H16S35 - Managing a web server

2
- Linux files
- Permissions

michael.ferrie@edinburghcollege.ac.uk

CONTENTS

Linux files

michael.ferrie@edinburghcollege.ac.uk

Unix-like file systems

- In Unix and operating systems inspired by it, the file system is considered a central component of the operating system
- It was also one of the first parts of the system to be designed and implemented by Ken Thompson in the first experimental version of Unix, dated 1969
- The filesystem appears as one rooted tree of directories, instead of addressing separate volumes such as disk partitions, removable media, and network shares as separate trees (as done in Windows) , such volumes can be mounted on a directory, causing the volume's file system tree to appear as that directory in the larger tree
- The root of the entire tree is denoted by a single /

Unix-like filesystem

/   "root"

/bin
"essential user
command binaries"
- bash
- cat
- chmod
- cp
- date
- echo
- grep
- gunzip
- gzip
- hostname
- kill
- less
- ln
- ls
- mkdir
- more
- mount
- mv
- nano
- open
- ping
- ps
- pwd
- rm
- sh
- su
- tar
- touch
- umount
- uname

/etc
"configuration files
for the system"
- crontab
- cups
- fonts
- fstab
- host.conf
- hostname
- hosts
- hosts.allow
- hosts.deny
- init
- init.d
- issue
- machine-id
- mtab
- mtools.conf
- nanorc
- networks
- passwd
- profile
- protocols
- resolv.conf
- rpc
- securetty
- services
- shells
- timezone

/sbin
"essential system
binaries"
- fdisk
- fsck
- getty
- halt
- ifconfig
- init
- mkfs
- mkswap
- reboot
- route

/usr
"read-only user application
support data & binaries"

/usr/bin
"most user
commands"

/usr/include
"standard include
files for 'C' code"

/usr/lib
"obj, bin, lib
files for coding
& packages"

/usr/local
"local software"
/usr/local/bin
/usr/local/lib
/usr/local/man
/usr/local/sbin
/usr/local/share

/usr/share
"static data sharable
accross all architectures"
/usr/share/man
"manual pages"

/var
"variable data files"

/var/cache
"application
cache data"

/var/lib
"data modified as
programmes run"

/var/lock
"lock files to track
resources in use"

/var/log
"log files"

/var/opt
"variable data for
installed packages"

/var/spool
"tasks waiting to
be processed"
/var/spool/cron
/var/spool/cups
/var/spool/mail

/var/tmp
"temporary files saved
between reboots"

/dev
"device files
incl. /dev/null"

/home
"user home
directories"

/lib
"libraries &
kernel modules"

/mnt
"mount files for
temporary
filesystems"

/opt
"optional software
applications"

/proc
"process & kernel
information files"

/root
"home dir. for
the root user"

5

etc and var (config & variable files)

Etc contains config Files
- Edit to configure (backronym)
- /etc/myapplication/myapplication.conf
- Usually instructions or options to comment in and out

Var contains variable files that are likely to change, such as log files or html/php/css files
- /var is a standard subdirectory of the root directory in Linux and other Unix-like operating systems that contains files to which the system writes data during its operation

lib and home (library & user files)

Lib contains libraries, /usr/lib contains dynamic libraries and static files needed in the boot process
- You'll never find an executable at /bin or /sbin that needs a library that isn't in /lib
- Kernel modules (device drivers) are under /lib

Home contains users Home Directories, the user's (real human beings) home directories.
- Software packages should never install files here (in installation time) E.g /home/bob/Documents/bobs_document.pdf is the equivalent of - C:\This PC\Users\Bob\Documents\bobs_document.pdf

Permissions

Absolute Paths

- An absolute path is defined as specifying the location of a file or directory from the root directory(/)
- In other words,we can say that an absolute path is a complete path from start of actual file system from / directory
- Always starts with /
- /home/bob/Documents
- /var/log/apache2/other_vhosts_access.log
- To show a files absolute path you can use:
        readlink -F {filename}

Relative Paths

- A relative path is defined as path related to the present working directory(pwd)
- Suppose I am located in /var/log and I want to change directory to /var/log/kernel
- I can use relative path concept to change directory to kernel
- I do not have to specify the entire path each time I want to change directory
- We can use an absolute path from any location
- If you wish to use relative path you must be in a directory that relates to your present working directory, you could call this a relative directory

Apache's Main Files

APACHE
HTTP SERVER

- /etc/apache2/apache2.conf - main config file
- /etc/apache2/ports.conf - layer 4 port settings
- /etc/apache2/sites-available - site config files
- /etc/apache2/sites-enabled - running websites
- /var/www/html/* - where we keep html/css/js files
- /var/log/apache2/error.log - the error log
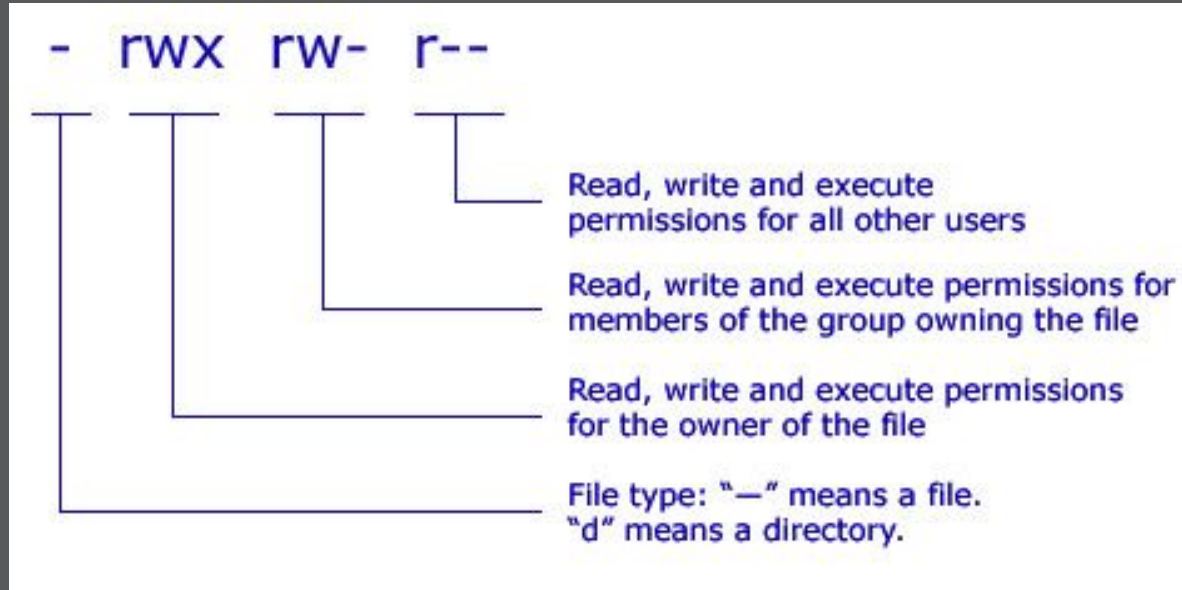- /var/log/apache2/access.log - the access log

Apache Config File

- Apache HTTP Server is configured by placing directives in plain text configuration files the main configuration file is usually called httpd.conf (apache2.conf in Debian)
- The location of this file is set at compile-time but may be overridden with the -f command line flag
- Other configuration files may be added using the Include directive, and wildcards can be used to include many configuration files
- Any directive may be placed in any of these configuration files
- Changes to the main configuration files are only recognized by apache when it is started or restarted

Apache Config Syntax

- Directives in the configuration files are case-insensitive, but arguments to directives are often case sensitive
- Lines that begin with the hash character "#" are considered comments and are ignored
- Comments may not be included on the same line as a configuration directive
- White space occurring before a directive is ignored, so you may indent directives for clarity
- Blank lines are also ignored
- You can check your configuration files for syntax errors without starting the server by using **apachectl configtest**

RWX

- rwx rw- r--

Read, write and execute permissions for all other users

Read, write and execute permissions for members of the group owning the file

Read, write and execute permissions for the owner of the file

File type: "—" means a file. "d" means a directory.

Chmod Example

- The permissions of a file are called the mode of the file, read mode write mode and execute mode, change mode changes these, which is shortened to chmod:
- chmod 762 myfile.txt
- ls –l myfile.txt
- -??? ??? ???
- To work this out, convert each number to binary, read is 4, write is 2 and execute is one:
- 7 in binary is 111, 6 in binary is 110, and two is 010
- `111 – 110 – 010`
- `rwx – rw– – –w–`

chown / chgrp

To change the owner of a file:
chown {{user}} {{path/to/file}}
● Change the user and group of a file:
chown {{user}}:{{group}}{{path/to/file}}

Recursively change the owner of a folder:
chown -R {{user} {{path/to/folder}}
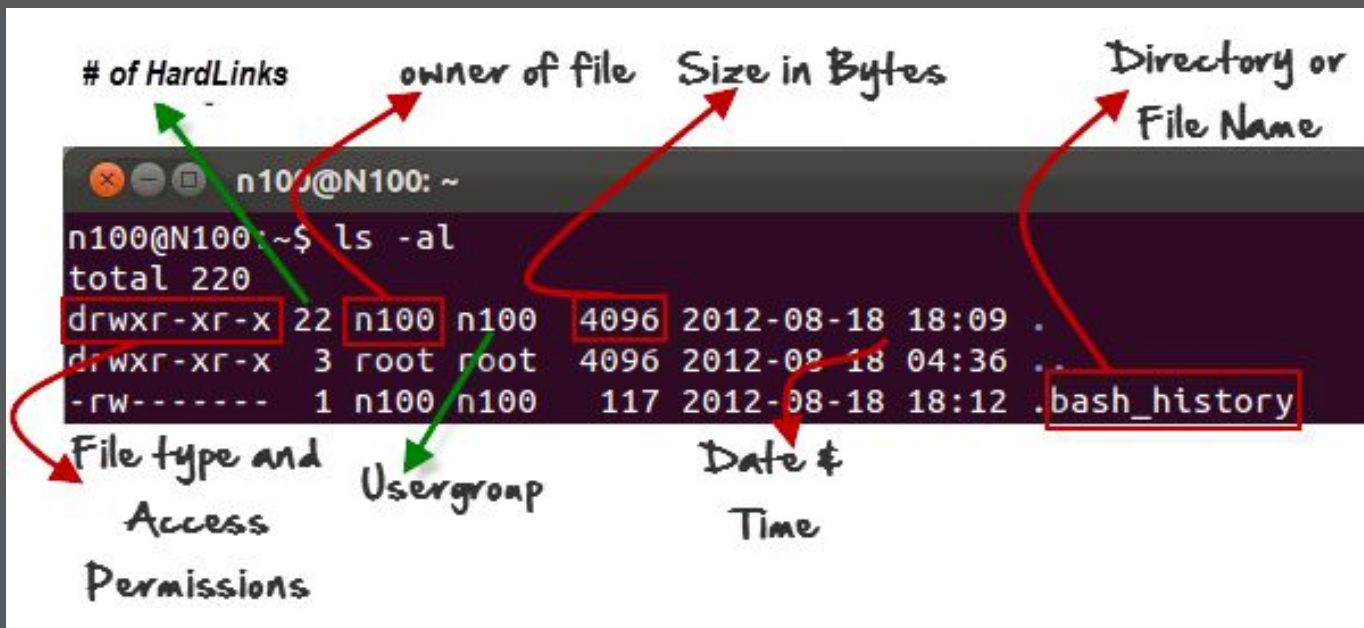● Change groups with chgrp:
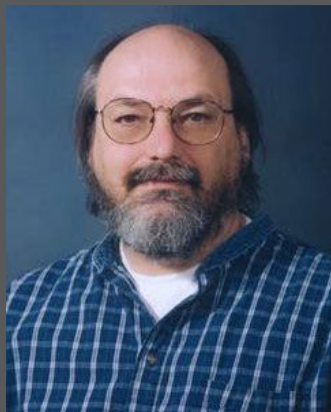chgrp {{group_name}} {{file_name}}
e.g. - chgrp accounts money.csv
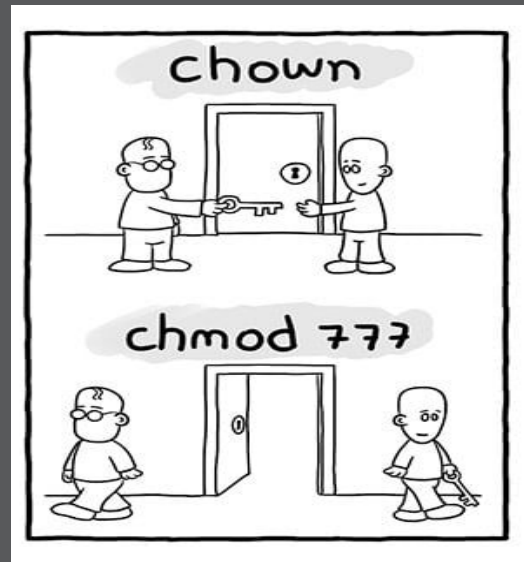
Questions?

- chmod 777 myfile.txt
- chmod 745 myfile.txt
- chmod 752 myfile.txt
- chmod 267 myfile.txt

- Work out these four examples, remember r=4, w=2 and x=1 just convert them to binary and you'll get the answer easily!

ls -l

Suggested Reading - [Unix History](#)

[Thompson & Ritche](#) - Dennis Ritchie passed away a few years ago but Ken Thompson is still alive - both are very important in computing

Thanks for listening
Any questions?

michael.ferrie@edinburghcollege.ac.uk