Tech Mahindra

Unix Fundamentals & Commands

Day 3

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

Objectives

- At the end of this session, you will be able to:
  - Use general purpose Unix commands

Agenda: Day 3

- The vi Editor
- Regular Expressions
- grep
- egrep
- fgrep
- Advanced Commands
- FTP Overview
- Unix Process Control

vi Editor

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

The vi Editor

- The vi editor is a screen-based editor which lets a user create new files or edit existing files
- A key concept in vi is combining a certain action with a movement
- vi is extremely powerful in moving around within (or between) files

The vi Editor (Contd..)

- A vi session begins by invoking the command “vi” with a filename

  `$ vi [filename]`

- You can start vi without a filename, but when you want to save your work, you will have to tell vi which filename to save it into

- The last line in the screen is reserved for some commands that you can enter to act on the text

- This line is also used by the system to display messages

Modes of Operation

The three different modes of operations are:

- Command mode:
  - This is the default mode where you can pass the commands to act on the text, using most of the keys of the keyboard
  - You can switch to this mode using “Esc” key
- Insert (Input) mode:
  - To enter the text, you have to enter into input mode
  - Press the key “i” to enter into insert mode from command mode
  - You can switch to command mode by pressing “Esc” key
- ex mode or line mode:
  - You have to save your file or switch to another file or make a global substitution in the file
  - You then have to use ex mode, where you can enter the instruction in the last line of the screen
  - To enter into this mode, press “Esc” key followed by “:”

Text Insertion Commands

Command	Description
i	Inserts text before cursor position
a	Appends text after cursor position
I	Inserts text at beginning of line
A	Appends text after end of line
o	Opens line below current line to insert text
O	Opens line above current line to insert text

Cursor Movement Commands

Command	Description
h	left by one character
l	right by one character
k	up by one line
j	down by one line
w	right by one word
b	left by one word
0 or ^	beginning of line
$	end of line

Text Search Commands

Command	Description
/text	searches and highlights the text downwards
n	moves between highlighted text
*	searches the identical text on which the cursor was
?text	searches the text upwards

Text Deletion Commands

Command	Description
x	character under cursor
X	character before cursor
[n]dw	delete n words
d0	beginning to cursor position
d$ or D	cursor position to end of line
[n]dd	n lines from current line
[n]dd	pp will paste deleted lines to current cursor position equivalent to Ctrl-X and Ctrl-V [in Windows]

Text Copy Commands

Command	Description
y	character
y0	beginning to cursor position
y$	cursor position to end of line
[n]yw	copy n words
[n]yy	n lines from current line in to the buffer
[n]yy p	p will paste copied lines to current cursor position equivalent to Ctrl-C and Ctrl-V [in windows]

Text Manipulation Commands

Command	Description
nc [space]	overwrites next n characters with space
c0	overwrites the portion between beginning of line to cursor position
c$	overwrites the portion between cursor to end of line
cw	overwrites current word
:%s/pattern1/pattern2/g	globally replaces pattern1 with pattern2 on the specified lines
:%s/pattern1/pattern2/gc	globally replaces pattern1 with pattern2 on the specified lines interactively

Text Related Commands

Command	Description
ab <abb> <longword>	set abbreviation for a long word
una string	unset abbreviation
>>	right shifting a line
<<	left shifting a line
R	replace characters starting with current character till ‘Esc’ is pressed
R	replace current cursor character

File Related Commands

Command	Description
ZZ or :wq	save and exit
:w	save & continue editing
:q!	quit without saving
:r filname	insert contents of file filname
:[addr1,addr2]w filname	write the lines between line number addr1 and line number addr2 in the file filename

File Related Commands (Contd..)

Command	Description
1,$s/source/target/	substitute string source by string target from line number 1 to last line
u	undo last change on the line
U	undo last changes on the line
Ctrl-R	redo the undone changes
e	edit multiple files
e#	return to previous file000

Visual Mode Commands

Command	Description
sp	splitting window
Ctrl-w	toggle between windows
<Ctrl-w>j	moves to lower window
<Ctrl-w>k	moves to upper window

Customizing vi

- Set commands:

Command	Description
set all	displays all set option
set autoindent/ai	does automatic indentation
set number/nu	shows all line duly numbered
set showmatch	helps to locate matching brackets
set tabstop=5	sets tab=5 for display
set ic	ignore case while pattern matching

- When the string “no” is prefixed to any option, it indicates that the option is inoperative

Regular Expressions and Grep

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

Regular Expression

- Often called a pattern, is an expression that describes a set of strings

- Example,

  a regular expression “amit” may match “amit”, “amita”,”amitabh”, “namit” etc…

- Many UNIX tools, primarily grep,sed & awk make use of regular expressions in text processing

Regular Expressions (contd.)

- Regular Expressions can be divided into:
  - Basic regular expressions (BRE)
  - Supported by grep
  - Extended regular expressions (ERE)
  - Supported by grep –E or egrep

Basic Regular Expression

Special Operators

\	Quote the next metacharacter
^	Match the beginning of the line
.	Match any character
$	Match the end of the line
[]	Character class

Extended Regular Expression

Special Operators

|	Alternation
( )	Grouping

Examples

- “a.g” matches aag, abg, a1g, etc
- “a[pmt]g” matches apg, amg or atg
- “a[^pmt]g” matches aag, abg but not apg or amg or atg
- “^ftp” matches ftp at the beginning of a line
- “tle$” matches tle at the end of a line
- “^$” matches a line with nothing in it
- “jelly|cream” matches either jelly or cream
- “(eg|pe)gs” matches either eggs or pegs

Quantifiers

BRE

*	Match 0 or more times

ERE

+	Match 1 or more times
?	Match 1 or 0 times

Examples

"adg*" ad followed by zero or more g characters

".*" Any character, any number of times

"[qjk]" Either q or j or k

"[^qjk]" Neither q nor j nor k

"[a-z]" Anything from a to z inclusive

"[^a-z]" No lower case letters

"[a-zA-Z]" Any letter

"[a-z]+" Any non-zero sequence of lower case letters

"(da)+" Either da or dada or dadada or...

grep family

- fgrep: fast searching for fixed strings

- $fgrep string file(s)
  - It handles fixed character strings as text patterns
  - Does not use regular expressions
  - Faster than grep and egrep for searching text strings

- Examples

  fgrep “Ramesh” datalist
  - If found, lists the line(s) containing Ramesh

grep family

- grep: is called as a global regular expression printer
- It searches for a given pattern in a file(s)

- $ grep -[cvnl] [pattern] [files]

Option	Description
-c	counts the total no of lines containing the pattern
-v	displays all the lines not containing the pattern
-n	displays lines with line number
-l	displays file names containing the pattern

- Example:

  grep “Agg*[ra][ra]wal” datalist
  - It lists all lines where the pattern matches some text
  - The possible matches for the text are:
    Agrawal, Agarwal, Aggarwal, Aggrawal
    (and many more combinations possible)

grep family

- egrep: extended grep, supports both BRE as well as ERE

- grep –E can also be used in the place of egrep
- Examples:
  - $ egrep ‘ (John|Johnathon) Smith ‘ employee.txt

  - This will search for John Smith as well as for Johnathon Smith

  - grep –E “(S|Sh)arma datalist
  - Matches Sarma or Sharma in the text from datalist

Advanced Commands

TLS

File Compression – gzip /gunzip

- Compressing a File:

  $ gzip testfile

  The command compresses the **testfile** to **testfile.gz**

  $ gzip –c testfile > testfile.gz

  The command makes a copy of **testfile** and compresses it to **testfile.gz**

- Decompressing a File:

  $ gunzip testfile.gz

  The command decompresses the **testfile.gz** to **testfile**

tar Utility

- The tar command is used for creating an archive of a directory hierarchy.
- tar archives are a handy way of sending a bunch of files (or a program distribution) across the network or posting them on the internet.
  - Begin by creating a tar archive of the files.
  - Transmit that tar archive over the network or post it online.
  - Untar the files where you want them.
- Syntax:

```
tar [–cvfx] <archive_name>.tar <files>
```

-c	Create a new archive
-v	verbosely list file processed
-f	use archive file
-t	list the contents of an archive
-x	extract files from an archive

tar utility

Usage:

- Create a tar archive of your home directory and place it in your working directory:
  - tar –cvf myhome.tar home/
- View the contents of the tar archive:
  - tar –tvf myhome.tar
- Extract the tar archive to your current working directory:
  - tar –xvf myhome.tar

Tech Mahindra

FTP Overview

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

FTP Overview

- File Transfer Protocol (FTP) is a common method of transferring files between computer systems
- The netstat command can be used by all the users to check the services that are running
- The example below shows the expected output, there would be no output at all if FTP is not running

```
$ netstat -a | grep ftp
```

Connecting to FTP

- Connection to FTP Server can be done from Windows command prompt by two ways as mentioned below:
  - ftp IPAddress

    ```
    e.g. C:\> ftp 10.11.5.208
    Connected to 10.11.5.208.
    220 (vsFTPd 2.0.1)
    User (10.11.5.208:(none)): user1
    331 Please specify the password.
    Password:
    230 Login successful.
    ftp>
    ```

  - ftp

    ```
    e.g. c:\> ftp
    ftp> open 10.11.5.208
    ```

Transferring a File: FTP

- To transfer a file from Windows to Unix use the ftp put/send command. (The file need to be present in the current dir on Windows)

```
ftp> put testfile
                 or
ftp> send testfile
```

- To transfer a file from Unix to Windows use the ftp get command. (The file need to be present in the current dir on Unix)

```
ftp> get testfile
                 or
ftp> recv testfile
```

Transferring Multiple Files: FTP

- To transfer multiple files from Windows to Unix use the ftp mput command. (The files need to be present in the current dir on windows)
  - ftp> mput testfile1 testfile2 testfile3
- To transfer multiple files from Unix to Windows use the ftp mget command. (The files need to be present in the current dir on Unix)
  - ftp> mget testfile1 testfile2 testfile2
- Note:
- All the files will be sent in text mode by default; To send binary files give the command:

  ftp>binary

Unix Process Control

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

ps

- Each command running on Unix system is termed as a process
  - ps command shows process status and displays the attribute of a process
  - Usage: $ ps
- Options

$ ps -f	--> full option
$ ps -f -u ELITE	--> gives processes of user ELITE
$ ps -a	--> all users processes.
$ ps -e	--> all processes on the system including system processes

Process Priority

- Each process has a priority
- It decides:
  - Sense of urgency
  - Which process should get execution time first
- Priority is denoted by a number from –20 to 19
  - -20 is the highest priority and 19 is the lowest

& and jobs

&

- To execute any process in background use & at the end of the command
  - Usage: <command name> &
  - Example: $ sh test &

jobs

- To check out jobs currently running in background use the command jobs
  - Usage: $ jobs

fg

- If we want to switch the background job to foreground we use fg command
- fg Brings any background job to foreground
  - Usage: fg [job number]
  - Example: fg 2
- The number specified here is the control number listed by jobs command, NOT the Process ID
- fg without any parameters takes the most recent task

kill

- When we need to forcefully finish some process we use this command
- Kill is used to terminate a process. The command uses one or more PIDs as its arguments.
  - Usage: kill <process id>
  - Example: Kill -9 105
  - It will terminate job with PID 105
- The option –9 indicates sure kill signal.
- $ kill $!
- The system variable $! Stores the PID of the last background job

bg

- Once a job has been suspended or stopped, it will not do any work
- If that job is switched to the background, it can continue on its way
  - Usage: bg [job number]
  - Example: bg 1
  - bg without any arguments moves the most recent task into the background

nohup

- Sometimes we need a job to be running even if we logout

- With the command, nohup you can continue to run programs even after you log out

  - Usage: nohup <command name>
  - Example: nohup sh a.sh

Summary

In this session, we have covered:

- The vi editor
- Regular Expressions
- grep family commands
- Advanced Commands
- FTP Overview
- Unix Process Control

TLS

Unix Fundamentals & Commands

Thank You

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

Tech Mahindra