**COMMANDS**

**Setting Up LAMP Server**

1. $sudo -i
2. # apt-get update
3. # apt upgrade . . (optional)
4. # apt install tasksel . . (for versions less than or equal to 20.04 only)
5. # tasksel [select LAMP server by pressing Spacebar and hit Enter]
6. # nano /var/www/html/info.php
   1. <?php

phpinfo();

?>

Ctrl+x,

press Y,

Press Enter

1. # apt install apache2 |
2. # apt install mysql-server | (A must for 22.04, but can be used for any version)
3. # apt install php |
4. # vi /var/www/html/index.html (vi editer)
   1. 9dd . . (to delete 9 lines)
   2. dd . . (delete selected lines)
   3. D . . (delete selected lines in)
   4. %d . . (delete all lines in the file)
   5. :wq! . . (save, quit, force)
   6. Ctrl+c . . (to exit from the update section)
   7. Ctrl+L . . (Clear screen)
   8. Esc

**Extra Commands for LAMP Installation**

# mysql –version

# ufw allow in “apache”

# ufw status

# apt install php libapache2-mod-php php-mysql

# php version

# sercvice apache2 status

**STRESS ON SERVER**

# sudo -i

# apt-get update

# apt-get install stress . . . installing stress

# stress -c -4 . . . . apply four core cpu load

# ctrl + c . . . . . to remove load

**AUTO SCALING PREREQUISITES**

AMI with LAMP

CLB without instance

SNS

Cloud Watch

Auto Scaling group / rules

**Basic Linux Commands**

1. # cd myfolder . . . (change directory, it will go back a step)
2. # cd /etc/passwd . . (change the directory to the specified path)
3. # cd .. . . . (this will take us one dir up)
4. # cd . . . (this will take us to home dir)
5. # cd / . . . (root dir)
6. # cd - . . . (previous working dir)
7. # ls . . . (list of files and folders in the current directory)
8. # ls path/of/dir . . ( ,, ,, ,, ,, ,, given path)
9. # ls -l . . . (contents of the current dir listed with permissions)
10. # ls -la . . . (all the files & folders are listed with permissions)
11. # mkdir myfolder . . (makes directory named ‘myfolder’)
12. # touch f1 . . . (create a empty file named ‘f1’)
13. # touch f1 f2 f3 . . . (creates requested number of folders)
14. # rm f1 f2 f3 . . . (removes a file/s in the current folder)
15. # rm -i . . (will ask before deleting each file)
16. # rmdir myfolder . . (Removes an empty folder)
17. # rm -r myfolder . . (force deletes even a non-empty folder)
18. # rm -r \* . . . (force deletes everything in the current folder)
19. # rm /path/to/dir/ . . (removes all files(only) in the provided path folder)
20. # rm -rfv /home/vivek/data/ . (removes all files and folders in the path)

**Other rm Commands**

# rm /path/to/dir/\*

# rm -rf /path/to/dir/\* . . (all exept hidden)

# rm -rf /path/to/dir/ . . (all witin dir)

# rm \*

# rm -rf /path/to/dir1/{\*,.\*}

# rm -rfv /path/to/dir1/{\*,.\*} . (all including hidden)

1. # adduser Kris . . (creats a user named kris)
2. # su Kris . . (we can log in to user named Kris)
3. # vi f7 . . (can edit file named f7, if not will create and edit)

i . . . . (for inserting mode)

o . . . . (second lilne)

Esc . . . . (to exit insert mode)

:wq! . . . . (save, quit, force)

1. # visudo . . . . (we can make use as sudoer)
2. # cat f7 . . . . (to view file f7)
3. # cat > f7 . . . (over write f7 or create f7 & edit, enter& ctrl+c)
4. # cat >>f7 . . . (we can add more data)
5. # cat f7>>f8 . . . (will copy f7 to the end of f8)
6. # cat f7 > f8 . . . (copy from f7 to f8)
7. # nano f7 . . . (will create and edit, ctrl+x and Y & enter)
8. # cp f7 /path/to/dir/ . . (copy f7 to the provided path)
9. # mv f7 /path/to/dir/ . . (move f7 to provided path)
10. # fdisk -l . . . (list the attached devices)
11. # mount . . . (list of mounted drives)
12. # mkfs.ext4 /dev/xvdf . . (formats the drive /dev/xvdf)
13. # mount /dev/xvdf myfolder . (mount the drive on myfolders)
14. # umount myfolder . . (unmounts device mounted on myfolder)
15. # less /etc/passwd . . (see content page-wise)

d . . . . (next page)

b . . . . (previous page)

/ . . . . (search for a word in the file)

v . . . . (to go ‘vi’ mode)

q . . . . (quit)

1. # more /etc/passwd . . (scroll line wise)

Enter . . . . (to scroll down line by line)

d . . . . (next page)

b . . . . (previous page)

/ . . . . (search for a word in the file)

v . . . . (to go ‘vi’ mode)

q . . . . (quit)

1. # head /etc/passwd . . (display the top 10 lines)
2. # head -n /etc/passwd . . (display the top n number of lines)
3. # tail /etc/passwd . . (display the last 10 lines)
4. # tail -n /etc/passwd . . (display the last n lines)
5. # sort f7 . . (will sort the files order wise)
6. # sort -u f7 . . (will sort and eliminates duplicates, only output)
7. # sort f7 >> f8 . . (will sort and save the output in f8)
8. # cut -d ‘:’ -fn /file/path . . (will display the desired field in a data file)

Eg: #cut -d ‘:’ -f5 /etc/passwd . (will display the 5ht field in passwd file)

1. # sed ‘s/old-word/new-word/g’ filename (will change the old word with new, only in output)

Eg: #sed ‘s/AWS/DEVOPS/g’ f1 . (will replace AWS with DEVOPS in the file f1, ,, )

1. # sed ‘s/oldword /newword /g’ f1 >> f2 (will save the changes in f2)
2. # grep word /file/path . . (shows only the lines containing the word)
3. # grep -c name /file/path . (sows count of the word in the data)
4. # grep -nA2 word /file/path . (shows till two lines after the word line)
5. Grep -nB2 word /file/path . (shows two lines before the word line)
6. # find . -name filename . (find the filename in the current and sub directories)
7. # find / -name filename . (will find any file in all the locations under the root)
8. # find / -user username . (all files and folders created by the user will display)
9. # find /home \*.jpg . (finds all jpg files under /home and its sub dir)
10. # find . \*.jpg . (finds all jpg files in the current dir and sub dir)
11. # find . -type f -mtime -2 size -10M (f=file,2=days,-10M=lessthan 10MB, “.”=current dir)
12. # find . -type d -mtime -2 size -10M (d=directories,2=days,-10M=lessthan 10MB)
13. # find . -type f -name "\*.php" . (all files ending with .php)
14. # find . -type f -perm 0777 -print (find all the files whose permissions are 777)
15. # find / -type f ! -perm 777 . (find all the files without permission 777)
16. # top . . . (shows cpu utilization, ctrl+z to stop)
17. # free . . . (RAM size, used and free size)
18. # df -h . . . (mount points)
19. # ps . . . (no of running process)
20. # ifconfig . . . (shows he ip address)
21. # who . . . (who uses default user)
22. # who am i . . . (shows how we are loged in)
23. # hostname . . . (ip address)
24. # uname -9 . . . (architecture, user etc.)
25. # chmod 776 f7 . . (change permission of f7 file)
26. # chmod g=rwx f7 . . (group=read, write, execute on file f7)
27. # chmod g=rwx,o=rwx . . (group, others= read, write, execute on file f7)
28. # chmod ugo=rwx f7 . . (user, group, others= read, write on file f7)
29. # umask . . . (current umask value, normally 0002)
30. # umask 022 . . . (umask chages to 0022)

File =777-111=666, 666-002=664(rw-, rw-, r--), dir=777-002=775 (rwx, rwx, r-x)

File =777-111=666, 666-022=644(rw-, r--, r--), dir=777-022=755 (rwx, r-x, r-x)

**MASTER & NODE CONFIGURATION:**

1. $ sudo -i
2. # apt update
3. # adduser kris
4. # visudo . . . (we can change user to sudoer)

Kris ALL=(ALL) NOPASSWD: ALL (enter the user details under “Root”)

1. # vi /etc/ssh/sshd\_config . (ssh config file will open)

Change: PasswordAuthentication **yes** :wq!

$ sudo -i

# su kris

$ cd

$ pwd

1. $ ssh-keygen . . (a public and a private key will be created)
2. $ ssh-copy-id <private ip of nodes> (attaching node and master servers)
3. $ ssh <private ip of nodes> . (we can long in to node with master server)

**GIT**

1. $ git clone <clone url>
2. $ git add -A
3. $ git commit -m “comment”
4. $ git push
5. $ git pull
6. $ git config –global user.email “your@example.com”
7. $ git config –global user.name “your name”

**TREE**

1. $ sudo apt install tre3e
2. $ tree mywebrole

**ANSIBLE**

STAGE I: Installing and Configuring ANSIBLE in the MASTER SERVER

1. $sudo -i
2. # apt update
3. # adduser kris
4. # visudo . . (Change the user to sudoer)

Kirs ALL=(ALL) NOPASSWD: ALL (Enter the user details under “Root”)

Ctrl+x, Press y and Press Enter

1. #vi /etc/ssh/sshd\_config

Change: PasswordAuthentication to **yes**

Press **Esc** and enter **:wq!**

1. #service ssh restart
2. #apt update
3. #apt install ansible

STAGE II: installing and Configuring ANSIBLE in the NODE SERVER

1. $sudo -i
2. #apt update
3. #adduser kris
4. #visudo . . (Change the user to sudoer)

Kirs ALL=(ALL) NOPASSWD: ALL (Enter the user details under “Root”)

**Ctrl+x**, Press **y** and Press **Enter**

1. #vi /etc/ssh/sshd\_config

Change: PasswordAuthentication to **yes**

Press **Esc** and enter **:wq!**

1. #service ssh restart

STAGE III: Connect Master server and Node server (from Master server)

1. #su kris
2. $cd
3. $pwd
4. $ssh-keygen . . (a public and a private key will be created)
5. $ssh-copy-id <private ip of nodes> (attaching node and master servers)
6. $ssh <private ip of nodes> . (we can long in to node with master server)
7. $exit
8. Vi myhosts

Create a file with the Private IP address of the nodes -> :wq!

1. $ansible all -i myhosts -m ping

On executing this cmnd if green coloured text appear then the installation of ansible and the connection b/w the node and master is good

**ANSIBLE AD-HOC COMMANDS**

1. $ansible all -i myhosts -m ping

We can check if the hosts are working

1. $ansible all -i myhosts -m setup

We can see the details of all the hosts

1. $ansible all -I myhosts -m setup -a filter=”\*family\*”

We can filter the hosts details in this case it will filter and display only the family of all the hosts

out put: ansible\_os\_family: ”RdHat”

out put: ansible\_os\_family: ”Debin”

1. $ ansible mywebserver -i myhosts -m ping

Only **mywebserver** group will respond

**ANSIBLE PLAYBOOK COMMANDS**

1. $ ansible-galaxy init mywebrole
2. $ansible-playbook -i myhosts mywebservers.yml

This will execute the playbook ”mywebservers.yml” on the all hosts listed in the “myhosts” dox file

1. $ ansible-playbook -i mywebserveres.yml

This will execute the playbook ”mywebservers.yml” on the all hosts listed in the default file in the ansible engine

1. $ ansible-playbook -i myhosts mytags.yml --tags=”mytree”
2. $ ansible-playbook -i myhosts mytags.yml – skip-tags=”mytree”

**DOCKER COMMANDS**

|# curl -fsSL https://get.docker.com -o get-docker.sh |

| Downloads docker setup |

|# sh get-docker.sh |

|installs docker |

1. # docker –version
2. # service docker status

DOCKER RUN Creats a container

# docker run --name myc1 -d -p 8081:80 nginx

# docker run --name myc2 -d -p 8082:8080 tomcat:8.5.37-jre8

# docker run --name myc3 -d -p 8083:8080 jenkins:2.60.3

# docker run --name myc4 --restart always -d -p 8084:80 nginx

# docker run –name myc5 -v /myvolume -d -p 8085:8080 jenkins:2.60.3:1

# docker run –name myc6 -v /home/ubuntu/mydocker:/myvolume -d -p 8085:8080 <image id>:1

# docker run -d -p 9000:9000 --restart always -v /var/run/docker.sock:/var/run/docker.sock -v /opt/portainer:/data portainer/portainer

[This is to interact with docker Graphically]

CREATING CONTINER

# docker search ­­­nginx Searches all images for niginx and a list of images will be displayed

# docker pull nginix . . . Downlodes nginx Image

# docker run -d nginx . . . Creates a container on top the downloaded image

LIST OF CONTAINERS & IMAGES

# docker ps . . . . Lists running containers

# docker ps -a . . Lists all containers both running and stopped containers

# docker ps -a -q . . . Lists all container IDs

# docker images . . . Lists all images

# docker images -q . . . Lists only the IDs of images

# docker inspect <container id/ name> More details of the container

DELETE OR STOPCONTAINERS AND IMAGES

# docker rm <container id/name> -f . Removes particular docker container only

Eg:- # docker rm myc1 -f

# docker rm $(docker ps -a -q) -f . Removes all docker containers both running and stopped

# docker rmi <image id/name> . . Removes particular image only

Eg:-# docker rmi nginix:latest

Eg:-# docker rmi tomcat:8.5.37-jre8

# docker rmi $(docker images -q) . . Deletes all docker images

# docker stop <container id/ name> . . Stop a running container

# docker start <container id/ name> . . Starts a container

DOCKER MODES

Attach: without -d . . goes to stop state after closing terminal

Detach: with -d . . runs even after closing terminal

Interactive: with -it . . getting into the container

Eg: # docker exec -it <container id/ name> /bin/bash

We can get into container (we can see the change of ip to container id in the curser)

DOCKER VOLUMES

# cd /var/lib/docker

We can see the docker data

# cd /var/lib/docker/containers

# cd /var/lib/docker/image/overlay2/imagedb/content/sha256 # ls sha256 code is displayed

# cd /var/lib/docker/volumes

we can see content of the mounted volume from out-side container

# cd /myvolume content of the volume from inside the container

# df -h can see list of mounted devices once inside the container

# cd /var/lib/docker/volumes/<HashCode of volume>/\_data (we can see the content of the volume)

# docker run –name myc5 -v /myvolume -d -p 8085:8080 jenkins:2.60.3:1

# docker run –name myc6 -v /home/ubuntu/mydocker:/myvolume -d -p 8085:8080

$ sudo deluser newuser . . delete only user not their data

$ sudo deluser --remove-home newuser delete user and also data

DOCKER NETWORK

# docker network create -d bridge --subnet 192.168.1.0/24 krisnetwork

# docker network ls

# docker run --name myc2 --network krisnetwork -d -p 8082:8080 <image id>:1

# ping <container name> (if containers are same private network)

Ex. ping myc1

# ping <container ip> (if containers are in same network)

Ex. ping 192.168.1.3

DOCKER SWAP & CGROUP

# vi /etc/default/grub 🡪 GRUB\_CMDLINE\_LINUX="cgroup\_enable=memory swapaccount=1"

# sudo update-grub

# sudo reboot

# docker run --name myc1 -d -m 300M kr15hna/myjavfileimage01sep2022:1

RAM = 300MB, swap = 600MB (swap default)

# docker stats

# docker run --name myc2 -d -m 300M --memory-swap -1 <image id>:1

We don’t want swap memory

# docker run --name myc1 -d -m 300M --memory-swap 1G <image id>:1

RAM = 300MB, swap = 1GB

# docker run --name myc1 -d -m 300M --memory-reservation 200M <image id>:1

RAM = 300M, Reserved RAM = 200M, Swap Memory = 600M (default)

# docker run --name myc1 -d --cpus=0.5 -m 300M --memory-reservation 200M <image id>:1

[CPU = AM = 300M, Reserve RAM = 200M and Swap Memory = 600M (default)]

**JAVA**

# sudo apt-get install openjdk-8-jre

# yum install java-1.8.0-openjdk

# sudo apt remove --autoremove openjdk-8-\* . . unistall java8

# sudo apt purge openjdk-11-\* . . . unistall java11

# sudo apt remove --autoremove openjdk-11-\* . . unistall java11

**KUBERNETESE COMMANDS**

KUBERNETES PODS

# kubectl get nodes . . [All the nodes connected to kube master]

# kubectl create -f mypods.yml [Creating pods with yaml file]

# kuectl get pods . . [Displyes all created pods]

# kubectl get pods/<name of pod> [Finds specific pods]

# kubectl get pods -o wide [More details about pods]

# kubectl describe pods [All the details of the pods]

# kubectl get pods --all-namespaces [All the pods created to maintain cluster are also displayed]

# kubectl delete pod/<name of pod> [Delete a particular pod]

KUBERNETES REPLICATION CONTROLLER

# kubectl create -f myrc.yml [Creates Replication Controller]

# kubectl apply -f myrc.yml [Updates Replication Controller]

# kubectl get rc [Get all the Replication Controller]

# kubectl delete rc/mynginx-rc [Delete particular RC]

KUBERNETES SERVICE

# kubectl create -f svc.yml

# kubectl apply -f svc.yml

# kubectl get svc

# kubectl describe svc hello-svc

Endpoints

# kubectl get ep

# kubectl describe ep hello-svc

**APPS AND SERVICES USED**

*Get aws account*: <https://aws.amazon.com>

*Downlode git-bash at*: <https://git-scm.com/download/win>

*Get git hub account*: <https://github.com> provide mail and name before “git commit” & use git token before “git push”