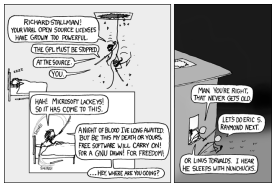
Chapter 6 Linux File System

Monday, February 9, 2015 9:25 AM



Objectives:

- Define hierarchical filesystems and files
- Determine the difference between absolute and relative Path
- Learn about the 3 "P's" of Linux
- Understand file permissions and ownership

Review - see Next week's notes (the tab on OneNote) (week 5)

Filesystems

↳ Basically an "index" to all your data on a harddisk.

↳ How the OS can access or modify data without having to know exactly about the underlying hardware

↳ Common Filesystems

→ NTFS

→ Ext2, Ext3, Ext4

→ Btrfs, Zfs

→ Fat32

→ HFS, UFS

Hierarchical Filesystem

↳ An upside down tree

↳ Directories then have subdirectories

↳ Example /home contains all user subdirectories

/home/aplusstudent

/home/jhajek

/home/soon so forth

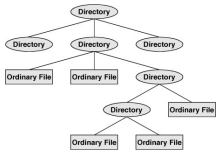
Ordinary Files and Directory Files

- Everything in Linux is a File

↳ Directories are files

↳ Files are files too.

↳ Files live under directories



File names

- Upto 255 characters
- No SPACES! EVER! (Trust me) http://www.urbandictionary.com/define.php?term=Linux+sucks+balls
- Files under a directory cannot have same name as parent directory

  /home/todd/todd (not allowed)
- Filenames are CASE SENSITIVE

  /home/todd != /home/Todd (Mac + Windows opposite
- Linux does not have file extensions by default

  ↳ Windows does

  ↳ Linux extensions are by convienience

  ↳ Some common ones

  ↳ .c → C programing file

  ↳ .t~~xt~~ → text file

  ↳ .Z → Compressed file using compress tool

↳ .gz → gnu zipped file

↳ .tgz or .tar.gz → a tar archived file (like an iso) the compressed with gzip

↳ .bz2 bzip

↳ file names can be hidden by prefixing a "."

↳ try it

↳ on a command line type

touch file1

touch .file2

ls (do you see .file2?)

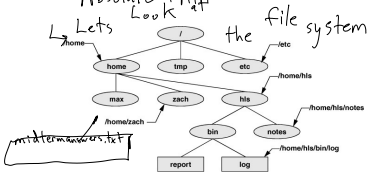
ls -la (do you see .file2?)

PATH → Very important

- This is what makes or breaks people using Linux

↳ Absolute PATH vs Relative PATH

↳ Lets look at the file system



▭ = file

⬭ = directory

↳ the top is / → pronounced "slash" or "root" (rüt)

↳ below "/" are system directories

↳ each directories has sub directories and files

↳ each file and directory on the system then can be referenced by it's absolute path from "/" down to its location

↳ each file can be referenced by its location relative to your current location.

↳ Let's try it:

What is the absolute path of the file "report"?

What is the absolute path of the file "midtermanswers"?

Note:

↳ Absolute PATH always starts with "/"

↳ the command pwd shows you your present working directory

Try it: the pwd

Also (tilde) key is a short cut to your home directory.

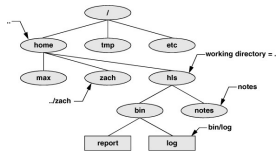
Relative PATH

→ You are able to descend the tree via "/"

→ You can ascend the tree by using ".." infront of

See slash chart

→



../../

If my pwd /home/zach

I use the cd's command (**change directories**)

If I want to change my location (pwd) to /home

type pwd cd ../ (this takes me up 1 level)

cd ~ → type cd ../../ to go up two levels

Chain Up + Down together (assuming pwd is zach)

../home/max (go up 1 level to home then down

(tilde) is a shortcut into max directory
to /home/aplusstudent

Questions?

Working With Directories

↳ mkdir (pronounced "make-durr")

↳ will create a directory
↳ try it! (From Command Line)
↳ pwd

mkdir literature
ls ~l
ls -F

↳ Try absolute file creation
↳ mkdir /home/aplusstudent/literature/good

↳ mkdir -p /home/aplusstudent/music/good
will create the music directory as well as the good
(-p directory)

Remove Directory

rmdir → only deletes empty directories will delete
rm -rf (recursive + force) a directory w/content
rm -rfiv music/good

(~) (move and copy)

mv & cp that
can move we understand PATH files around

Try it: (assume you are in ~ or your home directory)
mkdir literature
touch names temp books ~ /home/aplusst/
mv names literature → what are we doing here?
cd literature
ls

pwd
mv ../temp ../../home → what are we doing here?
cd /home
ls cd ~
cp books literature cp books /tmp.

ls /tmp

Getting Comfortable With the House

. Remember there is a "Linux Language"
. Every Linux Filesystem has a standard hierarchy
. Linux Foundation created a standard
↳ 1995

http://www.linuxfoundation.org/collaborate/workgroups/fhs

↳ you could officially call yourself "Linux"
↳ 1 These things needed to be supported so that
↳ 2
3

. Every Linux system "distro" Not these
↳ Mint Mint basically is the file system use will always have this

↳ Hence you need to memorize the second row
↳ Note Fedora Modifys this a bit but leaves a "shim"
↳ https://wiki.fedoraproject.org/wiki/Features/UsrMove (top of tree)
↳ Let's describe them (go ahead and "cd" to these)
/ bin → to be where all system executables (like ls, date, link...) were stored

/dev → Now its a ... locates all devices on the system (mouse, keyboard, screen, HDD, ...) so /usr/bin gives them a file name and bandwidth access

/etc → system and software configuration files are stored. where all

/etc/X11 → where all the system "X" windows conf files

/home → where all user (programs, mounts, server) are stored. You own this space. start to have you go here...)

/lib /lib64 → link to /usr/lib /usr/lib64

/mnt → space to mount new devices

cat /proc/cpuinfo

/proc → Kernel virtual and process information filesystem (rebuilt every time at boot)

/sbin → link to /usr/sbin

/tmp → Place to store temporary files on system (cleared out reboot)

/usr → Contains information used by operating system

/usr/bin → Standard Linux programs

/usr/sbin → Utilities (GUI tools for system administration and distro specific tools)

/var → system data

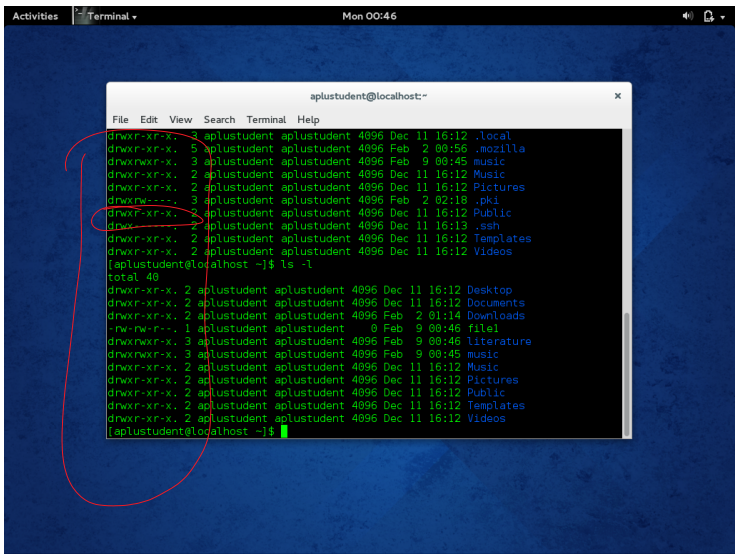
/var/log → critical logs for websites and know this location.

Access permissions

↳ Since every File is a File

→ each File has permissions on how + who can access it

→ permissions can be seen by typing ls -l



↳ Permissions break into 3 categories

↳ Owner
↳ group
↳ other (or "world")

Owner Group other

-rw- rw- r-- (see file 1 from above screenshot)

File type - = file
d = directory

Permissions are:

r = read 4 They have values
w = write 2
x = execute 1

Permissions can then be "spoken" or modified by a numeric value.

↳ example: rw- rw- r-- = 664
4+2+0 4+2+0 4+0+0

↳ This allows you to control access to files

↳ The chmod command lets you change file permissions ls -l

↳ chmod 660 file1 (pronounced "chuh-mod" not "huh-" or "shi-" or "chi-")
Try it

↳ touch secretcode.txt
ls -l (what is secretcode.txt permission?)
chmod 755 secretcode.txt
ls -l

ls -l

chmod 600 secretcode.txt (now who alone has what permissions?)

Root

↳ remember Root user can trounce all permissions

↳ Try this

echo "Hello" > lamb.txt

cat lamb.txt

ls -l

chmod 000 lamb.txt (what happens?)(why?)

cp lamb.txt ../

cat lamb.txt (what happens?)(why?)

sudo cat lamb.txt (what happens? why?)

Mode	Meaning
777	Owner, group, and others can read, write, and execute file
755	Owner can read, write, and execute file; group and others can read and execute file
711	Owner can read, write, and execute file; group and others can execute file
644	Owner can read and write file; group and others can read file
640	Owner can read and write file, group can read file, and others cannot access file

↳ There is also a chown command

↳ to change owner and group

---

3 P's

All problems in Linux can be described by the 3 P's

Path

Permission

Pendencies

de

Summary

↳ Linux has a heirarchy or upside down tree

↳ All files accessed by absolute and relative pathnames

↳ All file systems have a standard file structure

/

/etc /usr /home etc.....

↳ All files have 3 types of file access

And groups of access

be represented numerically

↳ Can Remember 3 P's

Questions?