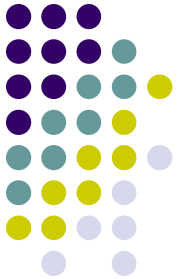
Linux / Unix

Santosh Kyadari (santoshk@tifr.res.in)

--CCCF

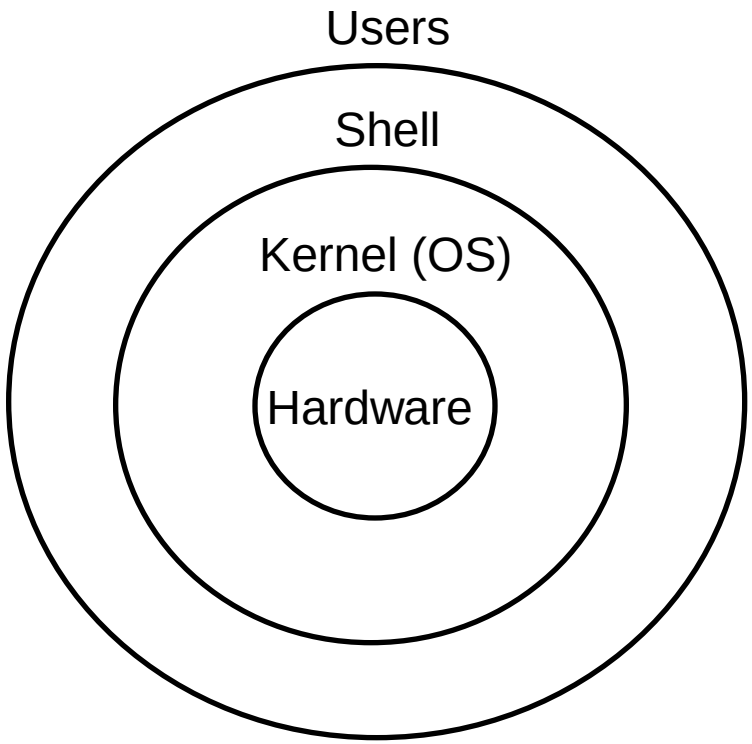
Date: 5 -9 -2012

Introduction

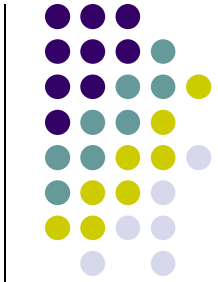


- Linus Torvalds – Creator of Linux
- Open Source Operating System
- Source Code Available
- Kernel can be customized to user’s needs

Linux/Unix system organization

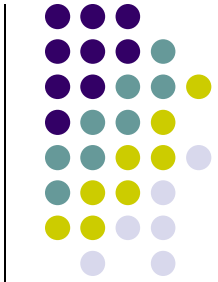


File structure



- /bin , /usr/bin , /usr/local/bin → user executables
- /etc → configuration files
- /root , /home/users → Home directories
- /var , /srv, /usr → server data
- /lib, /lib64, /usr/lib , /usr/local/lib →shared libraries
- /boot → Kernel , boot loaders
- /tmp →Temporary files
- /proc , /sys → system information
- /media , /mnt →mount points
- More info: http://www.comptechdoc.org/os/linux/commands/linux_crfilest.html

File system commands



- pwd - report your current directory
- cd <*to where*> - change your current directory
- ls <*directory*> -list contents of directory
- cp <*old file*> <*new file*> - copy
- mv <*old file*> <*new file*> - move (or rename)
- rm <*file*> -delete file(s)
- mkdir <*new directory name*> -make a directory
  - mkdir -p /work/junk/{one,two,three,four}
- rmdir <*directory*> -remove an empty directory
- man <command name>
  - man –k mail

$ man *command* gives you help on that command.

ls command

ls - list directory contents

Usage : ls [OPTIONS] [FILE]

OPTIONS

- -l Use a long listing format
- -a Do not ignore entries starting with . (for e.g .forward)
- -h Print sizes in human readable format (e.g., 1K 234M 2G)
- -d List directory entries instead of contents
- -R List subdirectories recursively
- -r Reverse order while sorting
- -S Sort by file size
- -t Sort by modification time
- -1 List one file per line

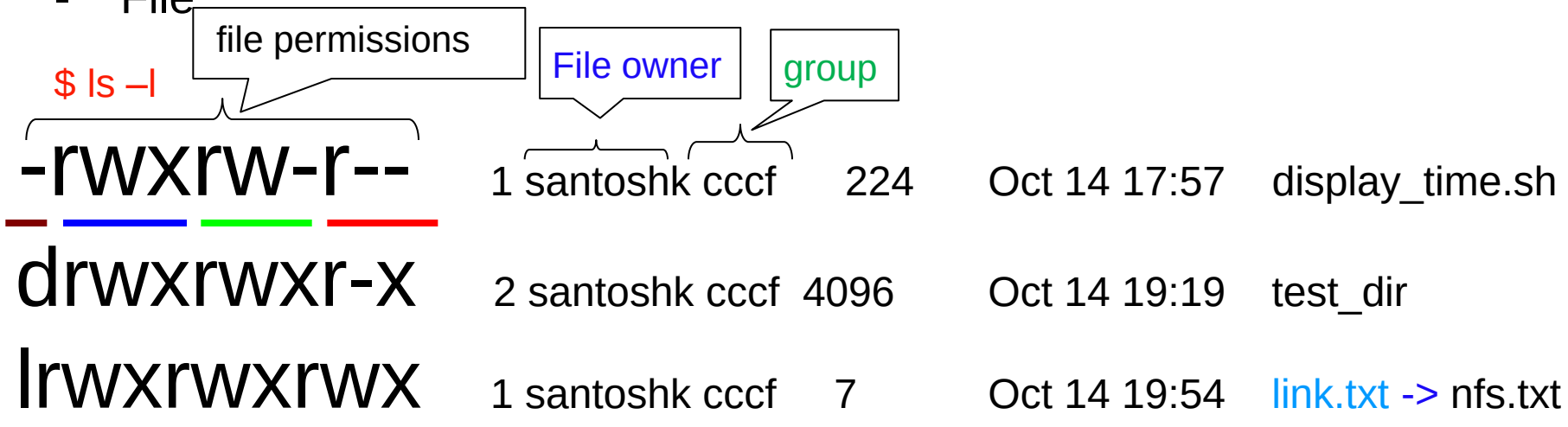
Mostly used options in ls

ls -l, ls –la, ls -1, ls -lh, ls -ltr, ls -lS

File permissions.

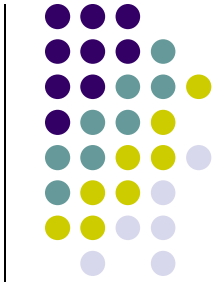
- There are 3 kinds of users in linux :
- you (user) u , your friends (group) g and everyone else (others) o.

r - Read permissions
w - Write permissions
x - execute permissions
d - Directory
- File



- For a file if x is set that user can execute the file
- For a directory if x is set that user can enter in that directory.

Changing File Permissions



- Make a file readable to your friends:

  $ chmod 765 <filename>

  7 -> 111 -> rwx

  6 -> 110 -> rw-

  5 -> 101 -> r-x

  -rwx rw- r-x 1 santoshk cccf 224 Oct 14 17:57 abcd.txt

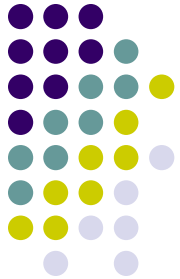
  OR

$ chmod +w abcd.xt

$ chmod o+w abcd.xt

$ chmod g+x,o+w abcd.txt

Changing File Ownership



- Change who owns a file:

  $ chown *<user> <filename>*

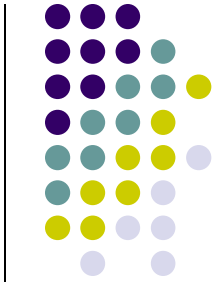
  chown ksri:cccf abcd.txt
  chown –R ksri:cccf scritps_dir

- Change to which group the file belongs:

  $ chgrp *<group> <filename>*

  chgrp cccf abcd.txt

Getting Recursive



- copy a directory and its contents to other hosts ID:

$ scp -r <*directory*> *santoshk@tifr.res.in:*

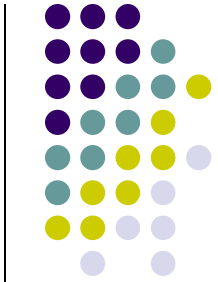
- copy a directory and its contents:

$ cp -r <*directory*> <*destibnation_dir*>

- Find a pattern in a directory and its subdirectories:

$ grep -r <*pattern*> <*destibnation_dir*>

Redirecting output to a file with >
Redirecting input from a file with <

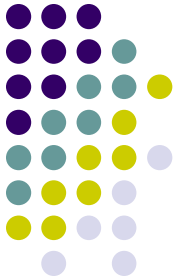


Redirection Symbols

>file	Make file the standard output
<file	Make file the standard input
>>file	Make file the standard output, appending to it if already exists
n>file	Make file the output for file descriptor n

File Descriptor	Name	Common Abbreviation	Usual Default
0	Standard input	stdin	Keyboard
1	Standard output	stdout	Terminal
2	Standard error	stderr	Terminal

Redirecting examples



ls –l > abcd.txt Redirects output to abcd.txt

sort < account.txt Accepts the input from account.txt

mail -s “Test subject" santoshk@gmail.com <body.txt

ls –l santosh.txt 2> error.txt Redirects error to error.txt

ls –l santosh.txt 2>&1 error.txt Redirect output and error to error.txt

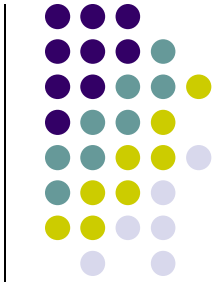
ls –l 2>&1 | tee –a log.txt

ls –l &> file

ls -l &>> test

ls –l >>log.txt 2>&1

Piping |



- Pipes take the output of the first program and feed that output into the input of the next program.
- The output of a command can be piped to another command for further processing
- Also sometimes known as “filters”.

Examples:

ls –l | wc –l

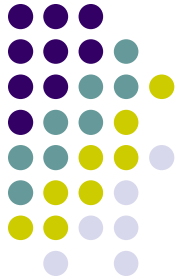
cat nfs.txt | more

last | grep “^root” | less

last | grep “^root” | cut -d -f 2 | less

grep “error” something.out | tail -1

(un)aliasing



- create shortcuts for yourself

  $ alias ll='ls –la'

- Use alias with no arguments to discover current aliases

  $ alias

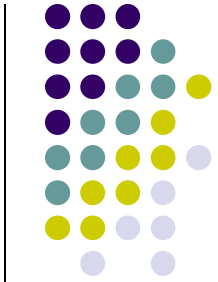
alias rm='rm –I'

alias ll='ls -l --color=tty'

alias q='lpq -Plp1; lpq -Plp2; lpq -Plp3; lpq -Plp8; lpq -Pnew; lpq -Pold'

Type "unalias rm" to remove alias.

Login using ssh



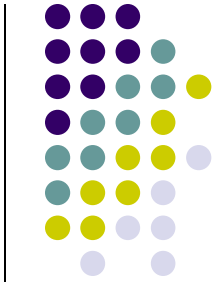
- ssh – remote login program

  $ ssh –l santoshk cc1.tifr.res.in

  $ ssh santoshk@cc1.tifr.res.in

ssh client in windows is putty. Download from
http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe

Copy to remote machine : scp



- copy local to remote

```
$ scp <source file>  user@machine:<path>
```

- copy remote to local

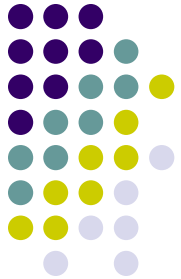
```
$ scp user@machine:<path> <source file>
```

`-p` Preserves mode, time stamps

-r Recursively copy entire directories.

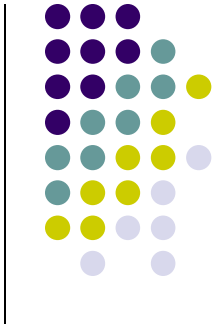
-v Verbose mode.

More commands



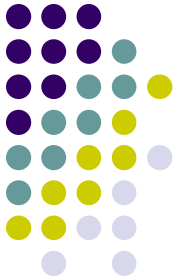
- grep - grep is global / regular expression / print
  - grep –i santosh /etc/passwd
- find - search for files in a directory hierarchyreport uniq lines
  - find ./ -name “*.txt” # Find *.txt files present directory
- date - date command prints or sets the system date and time
  - date #Wed Oct 13 17:23:56 IST 2010
  - date '+%d/%b/%Y %H:%M:%S' # displays 13/Oct/2010
- touch – creates the file if it doesn’t exists or changes date stamp to current if exists
  - touch abcd.txt #creates empty abcd.txt
- ln - Reference to another file or directory
  - ln –s nfs.txt link.txt # creates symbolic link of nfs.txt

More commands



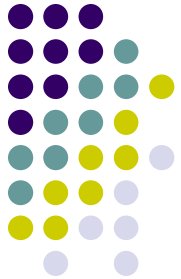
- sort <filename> - sort lines of text files
  - sort -nr +0 -1 <filename> # sorts according to first field
- uniq <filename> - report uniq lines
  - uniq –c <filename> # display the uniq entries with count
- tee - read from standard input and write to standard output and files
  - find / “abc*.*” 2>&1 | tee –a log.txt
  - #finds files and displays output and erro and tees to log.txt
- tar – backup / archiving utility
  - tar –cvf abcd.tar /usr #create a tar file of /usr directory
- head - output the first part of files
  - head -10 abcd.txt #displays top 10 lines of abcd.txt

More commands



- tail - output the last part of files
  - tail -5 abcd.txt # displays last 5 lines of abcd.txt
  - tail –f maillog.log # displays continuously the new appending data.
- cat - concatenate files and print on the standard output
  - cat a.txt b.txt >>z.txt #appends a.txt and b.txt to z.txt
- more – view the contents of a text file one screen at a time
- echo - display a line of text\
- tr - translate or delete characters
  - echo “Hello world” | tr '[a-z]' '[A-Z]' # will display HELLO WORLD
- expr - Evaluate an expression
  - expr 5 * 2 # multiplies 5 and 2

Advance Commands



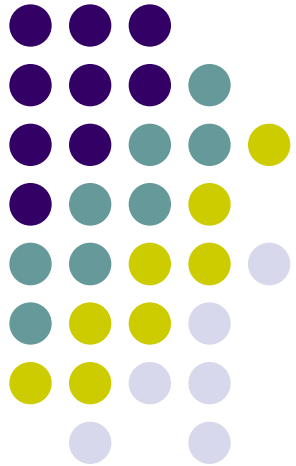
- Some of system related commands

exec, time, top, ps, logger, su, rpm, yum, dd, find, stat, lsof, xargs,chattr

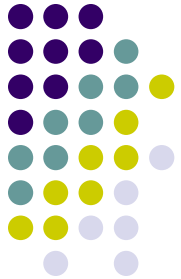
- Some of Network related commands

ping, netstat, ifconfig, ifup, ifdown, dig, nslookup, host, rsync, ftp, ssh, telnet, wget, lynx, ntpdate, whois, tcptrack

vi editor

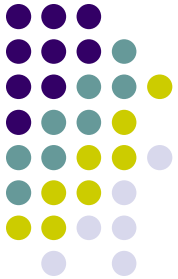


Introduction



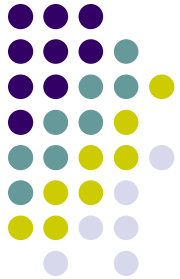
- vi is text editor
- Original vi program was written by Bill Joy in 1976
- Use vi editor to:
  - create text files
  - edit text files
- The vi editor is not a text formatter like MS Word
- The current iteration of **vi** for Linux is called **vim**
  Vi Improved

Starting vi



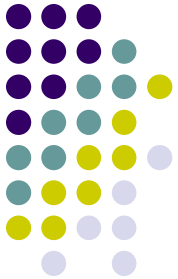
- Type **vi** <filename> at the shell prompt
- After pressing enter the command prompt disappears and you see tilde(~) characters on all the lines
- These tilde characters indicate that the line is blank

Vi modes



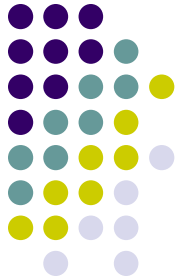
- There are two modes in vi
  - Command mode
  - Input mode
- When you start vi by default it is in command mode
- You enter the input mode through various commands
- You exit the input mode by pressing the Esc key to get back to the command mode

How to exit from vi



- First go to command mode
  - press **Esc** There is no harm in pressing **Esc** even if you are in command mode. Your terminal will just beep and/or or flash if you press **Esc** in command mode
- There are different ways to exit when you are in the command mode

How to exit from vi
(comand mode)

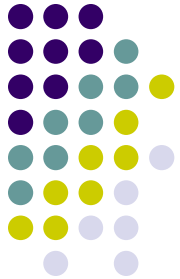


- **:q** <enter> is to exit, if you have not made any changes to the file
- **:q!** <enter> is the forced quit, it will discard the changes and quit
- **:wq** <enter> is for save and Exit
- **:x** <enter> is same as above command
- The **!** Character forces over writes, etc.
  :wq!

Moving Around

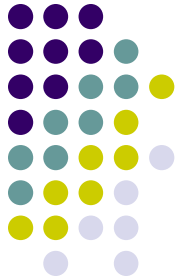
- You can move around only when you are in the command mode
- Arrow keys usually works(but may not)
- The standard keys for moving cursor are:
  - **h** - for left
  - **l** - for right
  - **j** - for down
  - **k** - for up

Moving Around



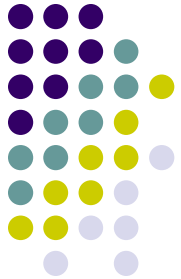
- **w** - to move one word forward
- **b** - to move one word backward
- **$** - takes you to the end of line
- <**enter**> takes the cursor to the beginning of next line

Moving Around



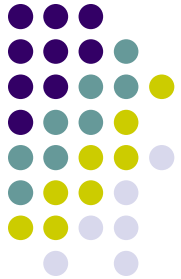
- **-** - (minus) moves the cursor to the first character in the current line
- **H** - takes the cursor to the beginning of the current screen(Home position)
- **L** - moves to the Lower last line
- **M** - moves to the middle line on the current screen

Moving Around



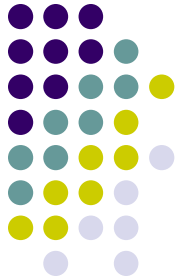
- **f** - (find) is used to move cursor to a particular character on the current line
  - For example, **fa** moves the cursor from the current position to next occurrence of '**a**'
- **F** - finds in the reverse direction

Moving Around



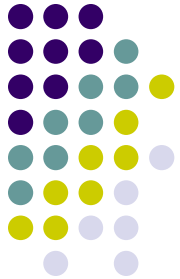
- **)** - moves cursor to the next sentence
- **}** - move the cursor to the beginning of next paragraph
- **(** - moves the cursor backward to the beginning of the current sentence
- **{** - moves the cursor backward to the beginning of the current paragraph

Moving Around



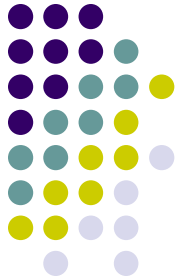
- **Control-d** scrolls the screen down (half screen)
- **Control-u** scrolls the screen up (half screen)
- **Control-f** scrolls the screen forward (full screen)
- **Control-b** scrolls the screen backward (full screen).
- xG- to go at x line
- G- takes you to bottom line of file
- gg- takes you to first line

Entering text



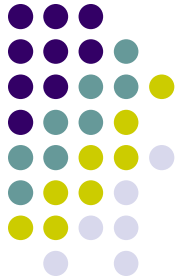
- To enter the text in vi you should first switch to input mode
  - To switch to input mode there are several different commands
  - **a** - Append mode places the insertion point after the current character
  - **i** - Insert mode places the insertion point before the current character

Entering text



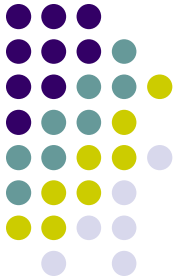
- **I** - places the insertion point at the beginning of current line
- **o** - is for open mode and places the insertion point after the current line
- **O** - places the insertion point before the current line
- **R** - starts the replace (overwrite) mode

Editing text



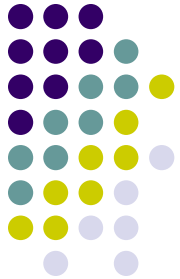
- **x** - deletes the current character
- **d** - is the delete command but pressing only d will not delete anything you need to press a second key
  - **dw** - deletes to end of word
  - **dd** - deletes the current line
  - **d0** - deletes to beginning of line

The change command



- **c** - this command deletes the text specified and changes the vi to input mode. Once finished typing you should press <**Esc**> to go back to command mode
- **cw** - Change to end of word
- **cc** - Change the current line
- There are many more options

Structure of vi command

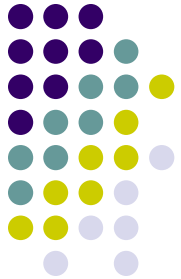


- The vi commands can be used followed by a number such as

  n<command key(s)>

  - For example **dd** deletes a line **5dd** will delete five lines.
- This applies to almost all vi commands
- This how you can accidentally insert a number of characters into your document

Undo and repeat command



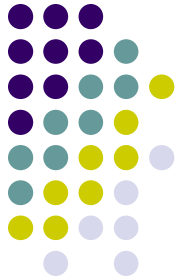
- **u** - undo the changes made by editing commands
- **.** (dot or period) repeats the last edit command

Copy, cut and paste

- **yy** - (yank) copy current line to buffer
- **nyy** - Where **n** is number of lines
- **p** - Paste the yanked lines from buffer to the line below
- **P** - Paste the yanked lines from buffer to the line above

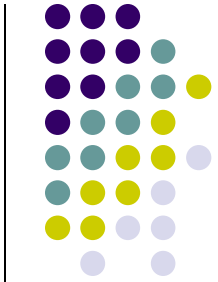
(the paste commands will also work after the **dd** or **ndd** command)

vi Tricks



- Indent four lines: 4>>
- Will delete the character under the cursor, and put it afterwards. In other words, it swaps the location of two characters: xp
- Similar to xp, but swapping lines: ddp

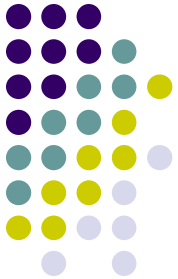
Creating a shell script using vi



- Create a directory **class**
- Change into **class**
- **vi myscript.sh**
- inside the file enter following commands

```
clear
echo "============"
echo "Hello World"
echo "============"
sleep 3
clear
echo Host is $HOSTNAME
echo User is $USER
```

Creating a shell script using vi

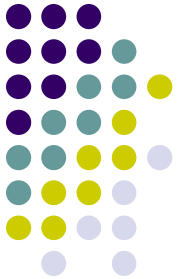


- Save the file
- Change the permissions on myscript.sh

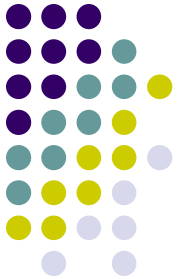
  chmod 700 myscript.sh <enter>
- Now execute myscript.sh

  myscript.sh <enter>
- Did the script run?
- Why not?
  - Hint, think about absolute vs relative path
  - Type **echo $PATH** to see your PATH variable
  - Try this **./myscript.sh** <enter>
  - The **./** mean right here in this directory!

References



- Unix shell programming -by Yashwant Kanetkar
- Unix Concepts and Applications –by Sumitabha Das
- http://www.grymoire.com/Unix/Sed.html
- http://www.grymoire.com/Unix/Awk.html
- http://www.grymoire.com/Unix/Quote.html
- http://www.grymoire.com/Unix/Find.html



Thanks