WINTER 14~15

OPERATING SYSTEMS LAB - ITE209
SESSION 1

ARIVUSELVAN.K

Asst .Prof (Senior)

SITE

VIT UNIVERSITY - VELLORE

Getting Started with UNIX/LINUX

OBJECTIVE:

To learn

- how to login to the Unix server from the lab
- how to know , how many and what are the commands available in UNIX that can be used by the user
- how to check the details of each command
- some UNIX commands and practice by executing them

INTRODUCTION:

UNIX Basic:

UNIX is an operating system which was first developed in the 1960s, and has been under constant development ever since. By operating system, we mean the suite of programs which make the computer work. It is a stable, multi-user, multi-tasking system for servers, desktops and laptops. UNIX systems also have a graphical user interface (GUI) similar to Microsoft Windows which provides an easy to use environment. There are many different versions of UNIX, although they share common similarities. The most popular varieties of UNIX are Sun Solaris, GNU/Linux, and MacOS X. The UNIX operating system is made up of **three** parts; **the kernel, the shell and the programs**.

The kernel

The kernel is the **core** of the UNIX operating system. Basically, the kernel is a large program that is loaded into memory when the machine is turned on, and it controls the allocation of hardware resources from that point forward. The kernel knows what hardware resources are available (**like the processor(s), the on-board memory, the disk drives, network interfaces, etc.**), and it has the necessary programs to talk to all the devices connected to it. As an illustration of the way that the **shell** and the **kernel** work together, suppose a user types **rm myfile** (which has the effect of removing the file myfile). The shell searches the file store for the file containing the program **rm**, and then requests the kernel, through system calls, to execute the

program **rm** on myfile. When the process **rm myfile** has finished running, the **shell** then returns the UNIX prompt to the user, indicating that it is waiting for further commands.

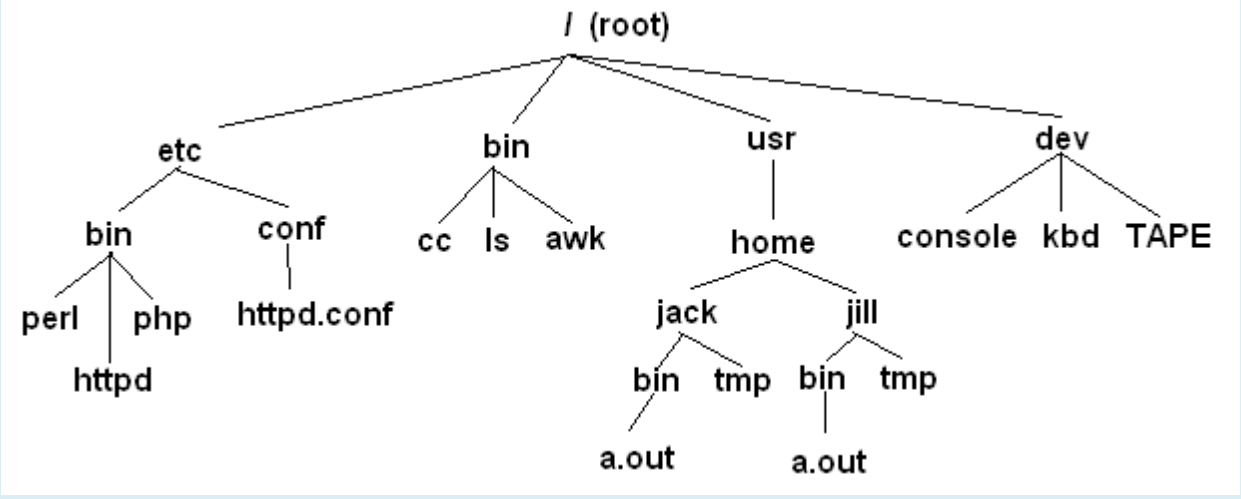
The shell

The shell acts as an interface between the **user** and the **kernel**. The shell is a **command line interpreter (CLI)**. It interprets the commands the user types in and arranges for them to be carried out. The commands are themselves programs: when they terminate, the shell gives the user another prompt .The adept user can customize his /her own shell and users can use different shells on the same machine. UNIX system offers verity of shells like **1) Bourne shell 2) c shell 3) Korn shell 4) Bash shell** (very powerful & recommended for use, Linux default shell)

Note: - The shell keeps a list of the commands you have typed in. If you need to repeat a command, use the cursor keys to scroll up and down the list or type history for a list of previous commands.

The UNIX file system

All the stored information on a UNIX computer is kept in a file system. Any time we interact with the UNIX shell, the shell considers us to be located somewhere within a file system. The place in the file system tree where we are located is called the current working directory. The UNIX file system is hierarchical (resembling a tree structure). The tree is anchored at a place called the root, designated by a slash "/". Every item in the UNIX file system tree is either a file, or a directory. A directory is like a file folder. A directory can contain files, and other directories. A directory contained within another is called the child of the other. A directory in the file system tree may have many children, but it can only have one parent. A file can hold information, but cannot contain other files, or directories.



To describe a specific location in the file system hierarchy, you must specify a "path." The path to a location can be defined as an absolute path from the root anchor point, or as a relative path, starting from the current location. When specifying a path, you simply trace a route through the file system tree, listing the sequence of directories you pass through as you go from one point to another. Each directory listed in the sequence is separated by a slash. UNIX provides the shorthand notation of **"."** to refer to the current location, and **".."** to refer to the parent directory.

Linux

Linux is not UNIX, as UNIX is a copyrighted piece of software that demands license fees when any part of its source code is used. Linux was written from scratch to avoid license fees entirely, although the operation of the Linux operating system is based entirely on UNIX. It shares UNIX's command set and look-and-feel, so if you know either UNIX or Linux, you know the other, too. Linux is a freely distributable version of UNIX developed primarily by Linus Torvalds at the University of Helsinki in Finland.

---

1. Login

Click the **start** Button in Windows desktop, and click **run**, and type **telnet 10.10.131.132**
Type your username and password. Now you have logged in to the UNIX server.

2. Commands in UNIX

How to know, how many and what are the commands available in UNIX that can be used by the user?

In the prompt, type **ls /usr/bin** this will display all the commands available in UNIX system that can be used by the user.

3. Details about UNIX commands

How to check the details of each command?

To check the detail of a command (for example **ls** command) type **man ls** where **man** is the command which displays the manual pages of the given command

4. UNIX commands

You interact with the Unix operating system by entering a command at the shell prompt.

Command syntax: The basic form of any Unix command is:

command_name options argument(s)

Most descriptions for commands such as those given in the On-line Manual use a much more precise syntax. For example:

cp [-iprR] filename ... directory

This syntax has a few simple rules. Apply them to the command description and you can understand how the command is to be used.

Command syntax rules: To understand the command syntax applies the following simple rules:

1. Any options or arguments enclosed in [] square brackets are optional.
2. Anything not enclosed in [] square brackets must be entered.
3. Boldface words are considered to be literals and must be typed exactly as they appear. This usually applies to the command name and command options.
4. Arguments shown in italics must be replaced by whatever it is that they represent. This is usually the name of a file or directory.
5. Ellipses '...' mean that the previous argument can be repeated any number of times.

Command options: Options modify the way that a command works. They usually consist of a hyphen followed by a single letter.

For example the **wc** command counts the number of words, characters and lines in a file. By using a different option you can choose what is counted.

wc -w file1	counts the words
wc -c file1	counts the characters
wc -l file1	counts the lines

Using multiple options Some commands accept multiple options. These can be grouped together after a single hyphen:

command -abc argument

Where are commands located? Unix commands are executable binary files located in directories with the name **bin** (for binary). Many of the commands that you use are located in the directory **/usr/bin**.

When you enter the name of a command the shell checks to see if it is a built-in command. If it is not, it looks for the binary file that the command name refers to in each of the directories that are defined in the **PATH** environment variable.

Locating a command: To find out if a command or utility is available on your system enter the command:

which command_name

If the command is found its pathname is displayed. The message *command_name: Command not found* is displayed when the command cannot be found.

Entering more than one command: To enter several commands on the one command line, use a **;** (semicolon) to separate each one from the next. For example:

cd .. ; ls -l

This command line contains two commands. The first, **cd ..** changes the current directory to the parent directory. The second, **ls -l** produces a long listing of the contents of the current directory. If necessary you can continue the commands onto another line.

Using the command history: Each time you enter a command it is added to a command history list. You can reuse commands from this list. This command history facility is not available with the Bourne shell (sh).

To display the command history list enter the command:

history

This displays a numbered list of commands in the order in which you have used them.

Re-running a command from the history list To run a command from the history list, use a command from this table.

Run the previous command	**!!**
Run command number n	**!n**

Changing the size of the command history: By default only the previous command is saved in the command history list. To save a larger number of commands set the history variable in your shell start-up file. For example:

set history=50

This will keep a history of the fifty (50) most recent commands that you have used.

Connecting commands together: Unix allows you to link two or more commands together using a pipe. The pipe takes the standard output from one command and uses it as the standard input to another command.

command1 | command2 | command3

The | (vertical bar) character is used to represent the pipeline connecting the commands. With practice you can use pipes to create complex commands by combining several simpler commands together.

To pipe the output from one command into another command:

who | wc -l

This command tells you how many users are currently logged in on the system: a lot!
The standard output from the **who** command - a list of all the users currently logged in on the system - is piped into the **wc** command as its standard input. Used with the -l option this command counts the numbers of lines in the standard input and displays the result on the standard output.

Text editors in Linux

Linux is just as well suited for word processing as any other operating system. There are several excellent word processing programs for Linux like AbiWord, KWord, part of the KOffice suite and the OpenOffice.org as well as StarOfficesuite's word processor.

Why use a text editor?

A text editor is just like a word processor without a lot of features. All operating systems come with a basic text editor. Linux comes with several. The main use of a text editor is for writing something in plain text with no formatting so that another program can read it. Based on the information it gets from that file, the program will run one way or another.

The text editor "vi"

The most popular text editor for Linux is called 'vi'. This is a program that comes from UNIX. There is a more recent version called 'vim' which means 'vi improved'. The problem with 'vi' or 'vim' is that a lot of people don't like it. You have to remember a lot of key combinations to do stuff that other text editors will do for you more easily.

Working with 'vi'

Let's make a text file. Type:

vi OS

We'll see a line of tildes down the left side and the name **'OS'** at the bottom and [new file].

To write something, we have to press **ESC** and the **'i'** key (i for insert). We can always erase our mistakes with the **backspace** key.

To save this file, we would press **ESC** then the colon key **':'** then **'w'** (write)

To save the file and quit **vi**, you would press ESC, ESC the colon key **':'** then **wq** (write, quit)

To quit without saving, press **ESC, ':' then 'q'**. vi may protest if we've written something and we don't want to save it. If we press **ESC ':' 'q!'** with an exclamation point, vi will accept it and not save our changes.

Practice the following UNIX commands:

Try the following commands and observe the output. Try the option in each command.

date Displays the date and time

cal Displays the current month calendar

echo Advanced OS Displays the text typed next to it (**Advanced OS)**

who Displays information about all users who are currently logged into the system.

whoami Displays information about your login.

pwd Displays the current working directory

id Displays userid and groupid.

finger Displays information about a specified user id if user id is specified otherwise it gives Information about all currently logged users.

cd Change directory

cd .. To go to the **parent directory**

cd - To go to the **previous directory**

ls List the contents of a directory with **ls**

cp Copying files from one place to another

tty Shows special file that represents your terminal.

du Displays disk storage usage statistics.

clear Clears the contents of the screen

whatis Display a one-line summary about a keyword.

which Locate a command; display its pathname

cat /etc/profile Displays the content of the given file **profile** which is in the directory **/etc**
