**Sir – Dheeraj madhukar**

**Type of OS:**

``

1. Single User Single Tasking [DOS]

2. Single User Multitasking [Win]

3. Multiuser Multitasking [Linux/Unix]

4. Real Time [flight radar system]

5. Embedded [Robotics]

**Linux is Open Source-**

```

- Free to study

- Free to modify

- Free to dist.

---------------------------------------------------

**Structure of OS**

````

kernel kernel

user --i/p--> SHELL --i/p--> h/w

user <--o/p-- SHELL <--o/p-- h/w

**Mr. linus Torvalds in 17 sept 1991 (also Github)**

* Linux is an a kernel not an operating system.
* Where Unix is commercial & Linux is open source .
* Linux can use Shell name – BASH

BASH-(Bourne Again Shell)

**History of SHELL:**

``

SH [SHELL] : UNIX shell

KSH [Korn Shell] : UNIX Script

CSH [C Lang] :

TCSH [Turbo C]

BASH [KSH + TCSH]

--------------------------------------------------

GUI

CLI

\\_TUI [Text-mode user interface]

-----------------------------------------------

**Installation:**

Recommanded: 20G

Partitions

1) / [Parent Partition]

2) /boot [Booting Config]: GRUB [512MB]

3) SWAP [RAM x 2] =8G

4G + 4G [HDD]

A part of HDD act as RAM, virtual mem.

--------------------

FORMAT

**File Systems:**

````

ext2,ext3,ext4,xfs

exFAT [FAT,FAT16,FAT32]

Download OS [iso]

VMware Workstation

Server Side OS: CentOS 7

Debian : ubuntu, Kali etc

RedHat : Centos & RedHat etc

HOST OS: WIN

Virtualization : VMware Workstation Pro

sda

Storage Device a

sda1

sda2

sda3

--------------------

**[root@localhost ~]#**

root : USERNAME

localhost : HOSTNAME

~ [Tilde] : Home Dir of loggedIn user

Regular User: /home

Super User : /root

# : Loggedin as super user

$ : LoggedIn as regular user

-----------------------------------------

<https://tryhackme.com/room/linuxfundamentalspart1>

**Basic Users & group management:**

------------------------------

1) Super User

2) Normal User

3) Service Accounts

**=> user related information stored into:**

# cat /etc/passwd

user:x:1000:1000:COMMENT:/home/user:/bin/bash

name | UID GID Home Dir Shell

\\_ Password required

UID (0-60000) | GID (0-60000)

---------------------------------------------

0-999 [Super User] | 0-999 [Super group]

1000+ [Normal User] | 1000+ [normal Group]

-----------------------------------------------------

=> user password related information stored into:

# cat /etc/shadow

user:Encrypted Password:Acc. Age

SHA512 + Salt Algo

# chage -l username

**Groups:**

-------

=> Group related information stored into:

# cat /etc/group

grp:x:GID

=> Group password related information stored into:

# cat /etc/gshadow

grp:encrypted pass

================

**a) Create user:**

```

# useradd username

OR

# adduser username

**b) Assign/Change the password:**

# passwd username

# passwd -d username

[-d : delete]

**c) Remove User:**

- Without home dir:

# userdel natasha

- with home dir

# userdel -r username

**d) Create a group:**

# groupadd groupname

**e) Assign/Chnage Password:**

# gpasswd groupname

**f) To delete a group:**

# groupdel groupname

**To switch the user:**

# su - username

$ exit [Logout]

# id username

======================================

**Basic Permissions: [ File based ]**

------------------

**1) Symbolic Method**

===================

read : r

write : w

execute : x

------------

user owner : u

group owner : g

Others : o

All : a

------------

Assign : +

Remove : -

Overwrite : =

------------

Permission Area:

--- : user owner

--- : group owner

--- : Others

# ls -l

-rw-r--r--. 1 root root 43 Sep 23 01:56 file1

**File Type:**

````

- : Regular file

d : Dir

c : char /dev/tty1

b : block /dev/sda1

p : pipe

s : socket

l : Link

Default permissions for file: 644

-----------------------------

read & write : u

read : g

read : o

Default permissions for dir: 755

----------------------------

read, write & execute : u

read & execute : g

read & execute : o

**Change permissions in Symbolic menthod:**

---------------------------------------

# chmod u+rx file/dir

# chmod u-r,g+rw,o-rwx file/dir

# chmod u=rw file/dir

# chmod a=r file/dir

# chmod ugo=r file/dir

**2) Numeric Method**

-----------------

read : 4

write : 2

execute : 1

Full : 7

No : 0

--- 000 0

r-- 100 4

-w- 010 2

--x 001 1

rwx 111 7

# chmod 711 file/dir

# chmod 111 dir -R

[-R : recursively]

========================

- **ACL [ Access Control List ]**

System provide the permissions to the users/groups for accessing any objects but without changing any ownership & without assigning any permission to the other.

**How to assign ACL:**

------

# setfacl -m u:username:rw- file/dir

# setfacl -m g:groupname:r-x file/dir

-m : Modify

# ls -l

. [No ACL]

+ [ACL Applied]

**Check the ACL for File/Dir**:

---------------------------

# getfacl file/dir

# file: marvels

# owner: root

# group: root

user::rw-

user:kiosk:r--

group::r--

mask::r--

other::r—

**Ownership:`**

**user Owner:**

```

# chown user1 file/dir

**group owner**

``

# chgrp grp1 file/dir

**Membership:**

``

**a) create a user as a secondary member of a group**

# groupadd grp1

# useradd -G grp1 u1

-G : Secondary Membership

**b) add an existing user as secondary member of a group**

# usermod -G grp1 u2

**Q. Create the following users, groups, and group memberships:**

-A group named admin

-A user harry who belongs to admin as a secondary group

-A user natasha who also belongs to admin as a secondary group

-A user sarah who must have access to an interactive shell on the system, and who is not a member of admin

-harry, natasha, and sarah should all have the password as pas

# groupadd admin

# useradd -G admin harry

# useradd -G admin natasha

# useradd sarah

# passwd users

**Q. Create a collaborative directory /common/adm with the following characteristics:**

-Group ownership of /common/adm is admin

-The directory should be readable, writable, and accessible to members of admin, but not to any other user. (It is understood that root has access to all ﬁles and directories on the system.)

# mkdir -p /common/adm

# chgrp admin /common/adm

# chmod 770 /common/adm

**IP & MAC:**

```

# ifconfig

GW IP:

```

# route -n

192.168.206.2

+ Auto IP Assign [proto: DHCP]

# dhclient

+ manual network config [Server : CentOS 7]

IP: 192.168.206.140/24

GW: 192.168.206.2

# nmtui [ Network manager TUI ]

Service: network

# systemctl restart network

**To handle services:**

systemctl :

Actions:

``

start,stop,restart,reload,status .....etc

Command:

# systemctl <action> <service>

ping x.x.x.x

Machine1: 192.168.206.104 [CentOS]Server

Machine2: 192.168.206.133 [Ubuntu]Client

-------------------------------------------

**Package Manager:**

````

To install, uninstall, update, upgrade etc...

CentOs: YUM/DNF

**1) UPDATE:**

`

# yum update

**2) UPGRADE:**

``

# yum upgrade

**3) DIST UPGRADE:**

````

# yum dist-upgrade

**4) SEARCH:**

`

# yum search package\_name

**5) INSTALL:**

``

# yum install package\_name

**6) REMOVE:**

`

# yum remove package\_name

**7) LIST of INSTALLED PACKAGES:**

``````

# yum list

**TELNET:**

`

Used to access the remote machine on CLI.

Package: telnet-server xinetd

Port: 23

service/Daemon: xinetd

**1) Install Packages:**

# yum install telnet-server xinetd -y

**2) Start services:**

# systemctl start xinetd telnet.socket

# systemctl status xinetd telnet.socket

**3) Check for listening port**

# lsof -i:23

**Client Machine:**

```

# yum install telnet -y

# apt install telnet -y

**Access:**

`

# telnet x.x.x.x

Trying 192.168.206.140...

Connected to 192.168.206.140.

Escape character is '^]'.

Kernel 3.10.0-1160.el7.x86\_64 on an x86\_64

localhost login: natasha

Password: \*\*

[natasha@localhost ~]$

**Extra:**

``````

# lsof -i:23

Look for PID

# kill pid

**IPTABLES:**

```

check:

# iptables -L

OR

# iptables -S

**Temp Flush**

````

# iptables -F

# ip6tables -F

**Extra:**

``````

# lsof -i:23

Look for PID

# kill pid

**vsFTP [Very Secure FTP]**

`````

Package: vsftpd

service: vsftpd

port: 21

config file:

1) # yum install vsftpd -y

**2) Start the service**

# systemctl start vsftpd

3) # lsof -i:21

**4) Config the service:**

# vim /etc/vsftpd/vsftpd.conf

**5) Default Dir:**

DocumentRoot: /var/ftp/

# cd /var/ftp

# mkdir data

**On Client Machine**

``````

# yum install ftp

# apt install ftp

# ftp x.x.x.x

Connected to 192.168.206.140.

220 (vsFTPd 3.0.2)

Name (192.168.206.140:root): ftp

331 Please specify the password.

Password:

230 Login successful.

ftp> ls

**Download a file**

ftp> get filename

**Download a dir**

ftp> mget dir

**Upload a file**

ftp> put file

**Upload a dir**

ftp> mput dir\_path

ftp> bye

**SSH [Secure Shell]**

````

To access the remote machin on CLI

Package: openssh-server

Port: 22

service: sshd

config:

1) # yum install openssh-server -y

**2) Start the service**

# systemctl restart sshd

3) # lsof -i:22

**On Client Machine:**

```

# yum install openssh-client -y

# apt install openssh-client -y

**Access:**

`

NOTE: By default root access is not allowed.

# ssh x.x.x.x [ Always try to login as root ]

# ssh username@x.x.x.x [ Login as user ]

**ENABLE ROOT ACCESS: [On Server Machine]**

``````

# vim /etc/ssh/sshd\_config

UNCOMMENT "PermitRootLogin" {remove '#'}

:wq

THEN RESTART THE SERVICE TO APPLY CHANGES

# systemctl restart sshd

**Process Management:**

````

**Process:**

``

- Any running program is a process

- Multiple instances of the same program are processes.

- Shell is also a process

**Process ID (PID)**

````

- Each linux process is identified by unique id PID

- Every process has a Parent Process ID (PPID)

+ Except "init"

- When a process is loaded into memory there is structure:

+ stack

+ heap

+ data segment

+ code segment

[+] Stack: Used for static memory allocation.

[+] Heap: Used for dynamic memory allocation.

[+] Data: Stores any static or global variables if defined.

[+] Code: Is the instructions of the program.

**Create a Process:**

``

Terminal#1

``

# sleep 100

Terminal#2

``

# ps -la [ Show the processes with PID & PPID ]

**Process States:**

**```**

- The process is built and executing , so now...

- We enter the Process State Machine

- States of Process:

[N] New When a new process is being created

[R] Running Instructions are being executed

[W] Waiting The process is waiting for some event to occur

[R] Ready The process is waiting to be assigned to a processor

[T] Terminated The processes has finished execution and is exiting

**Managing Processes:**

`````

- Linux kernel tracks what each process is doing

- Process is assigned a priority

- Address space assigned to the process

- Files is the process allowed to access

- Is the process a ?:-

+ Parent

+ Child

+ Zombie

=> Sometimes the parent dies first

````

- If the parent just exits or dies, the child process is left running

- The child's PPID is no longer valid due to parent is gone.

- Linux calls these children processes "Zombies"

- In Linux a zombie is just a process in which the children are adopted by the "init" process

- The init process will eventually cleanup the zombie childrens

PPID[kill]

\\_\_\_\_\_\_\_\_\_\_\_PID[kill]

PPID[die]

\\_\_\_\_\_\_\_\_\_\_\_PID[zombie]

**Process Termination:**

``

- When a parent forks a child, they can finish in any order( parent first or child first)

- Sometimes the parent process could encounter and error and die

- Sometimes the parent process will just wait around until the child processes all complete brfore exiting

+ it calls a wait() command

- There are variety of wait command.

**process manager:**

````

ps, top, htop

`````

# top

press 'q' to exit

# sleep 10 &

# ps [ PID - Process ID]

# ps -l [ Show the PPID - Parent PID ]

# ps -e [ All the processes ]

# ps -la

**Kill:**

-----

# kill PID

OR

# kill -9 PID

-9 : Forcefully

# pkill sleep

# killall sleep

--------------------

# sleep 10000 &

# sleep 20000 &

**Check backgroup process:**

# jobs

[1]- Running sleep 10000 &

[2]+ Running sleep 20000 &

- : Second last added in jobs

+ : last added in jobs

# jobs -l [ with PID ]

[1]- 971 Running sleep 10000 &

[2]+ 972 Running sleep 20000 &

# fg %2

ctrl+z [stop]

# bg %2 [ To start in BG ]

https://github.com/Donivr/Linux-PDF/blob/master/Linux%20Command%20Line%20and%20Shell%20Scripting%20Bible%203rd%20Edition%20%7BPRG%7D.pdf

**Q. Copy the ﬁle /etc/fstab to /var/tmp. Conﬁgure the permissions of /var/tmp/fstab so that:**

-The ﬁle /var/tmp/fstab is owned by the root user.

-The ﬁle /var/tmp/fstab belongs to the group root.

-The ﬁle /var/tmp/fstab should not be executable by anyone.

-The user harry is able to read and write /var/tmp/fstab.

-The user natasha can neither write nor read /var/tmp/fstab.

-all other users (current or future) have the ability to read /var/tmp/fstab

# cp -av /etc/fstab /var/tmp

# chown root:root /var/tmp/fstab

# setfacl -m u:harry:rw- /var/tmp/fstab

# setfacl -m u:natasha:--- /var/tmp/fstab

# chmod o=r /var/tmp/fstab

**#!/bin/bash**

echo "Total users: $(cat /etc/passwd | awk -F":" '{print $1}' | wc -l)"

echo "CPU Core: $(cat /proc/cpuinfo | grep "cpu cores" | sort -u | awk -F":" '{print $2}')"

echo "RAM: $(free -mh | grep "Mem" | awk '{print $2}')"

echo "IP: $(ifconfig wifi0 | grep "inet" | grep -v "inet6" | awk '{print $2}')"

#!/bin/bash

echo $((2 + 2))

echo $((2 - 2))

echo $((2 / 2))

echo $((2 \* 2))

expr 2 + 2 expr 2 / 2

expr 2 – 2 expr 2 \* 2

[[ STRING == STRING ]] Equal

[[ STRING != STRING ]] Not Equal

[[ NUM -eq NUM ]] Equal

[[ NUM -ne NUM ]] Not equal

[[ NUM -lt NUM ]] Less than

[[ NUM -le NUM ]] Less than or equal

[[ NUM -gt NUM ]] Greater than

[[ NUM -ge NUM ]] Greater than or equal

[[ STRING =~ STRING ]] Regexp

**File Ops:**

```

[[ -e FILE ]] Exists

[[ -r FILE ]] Readable

[[ -d FILE ]] Directory

[[ -w FILE ]] Writable

[[ -f FILE ]] File

[[ -x FILE ]] Executable

[[ -z STRING ]] Empty string

[[ -n STRING ]] Not empty string

#!/bin/bash

read -p "Enter the value " a

read -p "Enter the value " b

read -p " Enter the Expression" c

if [[ $c == "+" ]]; then

echo "$((a + b))"

elif [[ $c == "-" ]]; then

echo "$((a - b))"

elif [[ $c == "/" ]]; then

echo "$(( a / b))"

elif [[ $c == "\*" ]]; then

echo "$((a \* b))"

else

echo "Enter the Valid Expression"

fi

#!/bin/bash

which ifconfig

if [[ $? == 0 ]];then

echo "comm exists"

else

echo "comm does not exist"

fi

**A linux process has three standard streams:**

stdin (file descriptor 0)

stdout (file descriptor 1)

stderr (file descriptor 2)

**stdin defaults to your keyboard**

stdout and stderr default to your screen

You can redirect the standard streams

‘< file’ connects a file to stdin

‘> file’ redirects stdout to a file

‘2> file’ redirects stderr to a file

‘&> file’ redirects stdout and stderr to a file

‘2>&1’ redirects stderr to stdout!

cat > subs.txt

admin

www

login

feedback

contact

lib

about

help

qa

search

api

careers

# cat subs.txt | while read sub;do echo "$sub.cdac.in"; done

# if host careers.cdac.in &> /dev/null;then echo "Alive";fi

while read sub;do

**if** host $sub.cdac.in &> /dev/null;then

echo "$sub.cdac.in : Alive"

fi

done < subs.txt

#!/bin/bash

function main(){

**while** read sub;do

if host "$sub.yahoo.com" &> /dev/null;then

echo "$sub.yahoo.com : Alive"

fi

done < subs.txt

}

function help(){

echo "-d/--domain : Domain as target"

echo "-h/--help : help"

}

**case** $1 in

"-d"|"--domain")

main

;;

"-h"|"--help")

help

;;

\*)

echo "Invalid argument : use -h/--help."

Esac

#!/bin/bash

**function** main(){

while read sub;do

if host "$sub.$domain" &> /dev/null;then

echo "$sub.$domain : Alive"

fi

done < subs.txt

}

function help(){

echo "-d/--domain : Domain as target"

echo "-h/--help : help"

}

case $1 in

"-d"|"--domain")

domain=$2

;;

"-h"|"--help")

help

;;

\*)

echo "Invalid argument : use -h/--help."

exit 127

;;

esac

if [[ -z $domain ]];then

echo "Domain is empty or invalid. Use -h/--help."

exit 127

fi

main

#!/bin/bash

function main(){

while read sub;do

if host "$sub.$domain" &> /dev/null;then

echo "$sub.$domain : Alive"

fi

done < subs.txt

}

function help(){

echo "-d/--domain : Domain as target"

echo "-h/--help : help"

}

case $1 in

"-d"|"--domain")

domain=$2

;;

"-h"|"--help")

help

;;

\*)

echo "Invalid argument : use -h/--help."

exit 127

;;

esac

if [[ -z $domain ]];then

echo "Domain is empty or invalid. Use -h/--help."

exit 127

fi

main

#!/bin/bash

function main(){

while read sub;do

if host "$sub.$domain" &> /dev/null;then

echo "$sub.$domain : Alive"

fi

done < $file

}

function help(){

echo "-d/--domain : Domain as target"

echo "-h/--help : help"

}

while true;do

case $1 in

"-d"|"--domain")

domain=$2

shift

;;

"-f"|"--file")

file=$2

shift

break

;;

"-h"|"--help")

help

;;

\*)

echo "Invalid argument : use -h/--help."

exit 127

;;

esac

shift

done

if [[ -z $domain ]];then

echo "Domain is empty or invalid. Use -h/--help."

exit 127

fi

if [[ -z $file ]];then

echo "Must provide a wordlist. Use -h/--help."

exit 127

fi

main

**CASE:**

``

#!/bin/bash

read -p "Enter: " x

case $x in

"a"|"b")

echo "WoW"

;;

"c"|"d")

echo "HPCSA"

;;

\*)

echo "Bye"

;;

esac

# bash tool.sh -h

# bash tool.sh --help

-d : Domain name as target

-h : Help

# bash tool.sh

Invalid argument : use -h/--help.

# bash tool.sh -p

Invalid argument : use -h/--help.

# bash tool.sh -d

**execute**

+++++++++++++++++++++++++++

**Argument:**

```

**Positional Parameter:**

# set this is sparta

# echo $1

this

# echo $2

is

# echo $3

sparta

# echo $#

3

# echo $@

this is sparta

-----------

#!/bin/bash

echo "First : $1"

echo "Second: $2"

# bash pp.sh lol pop

$0 $1 $2

**FUNCTIONS:**

`

function myFun(){

CODE

}

myFun

# bash tool.sh -d google.com

$0 $1 $2

# bash tool.sh -d

$0 $1 $2

# bash tool.sh

-----------

Problem:

# bash tool.sh -d google.com -f wordlist.txt

$1 $2 $1 $2

OR

# bash tool.sh -f wordlist.txt -d google.com

**Log Analysis**

```

Dir which contain logs: /var/log/

# cd /var/log

# ls

Read the logs Live:

# tail -f log\_filename

**messages:**

```

which maintain system related logs [like: systemctl etc]

dmesg:

Device related logsOR we also have a command: dmesg

wtmp [last reboot or shutdown]

btmp [last login]

``

# utmpdump wtmp

# utmpdump btmp

OR

# lastlog

yum.log: YUM logs

/var/log/audit/

tail audit.log

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++