Intro to Linux

1st July 2015

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

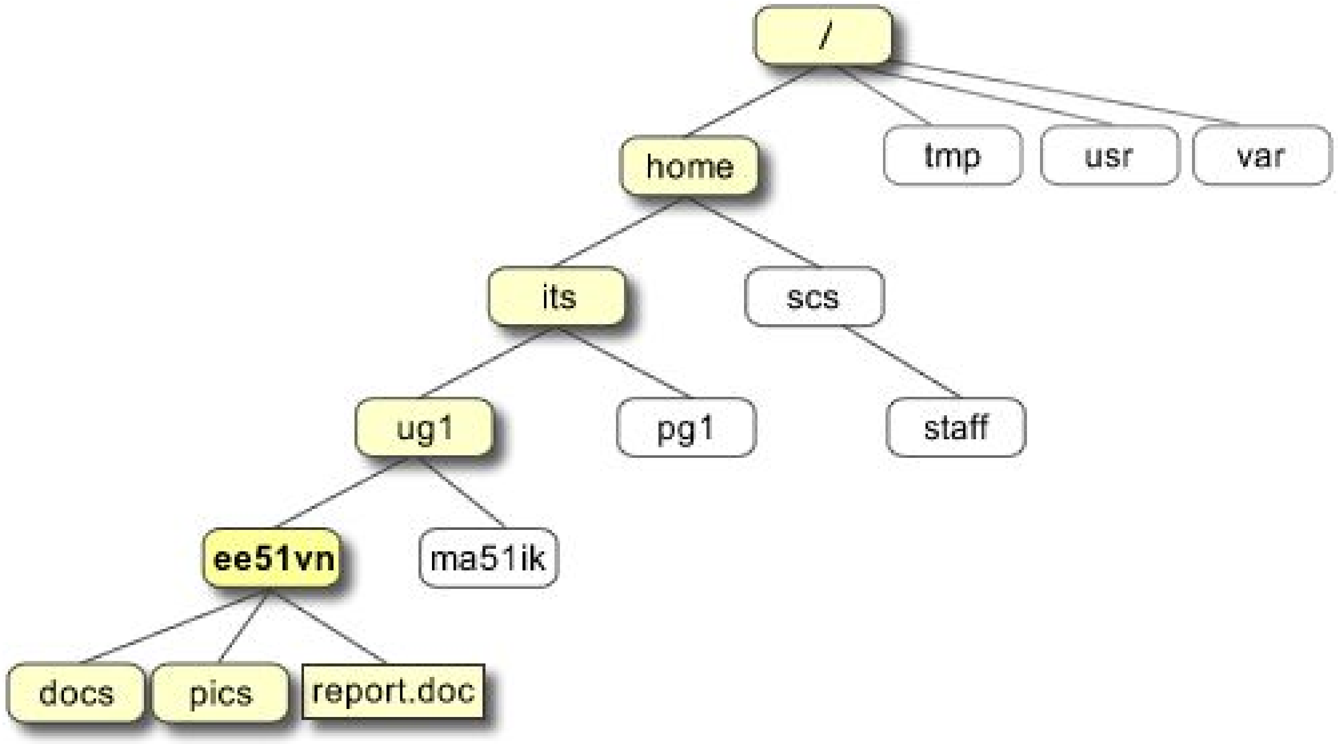
UNIX

- an OS first developed in the 1960s
- has been under constant development ever since
- most popular varieties are: Sun Solaris, GNU/Linux, and MacOS X
- a UNIX OS is made up of three parts
    - **Kernel;** The kernel of UNIX is the hub of the OS: it allocates time and memory to programs and handles the filestore and communications in response to system calls,
    - **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,
    - and **Programs;** The commands are themselves also programs: when they terminate, the shell gives the user another prompt.

Files and processes

1. Everything in UNIX is either a file or a process.
2. A process is an executing program identified by a unique PID (process identifier).

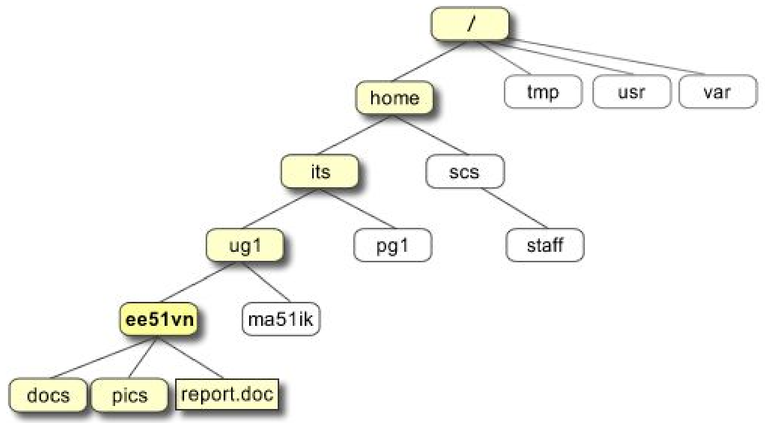
The Directory Structure



Paths

- We are in the 'its' directory right now
- We want to access 'report.doc'
- Relative
  - **./ug1/ee51vn/report.doc**

- Absolute
  - **/home/its/ug1/ee51vn/report.doc**



Terminal Commands

ls	//list files and directories
ls -a	//list all files and directories
mkdir	//make a directory
cd “directory”	//change to named directory
cd	//change to home-directory
cd ~	//change to home-directory
cd ..	//change to parent directory
pwd	//display the path of the current directory

More Terminal Commands

cp “file1” “file2”	//copy file1 and call it file2
mv “file1” “file2”	//move or rename file1 to file2
rm “file”	//remove a file
rmdir “directory”	//remove a directory
cat “file”	//display a file
less “file”	//display a file a page at a time
head “file”	//display the first few lines of a file
tail “file”	//display the last few lines of a file
grep 'keyword' “file”	//search a file for keywords
wc “file”	//count number of lines/words/characters in file
banner “anything”	//prints a banner for “aything”

Redirection and Pipes

“command” > “file”	//redirect standard output to a file
“command” >> “file”	//append standard output to a file
“command” < “file”	//redirect standard input from a file
“command1” | “command2”	//pipe the output of command1 to the input of command2
cat “file1” “file2” > “file0”	//concatenate file1 and file2 to file0
sort	//sort data
who	//list users currently logged in

Wildcards and help

*	//match any number of characters
?	//match one character
man “command”	//read the online manual page for a command
whatis “command”	//brief description of a command
apropos “keyword”	//match commands with keyword in their man pages

File Access Rights

- -rwxrwxrwx a file that everyone can read, write and execute (and delete).
- -rw------- a file that only the owner can read and write - no-one else can read or write and no-one has execution rights (e.g. your mailbox file).

- chmod

Special input

1.	ls –lag	//list access rights for all files
2.	chmod [options] “file”	//change access rights for named file
3.	command &	//run command in background
4.	ˆC	//kill the job running in the foreground
5.	ˆZ	//suspend the job running in the foreground
6.	bg	//background the suspended job
7.	jobs	//list current jobs
8.	fg %1	//foreground job number 1
9.	kill %1	//kill job number 1
10.	ps	//list current processes
11.	kill 26152	//kill process number 26152
12.	history	

Compiling Source Code

- The **make** command
  - allows programmers to manage large programs or groups of programs
  - aids in developing large programs by keeping track of which portions of the entire program have been changed, compiling only those parts of the program which have changed since the last compile
  - gets its set of compile rules from a text file called **Makefile** which resides in the same directory as the source files

- The **configure** Shell Script
  - attempts to guess correct values for various system-dependent variables used during compilation
  - uses those values to create a **Makefile** in each directory of the package

Compiling Source Code

1. **cd** to the directory containing the package's source code.
2. Type **./configure** to configure the package for your system.
3. Type **make** to compile the package.
4. Optionally, type **make check** to run any self-tests that come with the package.
5. Type **make install** to install the programs and any data files and documentation.
6. Optionally, type **make clean** to remove the program binaries and object files from the source code directory

File zipping

- Gunzip
- Tar
- Bzip2

Example:

```
> gunzip sample.tar.gz

> tar -xvf sample.tar

> bzip2 -d sample.bz2
```

Environment Variables and Paths

1. http://www.ee.surrey.ac.uk/Teaching/Unix/unix8.html

Books

1. http://www.ee.surrey.ac.uk/Teaching/Unix/books-uk.html