Using the Bash Shell

Unit objectives

After completing this unit, you should be able to:

- What is Shell?
- Gaining Access to Shell
- Entering Commands
- Using Aliases
- Getting Information about Command
- Working with Shell History
- Bash command-line Editing
- Autocompletion
- Filename Shorthand
- Redirection

What is Shell?

- Interpreter
- Provide command line interface
- Variant of Shell
  - Bourne shell
  - Korn shell
  - Bourne Again shell
  - C Shell
  - Z Shell
  - TC Shell

Variant of Shells

Shell	Descriptions
Bourne	First initial shell Limited functionality
C Shell	Based on C Programming language For programming Features : Command-line editing and history
Korn	Based on Bourne and C Shell For scripting Widely used in Unix platforms
BASH (Default)	GNU Project Based on Bourne, C Shell and Korn More user friendly Has lots of shortcut key Mapped history to navigational key
TC Shell	Improvised from C Shell
Z Shell	Improvised from Korn shell Widely used in Unix platforms Based on Bourne, Korn and TC Shell

Gaining Access to Shell

- Console
- Thru Terminal (any desktop environment)
- Thru remote connection (ssh, rlogin, telnet)

Entering Commands

- The syntax
  - Command
  - Parameters
  - Argument

- Example

  # ls -ltr /etc/passwd
  - Command : ls
  - Parameters : -ltr
  - Argument : /etc/passwd

  # cat /opt/eap-acme/postfix.cfg | more
  - Command : cat and more
  - Parameters : none
  - Argument : /opt/eap-acme/postfix.cfg

Quiz

- The syntax
  - Command
  - Parameters
  - Argument
- Example

  # ansible all -m ping
  - Command :
  - Parameters :
  - Argument :

Quiz

- The syntax
  - Command
  - Parameters
  - Argument
- Example

  # ansible all -m ping
  - Command : ansible
  - Parameters : -m
  - Argument : all and ping

Quiz

- The syntax
  - Command
  - Parameters
  - Argument
- Example

  # ansible-playbook --syntax-check play.yml
  - Command :
  - Parameters :
  - Argument :

Quiz

- The syntax
  - Command
  - Parameters
  - Argument
- Example

  # ansible-playbook --syntax-check play.yml
  - Command : ansible-playbook
  - Parameters : --syntax-check
  - Argument : play.yml

Cat, touch, ls, mkdir, more

- cat command
  - View file's content
- touch command
  - Create empty file
- ls command
  - List files and directories
- mkdir command
  - Create empty directory
- more command
  - Pause every full content
- rm command
  - Delete file or directory

Entering Multiple Commands

- The piping |
  - Commands are related
  - Output of cmd1 became input of cmd2, and so on
  - cmd1 | cmd2 | cmd3
- The semicolon ;
  - Commands do not have to be related
  - cmd1 ; cmd2 ; cmd3

Using aliases

- Shortcuts to multiple commands

  # alias myinfo=”hostname; uname -a; id; who”

- Shortcuts to multiple parameters

  # alias ls=”ls -ltr”

- Alternative to another command

  # alias ls=”/jason/hacks/ls”

- List all aliases

  # alias

- Remove alias

  # unalias myinfo

Which command

- To locate path to command
- Is alias or pre-built command

Getting Information about Command

- Get short information

  # tar -?

  # tar --help

- Get detailed information

  # man tar

  # Info tar

Working with Shell History

- Show all commands previously executed

  # history

- Show last 20 commands previously executed

  # history 5

- Go previous command
  - Ctrl+p or ctrl+up arrow
- Go next command
  - Ctrl+n or ctrl+down arrow
- Use “!” character
  - To quickly execute previous command

Bash command-line Editing

Control + key	Function
Ctrl+a	Place cursor at beginning
Ctrl+e	Place cursor at end
Ctrl+w	Erase last word
Ctrl+u	Erase whole line
Ctrl+k	Erase to end of line from current cursor position
Ctrl+l	Clear screen
Ctrl+p	Go back previous command, similar to ctrl+up arrow
Ctrl+n	Go next command, similar to ctrl+ down arrow
Ctrl+ left and right arrow	Move from word to word
Esc then F	Place cursor forward one word
Esc then b	Place cursor back one word
Esc then d	Delete one word from current cursor position

AutoCompletion

- Press tab
- Press double tab

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files start with a

ls a*

ls /etc/a*

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files end with g

ls *g

ls /etc/*g

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files containing the word config

ls *config*

ls /etc/*config*

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files start with 3 letters word then config

ls ???config

ls /etc/???config

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files containing the word sda,sdb,sdc

ls sd[a-c]

ls /etc/sd[a-c]

Filename Shorthand

- * to matches all characters
- ? To matches single character
- {,} to matches multiple charaters
- [-] to matches range of occurrences

Example : Show all files start with sd, hd, jd and ends with single letter

ls {sd,hd,jd}?

ls /dev/{sd,hd,jd}?

Redirection

Redirection	Meaning
<	Standard input (default to keyboard) or stdin
<0	Standard input (default to keyboard) or stdin
>	Standard output when command executed successfully or stdout
1>	Standard output when command executed successfully or stdout
2>	Standard output when command executed unsuccessfully or stderr
2>&1	Redirect stderr to stdout
&>	Both stderr and stdout to same file

DEMO

Quiz

1. What does mkdir –p /etc/databases/configurations?
    a) It recursively creates empty files
    b) It recursively preserves configurations
    c) It recursively updates configurations
    d) It recursively creates directories

Quiz

1. What does mkdir –p /etc/databases/configurations?
    a) It recursively creates empty files
    b) It recursively preserves configurations
    c) It recursively updates configurations
    d) It recursively creates directories

Quiz

2. Which send stdout to /tmp/result but stderr to /tmp/error?

a) Cmd > /tmp/result 2> /tmp/error
b) Cmd 2> /tmp/result 1> /tmp/error
c) Cmd 0> /tmp/result 2> /tmp/error
d) Cmd > /tmp/error 2> /tmp/result

Quiz

2. Which send standard output to /tmp/result but standard error to /tmp/error?

a) Cmd > /tmp/result 2> /tmp/error
b) Cmd 2> /tmp/result 1> /tmp/error
c) Cmd 0> /tmp/result 2> /tmp/error
d) Cmd > /tmp/error 2> /tmp/result

Quiz

3. Which command pauses every full screen?

a) sleep
b) cat
c) more
d) pause

Quiz

3. Which command pauses every full screen?

a) sleep
b) cat
c) more
d) pause

Unit summary

Having completed this unit, you should be able to:

- Gaining Access to Shell
- Entering Commands
- Using Aliases
- Getting Information about Command
- Working with Shell History
- Bash command-line Editing
- Autocompletion
- Filename Shorthand
- Redirection