[StuCo 98008]
GNU/Linux for Beginners

Session 3

The importance of text files in GNU/Linux
Handling files and folders
Important console tools

By the end of this lecture you will know

- How important simple text files are
- How to handle files and directories from the console
- How to use the **vi** editor
- The power of GNU textutils

Text Files Control Everything

- Configuration options are saved in simple text files.
  - Device drivers (*/etc/modules.conf*)
  - Server options (*/etc/sendmail.conf*)
  - Application defaults (*/etc/cdrecord.conf*)
- Why?
  - Easy to handle, simple, well-known format (ASCII)
  - Distributed vs centralized storage
    - Distributed information under /etc
    - Centralized information (M$ Windows registry)

Reading Text Files from the Console

- **cat** <filename> **:** dumps contents of file to screen
  - Useful for quickly viewing small files
- **less** <filename> : view a text file interactively
  - Scroll up/down, search for text, etc.
- **vi** <filename> : open a file for editing

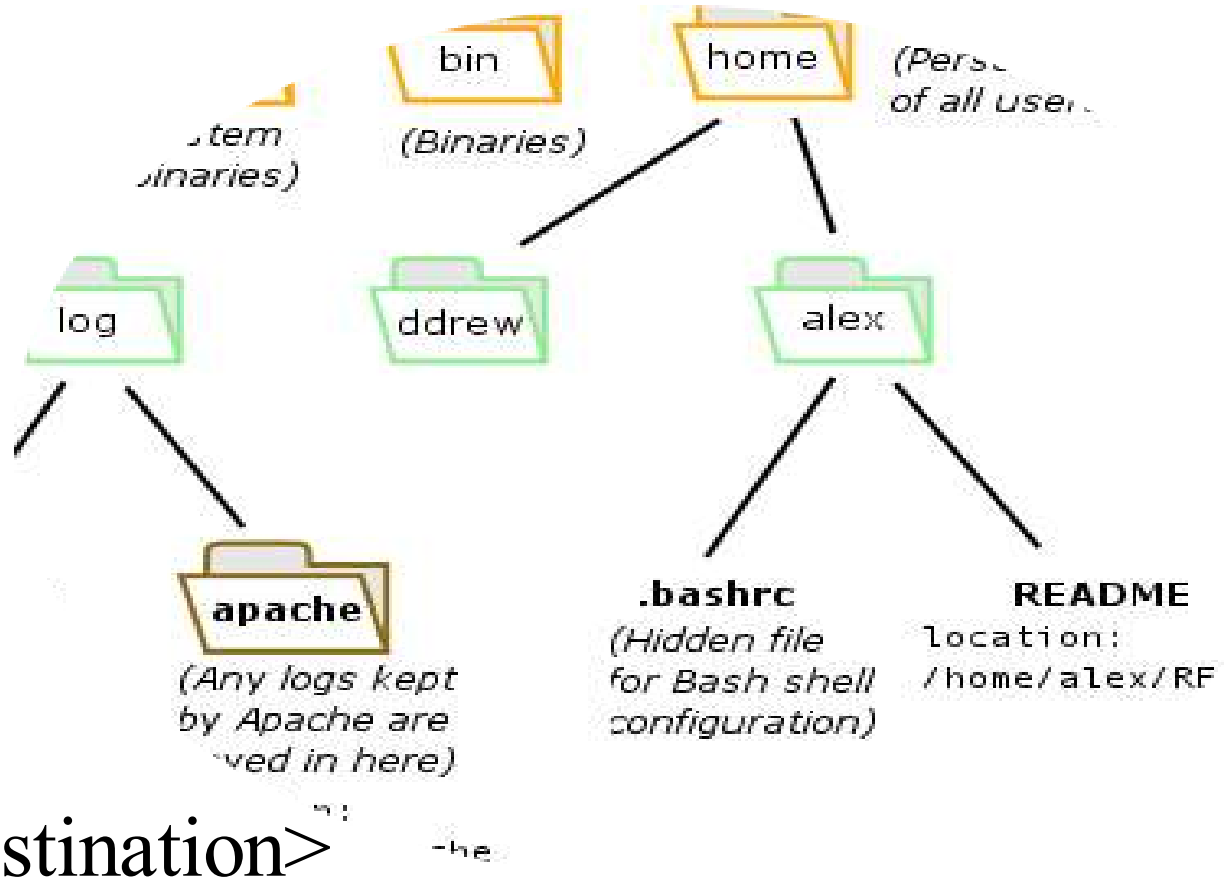
Using **vi**

- By default, the editor starts in *command mode*.
- Hit **<I>** to enter *insert mode*.
  - Insert text
  - Delete text
- Hit **<ESC>** to return to command mode.
  - Issue commands that affect the entire document

vi commands

- **:wq** (write changes to file and quit)
- **:q!** (quit without saving changes)
- **:set nu** (sets numbering of lines on)
- **yy** (yank/copy the current line)
- **p** (paste the copied line(s)) right below the cursor
- Anything potentially harmful (e.g. quitting without saving changes, writing to a read-only file), has to be forced with an exclamation mark **(!)**

Copying Files and Directories



- **cp** <source> <destination>
- **cp -R** <source_folder> <destination>

Moving Files and Directories

- **mv** <source> <destination>
  - This works for single files and entire directories alike, no need for the -R flag.
- If destination doesn't exist, **mv** practically functions as “rename”.

```
alex@debian:~$ ls -l grid*
-rw-r--r--    1 alex     alex        80714 Sep  7 19:34 grid_document.pdf
alex@debian:~$ mv grid_document.pdf grid_primer.pdf
alex@debian:~$ ls -l grid*
-rw-r--r--    1 alex     alex        80714 Sep  7 19:34 grid_primer.pdf
alex@debian:~$
```

Finding Files with **find**

- Is there a file or directory called **setup.log** anywhere under my current directory?

```
alex@debian:~$ find -name setup.log
./programs/OpenOffice.org1.1.0/setup.log
alex@debian:~$
```

- Is there anything ending in **.conf** under **/usr**?

```
alex@debian:~$ find /usr -name *.conf
/usr/share/doc/procps/examples/sysctl.conf
/usr/share/doc/adduser/examples/adduser.local.conf
/usr/share/doc/apt/examples/apt.conf
/usr/share/doc/setserial/serial.conf
/usr/share/doc/mtools/examples/mtools.conf
/usr/share/doc/cupsys/examples/printers.conf
/usr/share/doc/cupsys/examples/classes.conf
```

find is a real-time tool:
it searches the filesystem *only when* the user looks for something.

Finding Files with **locate**

- Get me all files or directories matching the string "**document**"
  - $ locate document
- **Locate** uses a pre-existing indexed database
  - **cron** invokes **updatedb** daily to update the index of files and directories found attached to the filesystem
  - Therefore, if you have just created a file, or downloaded a program, **locate** *will not find it!*

Searching the contents of text files

- To search for the string “program” in a file
  - $ grep program /usr/share/doc/whois/README
- Useful grep switches (parameters)
  - $ grep -R : search recursively the entire directory
  - $ grep -n : show the line number of any matches
  - $ grep -A 5 : show me 5 lines AFTER the match
  - $ grep -f <file> : read search patterns from <file>
  - $ grep -v : **do not show** matching lines

Handling fields in text files

- Get only the first field of your password file
  - $ cut -d : -f 1 /etc/passwd
  - This tokenizes the file **/etc/passwd** using **:** as the delimiter (-d :), and returns the first field (-f 1)
- When the *delimiter is not easy to establish*, use **awk**
  - $ awk '{print $2}' <file>
  - This returns the second token of each line of <file>

Finding unique fields, sorting, counting

- **sort** <filename>
  - Alphanumeric sort (both characters and numbers)
- **uniq** finds lines that are not repeated in a file
  - only works on sorted files
  - $ uniq sorted_file
- **wc** counts characters, words, or lines of a file
  - $ wc -c <file>
  - $ wc -w <file>
  - $ wc -l <file>

Output redirection

- Output is by default displayed on screen
  - $ ls -la
  - $ cat <filename>
- We can redirect it to a file with the > symbol
  - $ ls -la > output
  - $ cat output > output.2

Pipes

- With a pipe **(|)** we can feed the output of a tool as input to another.
  - $ ls -la | less
  - $ cat <filename>
- We can redirect it to a file with the **>** symbol
  - $ ls -la > output
  - $ cat output > output.2

Lab

- Write a command sequence that writes your current IP to the file **ip**
- Given the files **roster** and **addresses**, answer the following questions:
  - How many SCS students are currently enrolled? (F03)
  - What are their street addresses?
  - Do any of the enrolled students live outside Pittsburgh?