vi command mode

Many, many commands in vi, but some of the most common and useful are:

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

Saving a file or how to exit from vi

1. In vi press the *ESC*ape key to verify you are in command mode.
2. Depending on what you want to do press:
   - **: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* and write the file to disk and quit.

Speed-Up your config file editing!

1. In vi press the *ESC*ape key to verify you are in command mode.
2. To search for the first occurrence of something:
   - **`/string`** → press <ENTER>
   - **“n”** → press “n” for each following occurrence
   - **“N”** → press “N” for each previous occurrence
3. To replace *all* occurrences of a string in a file:
   - **`:%s/old_string/new_string/g`**
4. To replace *all* occurrences of a string in a file:
   - **`:%s/old_string/new_string/gc`**

Speed things up some more!

1. In vi press the *ESC*ape key to verify you are in command mode.
2. Go directly to a specific line number
   - **`:NN`** → press <ENTER>. If NN=100, go to line 100
3. Go to start/end of a line
   - press *Home* or press *End* on your keyboard
4. Go to top/bottom of a file:
   - press *ctrl-Home* or press *ctrl-End* on your keyboard
5. Undo the last change you made (in command mode)
   - **press "u"**

Configuration file patterns

There are patterns to how configuration files work:

- The most common comment character is “#”.
- After that you'll see “/* .... */” or “//”.
- There are a few others, but they are less common.

Editing configuration files cont.

Some configuration files have lots of comments and few directives. Others are the opposite.

Blocks of configuration may be indicated in a programmatic manner, i.e.:

```
<VirtualHost *>
<SubSection>
directive
directive
</SubSection>
</VirtualHost>
```

Editing configuration files cont.

Another standard is to do the following:

```
## comment

## comment

# default setting=off
```

To change the default do:

```
default setting=on
```

Editing configuration files cont.

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 “\”.

Software Management

Software management

Command Line

- **`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)
- repositories – Controlled by */etc/apt/sources.list*
- **`aptitude`**
  - `aptitude search, aptitude clean, aptitude remove, aptitude purge`

Services Management

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>
```

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
```

Some More ...

Use the following command and see what happens

- `top` Press ‘q’ to exit
- `free -g`
- `df -h`
- `netstat -anp |more`

Questions

?