b

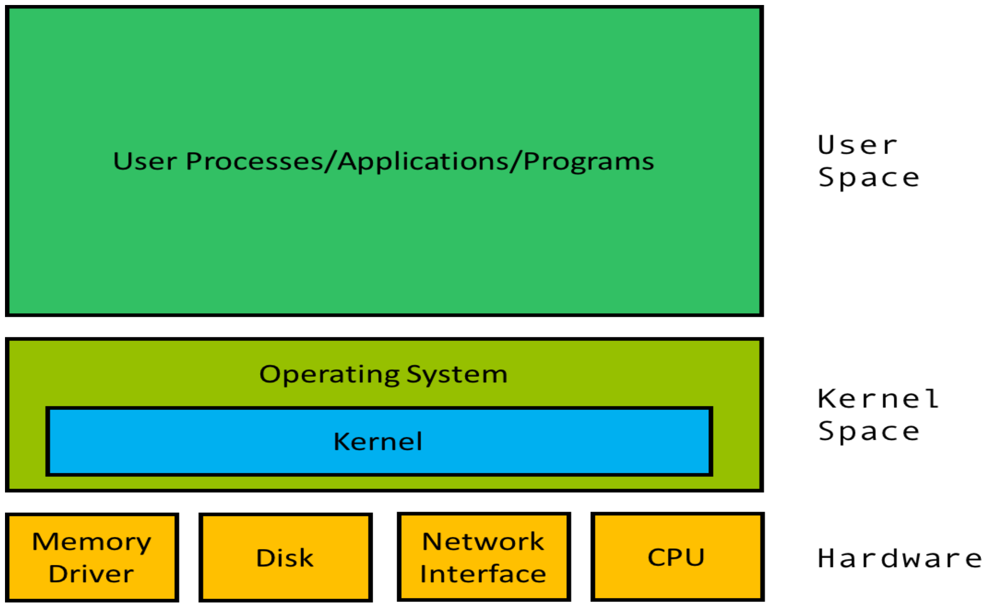
Workshop On
Linux Environment
Shell Scripting
Vim

Presented by:KOSS

Linux

Linux is a Kernel

kernel is a software written just above the hardware.



Why Linux???

- Code is open sourced
- Security - Most common malicious programs target windows.Linux is inherently more secure than windows
- Popular Linux distros are *Fedora*, *Ubuntu*, *Kali Linux*
- Easy to install programming tools and software not matter what field you are
- Updates are peaceful.

GUI Versus CLI

- GUI:Graphical User Interface
  Uses graphics,just like Icons ,to interact with the operating system
  A GUI needs to use additional system resources to load the graphical part thus it is going to be slower than using the command line
- CLI:Command Line Interface
  Uses textual commands to interact with the operating system
  CLI provides greater flexibility of use.It can be used to easily do things that are difficult or even impossible to do with a GUI.

What is a Shell?

- Shell is a program that lets you run shell commands.
- A Shell is a command line interpreter, which runs your shell scripts.
- Examples: tcsh , fish , **Bourne Again Shell (bash)**, zsh etc.

*Type **$cat /etc/shells** to display the shells available in your system*

Linux Directory

- */bin :Binary files*
- */boot : Boot loader files*
- */dev :device files*
- */lib : system libraries*
- */tmp : Temporary Files*
- */home :home directories .... and a lot more*

Basic linux Commands

pwd

- pwd stands for Print Working Directory.
- It prints the path of the working directory, starting from the root.

ls

- This is used to list files.
- Usage: ls -[Options] [path] *{Note:If a path is not given, current directory is assumed.}*
- Path can also contain wildcards. Example: ls *.pdf will list all the pdf files in that directory
- Options consist of one or a combination of character flags that invoke special functions:
  - l : List files with additional metadata
  - a : Show hidden files also. (Name begins with .
  - h : Show file sizes in MBs and GBs instead of bytes.
  - t : Sort by date modified.
  - r: option flag lists files/directories in reverse order.
  - Multiple options can be combined as: ls -lah .

Excercise: Open a terminal and create 2 hidden files and 2 visible files.Then the list of all files (both hidden and visible) .Show files sorted by the date modified and with reverse date modified.

cat,head ,tail

- cat stands for conCATenate .
- cat file1 file2 file3 will output all the 3 files combined in the given sequence.
- However, in practice, people use cat to print out the 1 full file only.
- head:prints out the first few lines of a file and tail:prints out the last few lines of a file.
- Both accept a parameter number n, This limits the output to n number of lines
- Example: to get the first 15 lines of a line, *run:head -n15 file*

echo and printf

- It is used to print stuff
- printf supports formatted output.
- UnLike C, echo is more common in use than printf
- Usage: echo "some text" or printf "format string" "parameters"

Operators for redirecting output from specific files / streams are:
> outputfile Redirects the output to a new file (Existing file is overwritten).
>> outputfile Appends the output to a file.

find

- find is quite powerful as a utility
- Its basic task is to recursively print out all the files and directories from a given path.

```
$ find . # Search starts from current directory
$ find / # Search starts from root
```

- Although there are many filters and actions that find can perform
- For example, find -type f finds only files. Change f to d and it will find only directories.

- find -name expr will match expr to the file names. expr can be a string with wildcards
- These filters can also be combined
- The default action is -print .
- However find -delete will delete all the files it was supposed to print.
- Furthermore, find -exec will execute arbitrary command on the file names.

> *For example:*
> *find -name "*.txt"*
> *find the text files*

- Know more by running man find

grep

- grep prints those lines in a given list of files that match a pattern.
- Usage: grep pattern filename .
- Another common usage is to show the output of some other command to grep. For example:

```
grep "name" in name.txt
```

- This will find all lines in a file that have the string “name” in it. Use -i for case insensitive output

which

- Most command that you run in the shell actually is an executable located somewhere in your PATH (it is an environment variable, more on that later).
- To find out which particular executable is being run, which is

```
$ which echo
/usr/bin/echo
```

cp, mv and mkdir

```
cp some/path/file some/other/path
```

- copies file from *some/path* to *some/other/path* .
- To recursively copy a folder and all its files and subdirectories, we use *cp -r* .
- The main job of*mv*is to move files and folders from one directory to the other
- Althought *mv file newname* renames the file *file* to *newname*
- Paths in cp and mv also support wildcards. For example,*cp yt-slides/*.pdf folder2/* copies only the pdf files.
- *mkdir* makes directories. *mkdir existing/path/new_directory_to_make* . This creates a new directory *new_directory_to_make* under the existing path *existing/path* .
- However, if the parent directory doesn't exist yet, we can create the whole hierarchy

rm and rmdir

- rmdir removes empty directories.
- rm is a general command for removal of files and folders.
- To recursively delete, use the -r flag with rm .
- To forcefully delete, use the -f flag with rm .

*Use rm -rf **Responsibly** as it can delete any file and maybe your system*

wc, sort

These fall under the category of text manipulation programs.

- wc returns the newline, word and byte count for each of the files that are passed to it.
- We can get the individual newline, word or byte count by using -l , -w or -c flags respectively.
- sort sorts the lines of a document in lexicographical order. Although the ordering can be changed using appropriate flags.
- sort -u gives the unique lines in the document
- To sort files numerically use -n

wget and curl

- These programs are used to fetch resources from the internet.
- wget , as the name suggests, performs only GET requests.
- By default, wget saves the output to a file in the current directory. However, this can be changed using the -o flag.
- cURL is a more generic tool. It can be used to perform arbitrary HTTP requests.

For example, sending a POST request to an URL through curl is as follows: curl -X POST -H 'Content-type: application/json' -d '{"message": "Hello"}' http://url/endpoint -X defines request method, -H defines headers, -b defines Request body.

Pipe (|)

- This direct connection between commands/ programs/ processes allows them to operate simultaneously and permits data to be transferred between them continuously
- command_1 | command_2 | command_3 | .... | command_N

Handy Commands

Look for the usage of these commands:

man : for manual
example:man echo,man ls
history : for viewing your terminal history
time : to check the compilation time

Variables and Control FLow

variables

- Variables here are not typed.
- All variables, when USED should be preceeded by $ symbol.
- However, while declaring you should never use the $ symbol

Example:
$ a=2 # Don't forget to put no space around equals
$ echo $a # Btw this is a comment.

Environment Variables

- Variables that you want save in your system so that you can access from anywhere
- For temporary usage use *export variable_name=value*
- For permanent usage save it in your ~/.bashrc file

Single and double quotes in bash

In bash, both single and double quotes are allowed. However there is a subtle difference in behaviour. Inside double quoted string, you can use sub-commands enclosed by $() . This is not possible with single quotes.
Exercise: Run echo"$(ls /bin)" and echo '$(ls /bin)' to see the difference

.sh scripts

- Apart from running from terminal, we can also put our commands in a script file.
- Files can be run as sh script.sh .
- To be able to run the script as an executable, we need to set executable flag on it. This is done by: *chmod +x script.sh*
- But before that, we need to declare which shell to use to run it.
- This is done on the very first line of the script, by writing:

#!/bin/bash *This is called the Shebang line. Following the previous 2 steps, one can then execute the script as:**$./script***

Conditionals

Bash doesn’t use braces or indentation to mark the blocks.

The basic structure of an if block is as follows:

```
if [condition1]
then
# Block to execute
elif [condition2]
then
# Block
else
# Block
fi
```

- [] are a reference to the test command, which is run internally to check for the conditions.

- Normal operators like = , != apply to String comparison.
- Integer comparisons are done using -eq , -gt and -lt . (Guess their meaning!).
- ! Expr negates the expression Expr .
- Some special comparisons:
    - -n str : Length of string is > 0.
    - -z str : Length is == 0.
    - -d file : file is an existing directory.
    - -e file : file exists
    - -r file : file exists and the read permission is granted.
    - -s file : file exists and is not empty
    - -w file : file exists and the write permission is granted
    - -x file : file exists and the execute permission is granted.

For Loops

- Bash’s for loop is similar to the Python one.
- Although the concept of array is not present in bash.

Common patterns: for i in var1 var2 var3 do # Do something with $i # vari can be numbers also done *for i in $(Command with multiple line output) do # $i will contain one line at a time done*

for i in {1..5} # {START..STOP} range for i in {1..5..2} # {START..STOP..STEP} range for i in $(seq 1 100) # 1 to 100 sequence

While loop

- while loops iterate while their conditions are true.

Syntax: while [condition] do # Something to do done

- A common pattern observed while using the while loop is incrementing the variables. This is done as shown: x=$(($x + 1)) .
- Another common usage of while is with the read command:

```
while read p do # Something with $p done
```

This reads from stdin line by line until EOF is received.

- while with no condition is an infinite loop.
- break and continue work as common sense predicts
- Exercise: Read about case statement

Hands On Excerise

Problem

create 2 directories - c_codes and py_codes
using loops create 6 files 1.c, 3.c, 5.c 2.py, 4.py, 6.py
put odd numbered files in c_codes directory put even numbered files in py_codes directory