College of Science and Technology

Royal University of Bhutan

LESSON – 5

USING ESSENTIAL TOOLS

ENVIRONMENT VARIABLES

- Understanding the Shell Environment
  - When working with shell, an environment is created to ensure that all that is happening is happening the right way
  - Environment variables: $PATH, $LANG, $HOME
  - Display the current environment: env, printenv
  - Set variable: variable_name = value, unset variable_name
  - BASH shell is widely and most commonly used shell and it is officially distributed with the Linux System (Default shell)

- Environment configuration files:
  - When user logs in, an environment is created for that user automatically (based on /etc/profile, /etc/bashrc, ~/.bash_profile, ~/.bashrc)

ENVIRONMENT VARIABLES

- Using /etc/motd and /etc/issue:
  - /etc/motd file will get Linux users to pay attention. Instead of sending email, the message of the day is being configured and all users will get this message.
  - Adding your own message of the day is easy.

ACTIVITY I

- Configure /etc/motd

1. Log in with root user and type **vim /etc/motd**
2. Type message in the file and save it
3. Switch to virtual console using Ctrl+Alt+F2 and login with user account (non-root user).
4. After login successfully, you will get the message.

ACTIVITY II

- Configure /etc/issue:

1. Log in with root user and type vim /etc/issue
2. Type message in the file after the existing line as:
   // Welcome User.
   // * All connections are monitored and recorded.
   // * * Disconnect IMMEDIATELY if you are not authorized user.
3. Save the file
4. Switch to virtual console using Ctrl+Alt+F2 and login with user account.
5. On the screen, you will see the message

EXECUTING COMMANDS

- Basic Commands:

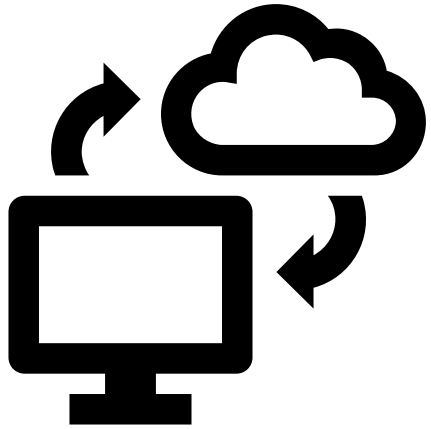
- Command is an instruction given to perform task through shell in both graphical terminal and in virtual console prompt
- **Syntax:**

Command [Options] Arguments

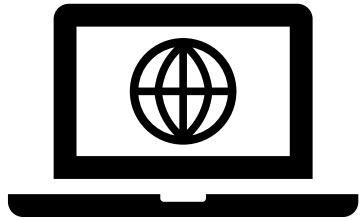
- Each item of the command is separated by space
- Options modify a command's behavior. Single letter option preceded by (-) and full-word (--)
- Example: -r, -f, -rf, --info, --force, --help

STANDARD I/0

INPUT Unit



Central Processing Unit



Output Unit

STANDARD INPUT, OUTPUT, & ERROR

Name	Default Destination	Use in Redirection	File Descriptor
STDIN	Computer keyboard	<	0
STDOUT	Computer monitor	>	1
STDERR	Computer monitor	2>	2

COMMON BASH REDIRECTION

- Use of direction operator.

Redirector	Explanation
> (same as 1>)	Redirects STDOUT. If direction is to a file, the current content of that file is overwritten
>> (same as 1>>)	Redirects STDOUT. If output is written to a file, the output is appended to a that file.
2>	Redirects STDERR
2>&1	Redirects STDERR to the same destination as STDOUT
< (same as 0<)	Redirects STDIN

Examples of Bash Redirections

- Overwrite Redirection (replace all the existing content):

  $cat > file.txt

  Type anything. When you are done press Ctrl+d

  $cat file.txt **or** **$cat < file.txt**

- Append Redirection (you can append the output):

  $cat >> file.txt

  Type anything. When you are done press Ctrl+d

  $cat file.txt **or** **$cat << file.txt**

  - The delimiter marks the ending point of the document

Examples of Bash Redirections

- Merge Redirection (redirect the output of a command or a program):
- If errors appear, they are managed differently. The stderr operator is 2>
- Example

$lls -> what is the output?

$lls 2> /dev/null-> what is the output?. The resulting error message is redirected to /dev/null instead of the stdout, so no result or error message is displayed on the screen.

Run:

$lls 2>error.txt

$cat error.txt -> what is the output?

Question:

a. Merge the content of two files using cat command.

b. Merge the contents of two files and save it in a third file using cat command.

USING PIPES (|)

- A pipe is a mechanism by which the output of one program(command) can be sent as the input to another program (command).:

  Example:

  ls | *less*

  cat /etc/passwd | *less*

- Linux administrator uses pipes a lot
- Using pipes makes Linux a flexible OS; by combining multiple commands, you can create kind of super commands that make almost anything possible.

ACTIVITY III

- I/O Redirection and Pipes:

1. Open a shell and type ***cd*** without any arguments. This ensures that the home directory of the user is the current directory.
2. Type ***ls***. You'll see the result on the screen
3. Type ***ls > /dev/null***. This redirects the STDOUT to the null device. Notice you will not see the result.
4. Type ***ls qwerty > /dev/null***. You will see "no such file or directory" message onscreen. This is not STDOUT, but an error message written to STDERR
5. Type ***ls qwerty 2> /dev/null***. Now, you will not see the error message anymore

ACTIVITY IV

- Pipes:

Takes the output of the first command and makes it the input of the second command.

1. Create a file eg.: ITM302.txt
2. Note down some of your friends name. Make few duplicates.
3. Save and check the content of the file.
4. Type ***cat ITM302.txt | grep name*** -> what is the output?
5. Type ***cat ITM302.txt | grep "name"*** -> what is the output?
6. Type ***cat ITM302.txt | grep "name" | wc -l*** -> what is the output?

FILE MANAGEMENT

- Managing Files:

As an administrator, you need to be able to perform common file management tasks.

1. Working with wildcards
2. Managing and working with directories
3. Working with absolute and relative pathnames
4. Listing files and directories
5. Moving files and directories
6. Deleting files and directories

MANAGING FILES

• Working with Wildcards (Globbing) :

Using wildcards can make your work a lot easier:

1. * (matches zero or more characters)
2. ? (matches single character)
3. [0 -9] (matches range of numbers)
4. [abc] (matches any of character in the list)
5. [^abc] (matches all except characters in the list)

MANAGING FILES Cont.

1. * (matches zero or more characters)

   $ ls file* **Or** **$ ls *.txt**

2. ? (matches single character)

   $ ls file?.txt **Or** **$ ls ?ile.txt**

3. [0 -9] (matches range of numbers)

   $ ls file[1-4].txt

4. [abc] (matches any of character in the list)

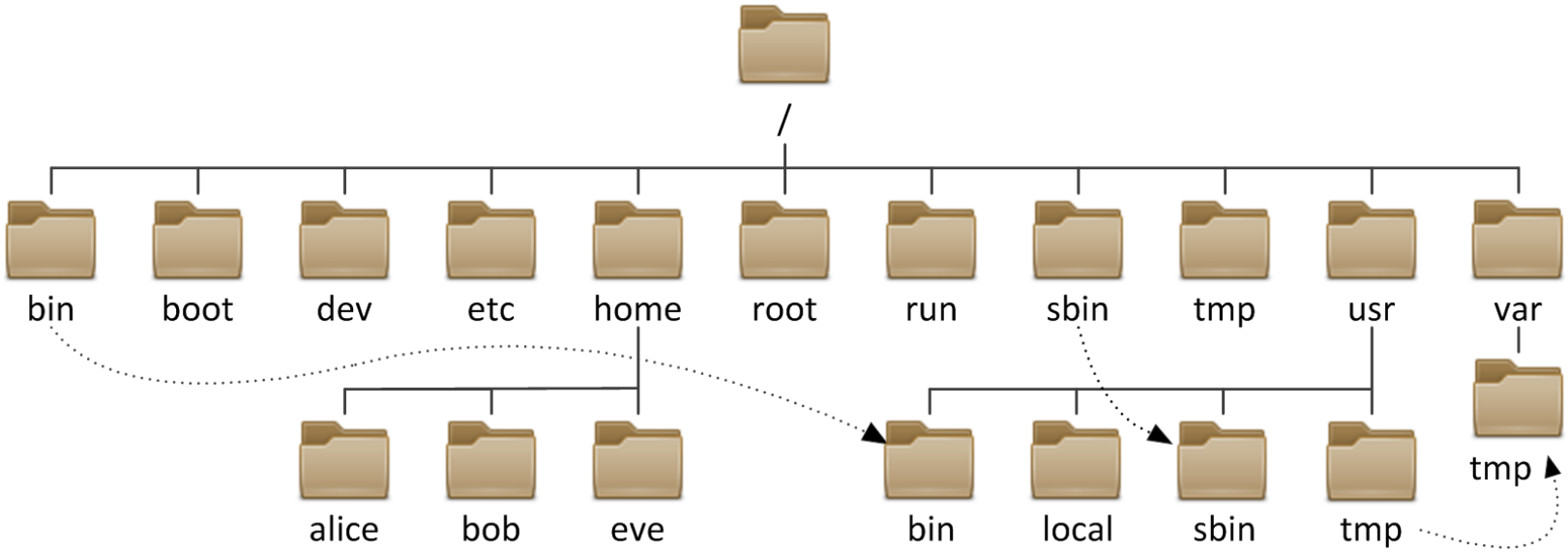
   $ ls fil[e].txt

5. [^abc] / ignore (matches all except characters in the list

   $ls –ignore=file1.txt

FILE SYSTEM HIERARCHY (FSH)

1. A file system is a data structure that resides on a disk
2. The single-rooted inverted-tree structure helps to identify the Linux file system.
3. It consists of a set of connected files allowing user to organize files for the easy accessing or locating.



LINUX FILE SYSTEM

1. File System hierarchy
   - The file system on most Linux systems is organized in a similar way.
   - Layout of the Linux file system is defined in the Filesystem Hierarchy.
   - Defined in ***man 7 hier***.

2. Usage of df command: (df –hT)
   - Shows seven columns: Filesystem, Type, Size, Used, Avail, Use%, Mounted on

   - Example:
     - df –a
     - df –h
     - df –i (The inodes are data structures that store information about files and directories, such as ownership, permissions, and timestamps.)
     - df –T (to check the file system type of your system)

     Question: What is the diff. between ***df*** and ***du*** command?

HISTORY

- A convenient feature of the bash shell is the bash history.
- Bash is configured to keep the last 1,000 commands you have used.
- The history feature makes it easy to repeat complex commands.
- Commands to use History features:
  - history (history –c, history –w)
  - Ctrl + R
  - !number
  - !sometext

- Change history size from .bashrc file.

Basic commands

- date
- cal (Example: cal 08, cal 09 2012)
- who - who am I - whatis (whatis ls) and whereis (whereis ls)
- head and tail
- more and less
- cat
- pwd

USING COMMAND LINE SHORTCUTS

- Bash Completion (Tab Key command line shortcuts)
  - Very, very powerful
  - Example: ls Doc <tab> (Autocompletes to file name), ca <tab> completes cat command

- We'll do this now:
  - cat /etc <TAB twicely quickly>
  - cat /etc/netw <TAB>
  - cat /etc/network/in <TAB >

OBTAINING HELP

- man Command
  - “man” is manual command
  - Usage: man - Example: man ls, **man man**, man –k partition
  - Different sections: (1)-Executable programs/shell commands, (5)-file formats and conventions, (8)- System administrator commands
  - **man man** (will explain to you that these are section numbers)

- --help Command
  - Example: help --help

- info Command
  - Example: info grub

SUMMARY

- In this lesson, you have learnt that:

- Environment variables are important for administrator
- Message to the user can be configured using /etc/motd and /etc/issue files
- Command line shortcuts is very, very important for good administrator.
- You don’t have to remember all commands as there are alternatives of using command to remember commands