6/09/21

CI :- → Jenkins used by devlopers (writing code, Built, Testing)

CM (configuration Management) :- It is infrastructure as code.

↳ chef / puppet / Ansible / terraform

↓ taken care by system administrator

↓ servers, networks etc

we add servers or delete or any other work by DSL (Domain Specified language).

continous Delivery & Continous Deployment :- → Docker These two are pratices used to automate Deployement.

used in Dev Environment
QA "
staging "
production " (pre prod?)

continous Delivery :-

The capability of Deployement at any time is called continous Delivery

In this we do upto only staging.

continous Deployement :- we will do prod" i.e automating upto prod".

Build :- Maven

[Docker :- used for containirization
↓ by using this we can run anywhere]

orchestration :- → Kubernetes
To make sure containers are working.
(or)
To manage containers.

curriculum

→ linux

→ Git (source code management)

→ Jenkins → Artifactory, tomcat, sonarcube, Jenkins pipelines

→ CM (Ansible)

A/f b/w Ansible & Terraform

→ Containization

→ Docker & Kubernetes

→ Build tool

→ maven

→ Monitoring

→ Nagios

→ log

→ EbK

→ cloud

→ 3 projects

→ Roles & Responsibilities

LINUX

→ Most of apps running on linux (70%).

→ Linux is os

why os is used?

It is interface b/w user & hardware.

Ex:- windows, linux, ubuntu

→ two types of interfaces → GUI → windows
→ CLI → linux & Unix.

→ linux is installed on servers.

windows → client → windows 7, 8, 8.1, 10, 11.
→ servers
Ex: 2008, 2008R2, 2012.

→ linux os → linux / unix
→ Redhat
→ centos
→ suse
→ Ubuntu.

→ linux is not user friendly when compared because of CLI.

why organisations prefer linux/unix?

① For security
↓
linux is more secure
↓
passwords & imp files are converted to encrypted form

② No Virus

③ Very less hacking

④ Now down time

→ In windows → Ransomware

↓

Virus

Virus → It is a .exe file.

↓

This virus will ~~connect~~ corrupt system files.

↓

C:/windows/system32.

→ In linux we use shell it doesn't recognise any .exe programme.

→ No downtime in linux.

In windows every 90 days restart is required.

1962 AT&T → Introduced

1969 → Devloped MULTIX

1993 → UNIX (Uniplex Interface computing system)

Kernel:-

It is module having

Unix flavours

Sun — Solaris

IBM — AIX

HP — HP-UX

Silicon graphy — IRIX

Santa Cruz — SCO-UX

1992 → Torwald Linuz

↓

he cloned unix and named as linux.

Linux & Unix file structure

→ In linux everything under "/"

→ under this we have folders

→ /bin → Binary directory
* All executables are stored
→ It has normal user commands
Ex:- ls, cat, rm.

→ /sbin

→ /usr

→ /etc

→ /opt

→ /var

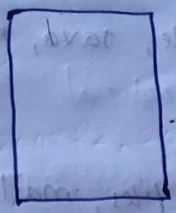
→ /media

→ /dev

→ /temp

→ /boot

→ /mnt



In linux two user logins → normal user logins
→ root user logins

/sbin (system binary directory).

→ All admin commands stored under sbin.

Ex:- useradd, mount, shutdown

/usr :- manual pages are stored

Ex:- help

Man command

→ we get discription, options

→ /etc :- configuration files

All configuration files are stored under etc directiory.

Ex:- ~~test.sh~~ software installations files are stored

→ /opt :- optional

Third party software files are stored

Ex:- oracle, java, maven.

→ /var :- log files, mail, messages are stored

It will capture infor of system every second

It is used for boot cause.

why app cannot be accessible we can see on log files

→ /media :- Any removable devices can be accessed

Ex:- cd's, dvd's, pendrives

→ /dev :- logical devices information

Ex:- c drive, d drive etc.

→ /tmp :- to download, to share

any downloads from internet are stored

→ /boot :- Kernel files are stored

→ /mnt :- temporary mount point.

→ /home :- If we create user they will be under this directory.

Ex :-
① /home/user1
② /home/user2

08/09/21

How to create a files

files → regular files Ex :- Text files & directories
→ special files Ex :- Any devices, videos

For file creation we use three commands

① cat
② touch
③ vi

① ~~cat > file name~~ cat > filename
(spr)

now spr file is created.

Ex :- cat > sample.txt

→ To save & ~~quit~~ quit → ctrl + d

→ To view the file cat filename

Ex :- cat sample.txt

for Java .java
for python .py
} extensions after file name.

→ To append the file → cat >> filename

↓

add new lines at the End of file.

Ex:- cat >> sample.txt

→ we cannot modify the file using cat command.

② Touch :-

It is for creating Empty files. we can create n number of files.

touch filename.

Ex:- touch sample1 sample2 ...

③ Vi :- (Visual Editor)

It is a Editor. we can create, modify the file.

Editors → Windows :- notepad

Linux :- vi, Emac, nano

syntax:- vi filename

Ex:-

vi abc

abc file is created

To write text → Esc + i (To insert text).

To save → Esc + : + W + q + !

↓ save ↓ quit ↘ force quit when other users also using.

How to create a directory (folders)

① mkdir directoryname.

Ex:- mkdir demo

- ls → to check cohether directory is created or not.

Black → files
Blue → directory
red → special files (zip files, rpm, etc)
green → ~~zip files~~.
Executable or recognized data file.

mkdir -p DevOps/linux/Basics/example.
↳ recursive ↳ It is not home directory, it is next directory
In this folders are created recursively i.e one folder under the other.

tree directory :-
To show what under directory.

cd devops
ls → linux

cd linux
ls → Basics

cd Basics
ls → example

cd example.

cd → change directory.

→ To come one directory back.

syntax:-

cd..

→ To go at a time two directory back

syntax:-

cd../..

→ To move to home directory

syntax:-

cd

How to remove a file.

① syntax:- rm filename.

Ex:-

rm abc

abc file is removed.

② rm -f filename:-

without asking confirmation we are deleting a file

③ rmdir :-

To remove an empty directory.

Ex:-

rmdir demo

④ rmdir -f dir

rm -rf directory name:-

recursive force

To remove non empty direct

⑤ ls :- To list content in a directory in alphabetical order.

⑥ ls -s
↓
size

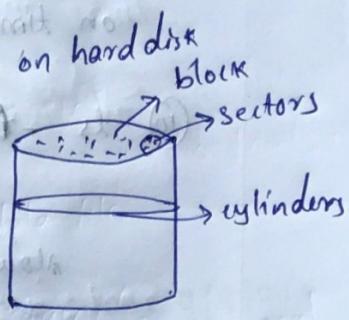
To list content along with size.

generally size is in block

1block = 2mb

Linux kernal represent size in

① block → a point surface area on hard disk
② cylinders
③ sectors



⑦ ls -a :-
To list the hidden files also

Before name it is with .sxe it is called

hidden files & directory.

To create a hidden file :-
Before name we give '.' symbol.

Ex :- .abc

abc hidden file is created.

⑧ ls -l :- long list of the output (full information).

permissions → — rw — r — — r — — 1 → no of links testuser devops 0
↓ owner ↓ group ↓ size
↓
Types

creation name.
Jan 2 2018 abc.txt

three types of are shown at first

- Text files

d → directory

l → link

(9) ls -lt
↘ time

Now newly created files are displayed first based on time

(10) ls -ltr
↘ reverse

Now to display files which are old first.

(11) ls -ld filename/directoryname;
To view particular file or directory information.

To change the file permissions.

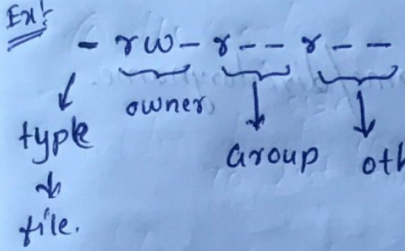
Three types of permissions

(1) read → 4
(2) write → 2
(3) execute → 1
↘ To execute a script.

Three types of users

(1) owner → who creates a file → u
(2) Group → g
(3) others → o
(4) all → a
↘ all permissions can be changed ata time

Ex:-



To change to rwx r-x r-x

permissions can be changed in two types

① Numbers (r w x)
4 2 1

② symbols (u g o).

ls -la → to view hidden files.

9/9/21

If we create file or directory it create with default permissions

for file → 664

for directory → 775

chmod :- used to change the permission

Ex:- chmod 654 sample.txt

syntax:- chmod permissions filename

Numbers permission:-

syntax:-

~~chmod~~ ~~chmod~~

chmod u=rwx,g=rwx,o=rwx filename.

syntax:-

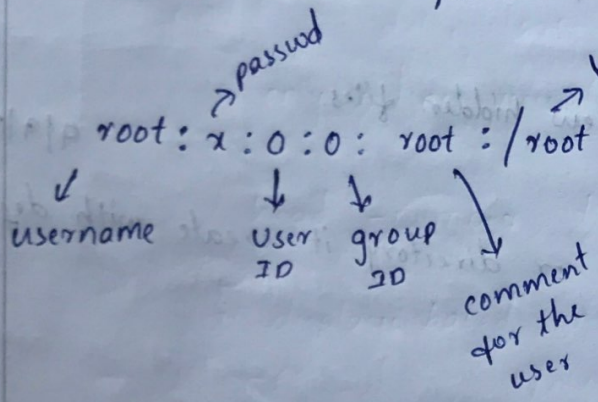
chmod a=rwx filename/directoryname

→ we can change owner also

cat /etc/passwd :-

To know how many users are there in linux system.

In windows → control panel → user Administration.



→ To create a new user

syntax:-

useradd username

Ex:-

useradd Kelly

→ To change owner

chown :-

syntax:-

chown user1 user2 chown newowner name file/ directory name

before user newuser

Ex:-

chown demo demodir.

new owner name file name

we can view at ls -ld demodir

userdel username

↓

It removes only user not userfiles.

userdel -r username

↓

To remove user & remove all files related to user

To change group name.

cat /etc/group:- To check existing groups information.

To create new group

syntax:
groupadd groupname.

Ex:- groupadd pradeep

To change group

syntax:
chgrp group1 group2
↓ old group ↓ New group

Ex:-
chgrp root pradeep

↓
we can check at ls -ld abc.

Filter commands:

↓
we have → simple filter commands
→ advanced filter commands

simple filter commands:- → filter commands are used only for display not for modifying file.

→ head
→ tail
→ wc
→ cutt
→ diff
→ sdiff
→ cmp
→ uniq
→ tr
→ sort
→ more
→ less

→ Advanced filter commands

→ grep
→ sed
→ find
→ awk

① head :-
To display first or starting 10 lines of files

Syntax :-
head filename
↓
only 10 lines viewed

head -number filename
↓
It displays based on number

head -20 abc
↓
It displays 20 lines of file abc

② tail :-
To view last 10 lines of file.

syntax :-
→ tail filename

→ tail -number filename
↓
displayed based on the number.

Ex :-
tail -10 abc.

tail is used see log files

→ tail -f /var/log/messages

↓

to view continous of adding lines ~~for~~ log files

→ ctrl+c to skip viewing log files

② wc :-

It is word count of lines, words, characters

wc /etc/passwd

↓

It shows lines, words, characters in /etc/passwd

syntax:-

wc ~~ab~~ filename

↓

wc -l " → to display line

wc -w " → to display word

wc -c " → to display character

④ cut :-

To cut and display characters (or) blocks (or) block of characters (or) ~~fi~~ fields.

syntax:-

cut -c5 /etc/passwd

↙ ↘

character 5th position character

→ cut -c5-8 /etc/passwd

↙ character ↘ group of character from 5th to 8th position displayed.

To cut the fields

→ cut -f1 -d ":" /etc/passwd

↙ field 1 ↓ delimitors

↓

Generally it is space, pull stor, comma

↓

In our case it is ':' (colun)

→ cut -f1 -d ":" /etc/passwd | wc -l

↓ word count ↘ It counts lines

↓

Now we can count no of users.

→ cut -f1,3 -d ":" /etc/passwd

↓

fields 1 & 3 are viewed.

→ To

⑤ diff :- To know diffrent lines in two files.

syntax:

diff file1 file 2

Now file1, file2 are compared and display diffrent lines in two files.

⑥ sdiff :-

To display the two files side by side.

syntax :-

sdiff file1 file2.

⑦ cmp :-

compare the two files character by character

syntax :-

cmp files file2

↓

It will stop the comparision when they will diffrent & show the o/p

↓

If same it dosent show o/p

⑧ uniq :-

It display the unique lines in the file

syntax :-

uniq filename

uniq -d filename

↓

To view what are lines duplicated.

uniq -D filename

↓

how many times lines are duplicated.

(9) tr :-
It will translate characters

syntax :-
tr '[a-z]' '[A-Z]' < filename
↓
already existing file we take as input

Ex :-
tr '[a-z]' '[A-Z]' <abc

>
↓
overwrite

(10) sort :-
syntax :-
⇒ sort filename
↓
It sort content in the file in alphabetic order.

13/09/21

Advanced filter commands :-

(11) more & less :- Both are used to scroll a file

Ex :-
cat etc/ssh/sshd_config
less file name

b → back page
f → front page
space → page by page scroll
Enter → line by line scroll
q → quit.

more → when we use more command it automatically quits after viewing the file.

Advanced filter commands :-

① grep :- global regular expression.

grep is used to search word or string or pattern.

syntax :-

grep linux.txt abc → They display string or patters
↓ word

grep 'linux'.txt abc
↓ word ↓ filename

grep -w 'linux' abc → It display words

grep -i word filename → It ignore case sensitive. either capital or small letters.

grep -n word filename → It displays along with which lines have those words.

grep -c word filename → It count how many lines have that word.

grep -b word filename → It ~~displays~~ exclude particular words from displaying in file.

grep -Rw Word → In current directory from all
↓
Recursively the files it display which every line have that word.

grep 'word'$ filename → line ending with particular word

grep ^'word' filename → To display line starting with particular word.

How to sort two words at a time.

sed :- stream Editor

To open (see) a particular line

① sed -n '5p' filename → onle line
(5 → line numbers, p → print)

② sed -n '5,8p' filename → group of lines
(5-8 lines.)

③ To display selected lines → multiple selected lines

sed -n '5p
8p' filename] Enter

It display 5th & 8th line.

④ sed '5d' filename → To not to display 5th line

(delete → only deleted from displaying)

Exclude single line

⑤ sed '5,10d' filename → not display 5th to 10th lines

Exclude range of line

⑥ sed '5d
>10d' filename] enter → not display 5th & 10th lines.

⑦ to replace certain word while displaying

(substitute)

sed 's/first/last/g' filename

old word → to

New word

global → If same word repeate multiple time in a single file if we want to replace we use that word.

find :-

To search the files

windows → search bar

syntax :-

① find /location -name filename

Ex :- find / -name sample.

↓ It ~~sample~~ search in whole system

find /root -name sample

↓ It search in root folder

(2) find /root -name "*.xml" (or) .txt (or) .csb (or) .sh

↓ any

Ex:- find /home/spr -name *.txt

(3) find /root -size 1G

↓

search with size here more than 1 GB

find /root -size 100M

" " " +100M → more than 100MB

" " " -100M → less than 100MB.

(6) find /root -perm 666

↓

It displays files having with 666 permission.

(7) find /root -type f

↓ ↓ files

It displays all text files.

(8) find /root -type d → to display directories

(9) find /root -type l → to display linked files.

(10) find /root -ctime +4 → exactly 4 days ago created file.

↓ created time (or) change time

0 → for today created file.

give read, write & executable permissions before this tasks.

(11) find /root -mtime 4
↓
modified time

(12) find /root -atime 4
↓
access

(13) find /root -cmin 4 → file created exactly 4 min ago

(14) find /root -mmin 4

(15) find /root -amin 4

awk :-

To display the fields

→ cut is also used for name

(1) awk -F':' '{print $1}' </etc/passwd
print upto first field
↓ field seperater
↘ delimiter
↓ In /etc/passwd file.

(2) awk -F':' '{print $1, $3}' </etc/passwd

To display first & 3rd fields

(3) awk -F':' '{print $0}' </etc/passwd

To display all the fields.

14/09/21

RPM (Redhat package manager)

→ In linux package → software.

→ package will be in ~~.rpm~~ .rpm format

→ zip files end with .zip

Ex:

chef - 16.1.0 - 1.el6 . x86_64 . rpm

- chef → package name
- 16.1.0 → version
- 1.el6 → Release
- x86_64 → architecture

chef-16.1.0-1.el6.x86_64.rpm

To install → first download from internet

↓

To download command

↓

Wget <url>

~~To install after down~~

To verify whether package downloaded

↓

rpm -q packagename (first name)

↓

query

Ex: rpm -q chef

To remove the package → copy full package

↓

rpm -e packagename (full name)

- -e → erase
- we cannot see .rpm extension here.

To verify whether package removed or not

rpm -q package (first name)

Now download from internet

→ Now to install package

↓

rpm -ivh package name.

Ex:- rpm -ivh chef-16.1.0-1.el6.x86_64.rpm

i → install

v → verbose

h → hash progress (how much % of installation).

→ Now verify package

rpm -q package name (first name)

→ To know package full information

rpm -qi package fullname.

→ package → collection of files

rpm -ql package fullname

l → list

It lists all the files of the package.

→ rpm -qR package full name

R → Resolvers

↓

To install this packages what are prerequisites it show.

→ to upgrade the package without uninstalling

rpm -Uvh package full name

→ rpm is offline package method.

↓

we install dependency package first manually

↓

then main package later manually.

yum :-

It is online package manager and resolve the dependencies automatically.

To install package :-

yum install packagename

It automatically install package

To remove package

yum remove packagename

To update package

yum update packagename.

Ex:-

yum install httpd

↘ ask for confirmation

↓

Y

In yum → ls /etc/yum.repos.d/

↓

In this repository we have package url's

↓ from here packages installed

Linux flavours → Redhat/centos

↓

we use rpm

in ubuntu/debian → package extension → .deb

↓ offline package

to install → dpkg -i <pkg name>

to query → dpkg -l <pkg name>

to remove → dpkg -r (or) purge <pkg name>

pkg

→ online package manager

to install → apt-get install pkg

to check status of service

↓

service pkgname ~~service~~ status

Ex:- service httpd status

to start service

↓

service pkgname start

Ex:- service httpd start

to stop service

↓

service pkgname stop.

→ In Redhat 7/centos 7 service command is replaced by systemctl

syntax: systemctl start pkgname.service.

→ To show list of installed services in our system

chkconfig --list

when we start a system d/f states (or) system run levels

0 → shut down

1 → single user mode

2 → multi user mode without n/w

3 → " " (command line)

4 → Reserved or unused

5 → multiuser mode (GUI)

6 → Restart

To know in which state our system is booted

who -r

Generally we use 3 & 5

To change run level

syntax: init runlevel number.

Ex:- init 5

↓

system will boot into graphical.

→ ps → To list the running process on our terminal

→ ps -ef

↓ ↘

every file

↓

It list all the running process on our system.

UID	PID	PPID	C	STIME	TTY	TIME	CMD
↓	↓	↓	↓	↓	↓		↓
UserID	process ID	parent process ID	priority	when process is started	terminal		process relevant command

→ To search particular service related process

ps -ef | grep servicename

Ex:- ps -ef | grep Jenkins

If any Linux related service not accessible (even running)

↓

Kill all the process

↓

command

↓

Kill -9 3392 → To kill process with ID.

syntax:- Kill -9 PID

↓ processID

pkill -9 (→ kill signal) pkgname

↓

To kill process with process name.

To run the process in background

Ex:-

cp fileA fileB &

↓ ambersent symbol.

To get the ipaddress of system.

↓

Ineonfig

we get

Stho → NIC (Network Interface card)

↓

IP address

lo → loop back → This is system default address.

To know system ~~version~~ processor

Uname -m

To know Kernel version

Uname -r

To know architecture → full information of a system.

Uname -a

To know which os we are using

cat /etc/redhad-release

(or)

cat /etc/issue.

20/09/21

How to create Ec2 Instances ?

offline → VM → centos/linux

online → Ec2 instance

AWS → It provide infrastructure as a service (IAAs) → servers

& platform as a service (PAAs)

E commerce → server → Data center (server, network) → administrator required

↳ AWS → without any server

AWS → GUI → console access

→ CLI, SDK, API → proggrammable.

AWS → North verginia → default region

services → EC2 (Elastic cloud compute) → Instances (virtual machine or servers) → launch instance

Azure, GCP (NM)

↓

7 steps

① choose AMI :-

↳ choose any one template

AMI (Amazon machine images)

② choose instance type

↓ choose configurations

AWS charge → based on hours

→ not number of instances.

③ configure instances :-

server settings

n/w settings.

④ Add storage :-

AWS offers two types of discs (volumes

Hard disk SSD

min volume → 1GB

max volume → 16TB

→ we can increase volumes

→ 30GB storage → free of cost

↓ when we launch more size

↓ charged according to 1GB.

⑤ Add tag :-

project → demo

purpose

→ easily identifying

→ " Billing

⑥ configure security group:- (fire walls)

RDP ?

SSH ? → port number → 22

cat /etc/services

↓

All available services & related port numbers

Telnet → 23

FTP → 21

MYSQL → 3306

RDP → 3389

HTTP → 80

HTTPS → 443

Jenkins → 8080

source:- from where you want to allow

Anywhere

custom n/w

My IP

Security group name:- any name

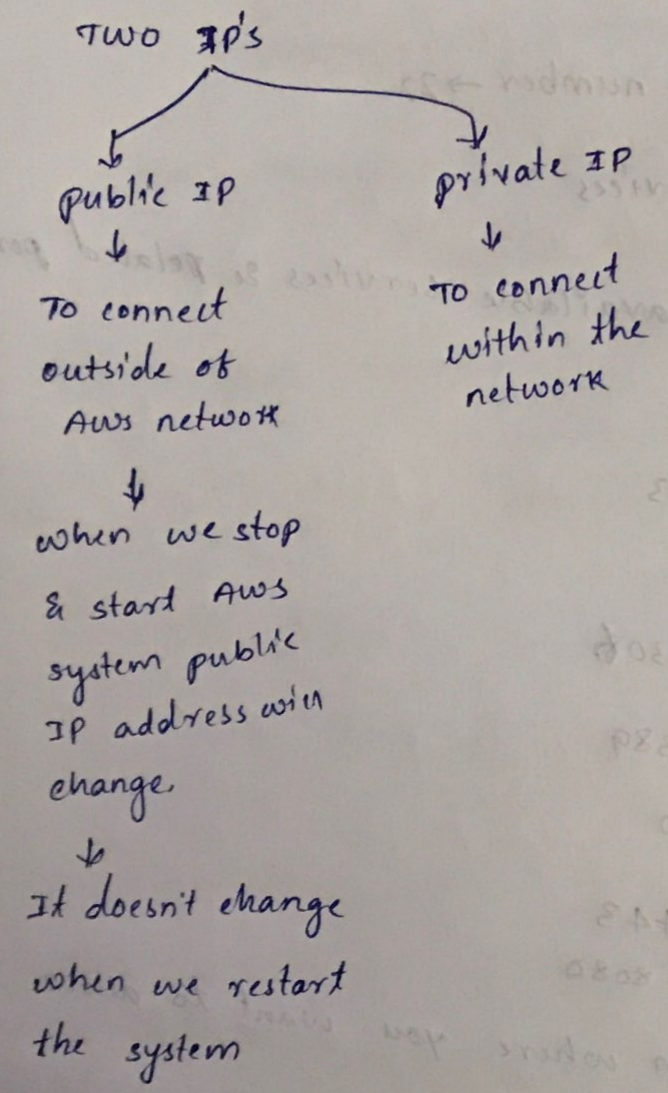
Ex:- LINUX-SG

Description:

Ex:- for linux server.

⑦ Launch.

→ Give key pair & now launch the instance.



windows → putty

linux → mac/linux → ssh

Public IP
↓
To connect outside of Aws network
↓
when we stop & start Aws system public IP address will change.
↓
public IP doesn't change when we restart the system

↓
To connect within the network

21/09/21

windows → putty

linux ↔ mac/linux → ssh -i <key> <username> @ip
↑ Aws

→ where to check free services
↓
Aws free tier

To connect from linux or mac:
open terminal
↓
ssh -i <key> <username> @ip
↓
aws private key

Ex 1

① ssh -i ~/Downloads/demo_us_west.pem ec2-user@54.191.210.107

- ~ → home directory
- ec2-user → default user
- 54.191.210.107 → public IP address

any Aws system default user name is ec2-user

for ubuntu system user name is ubuntu

② give yes

③ to go root user → sudo -i

④ Key permission should be changed first time

chmod 600 ~/Downloads/demo_us_west.pem

chmod (or) 400 " " "

⑤ exit

↓

Logout from command line.

TO connect from windows putty :- we use putty :-

→ In putty pem key should be converted into ppk.

→ putty gen is tool is used to convert pem key into ppk

→ make sure where key is located

→ convert pem into ppk.

putty :- used to connect linux machine.

Browser → puttydownload
↓
putty.org
↓
download .msi file.
↓
install putty

windows → all programs → putty

puttygen
↘ load private key → select file → open
(pem file)
↓
~~give~~ save private key
↓
give ~~same~~ name
(PPK file)

→ username :- ec2-user

→ to go to root user :- sudo -i

GIT

→ Git is a version control system

All working → workstations (PC's) → how can be known what are changes made.

↓

Generally known after all changes made

↓

duplicates may happen ← changes may not integrated ← for the collaboration among team members we use version control system

↓

→ It doesn't allow overwrite each other's changes.

→ It give version number ↔ how many line modified
↔ who did the change

↓

we can revert code using version number.

↓

If new changes are not good we can go back.

→ since we are backing up code ~~source~~ code is not lost.

Version control system:- It is a software that help software developers to work together and maintain complete history of their work.

Ex:- GIT, SVN, purfose, clearcase, mercural

Types of VCS:-

(1) centralised version control system (CVCS) Ex:- SVN

(2) Distributed/ Decentralised version control system (DVCS). Ex:- GIT

CVCS :-

In this server will act as a single point of failure.

Do changes in local system → commit in server

DVCS :-

here changes can be commited into local repository. there will be copy with every local repository.

repository :- group of files.

setup your username & email ID.

git config --global user.name "yourname"

git config --global user.email "mail id"

~~your editors~~

~~git config~~

git log :-

It shows the commit description.

.git :-

In this we have default files

config, Head, description, info, index, logs.

cat .git/config :-

config is a file where we can do the repository settings.

cat .git/description :-

description is a file to set the repository name.

ls .git/hooks :-

hooks is a folder where we can Keep automated scripts

Ex :- pre commit, post commit.

cat .git/Index :-

→ staging data is stored here.

→ we cannot read only kernel will read.

ls .git/info :- In info folder we can exclude the file.

it will update excluded files from commit.

*.[abc.txt]

↳ It will exculde abc.txt from commiting if we update it in info folder.

ls .git/log/Head :- It will store the commits information.

ls .git/objects :- any changes in file will store as two digit directory in object folder and cannot be read by user.

ls .git/Head (or) ls .git/refs :- latest commit information will store under this.

To know particular commit related information

git show commitid

↳ can be seen at git log.

green → added new line

Red → changed (or) modified.

git log --oneline :-
It shows first 7 characters of commit id and description.

git add . :-
↓ current directory
It will add changes made in current directory to staging area.

→ If we add 10 files & if we want to commit 5 files it is not possible.

→ If we added 5 files we need to commit 5 files.

→ If we unfortunately deleted (or) modified some lines we can restore changes from git

↓

git restore filename

Ex:- git restore sample.txt

do changes - revert # git restore file name

→ If we di after adding revert → git restore --staged filename
git restore filename.

commit revert → git ~~reset~~ ~~--commit~~
git reset --hard ~~Head~~ HEAD~

→ If we did changes and added to staging area.
now how can we revert changes.

git restore --staged filename

git restore filename

→ If we did commit and want to revert commited files.

git reset --hard ~~Head~~ HEAD

→ In git repository by default we have a branch called master.

Branch is nothing but copy of code. we do changes in copied branch instead of main branch.

git branch :- It list all the branches in git repository.

git branch branchname :-

Ex :- git branch dev :- It create a new branch.

when ever we create a new branch it copies objects(files) & commits.

git checkout branchname :- To switch to ~~dev~~ new branch

git checkout dev :- switched to dev branch.

git merge branchname:- To merge changes from one branch into another branch.

Ex:- git merge master

↓ files merged to master from master.

git merge sourcemerge targetmerge:- If we want specifically merge to some branch.

git merge dev QA:-

files in dev merged to QA.

If we edited same file in two branches

↓

switched from one branche to other

↓

and tried to merge from previous.

↓

It shows conflict occurred.

↓

In which lines we see greater & less than symbols there conflict occured

↓

now open conflicted file by vi filename

↓

fix conflicts

↓

added & commit the file now.

In single command if we want to create a branch and switch to that branch.

git checkout -b name.

In single command if we want to create a branch and switch to that branch.

git checkout -b name.

24/09/21

→ To delete git branch

git branch -d branchname

Ex:- git branch -d Kelly

→ To maintain code centrally we use remote repositories.

→ git diff filename

↓

It will compare with previous git repository with current state file.

and show the changes

→ git diff :-

It shows all the files diffrences before modified and after we modified.

GitHub :-

centrally to maintain code we use github (or) gitlab.

creating git repository

create account

① Repository name :- give name what ever files you want to store.

② Description :- give purpose.

③ Two types of repositories < public / private

④ create repository

⑤ we can access repository by < https / ssh

git remote -v :-

This list any added remote repositories.

copy ~~from~~ from github

git remote add origin git@github.com:kellydevops/Demo34.g

origin → default name for our remote repositories

git remote -v :- It shows added remote repositories.

git branch -M main :-

It is used to rename master branch as main branch.

git push -u origin main :-

To push the code from main branch to origin.

whenever we use https it asks username and password.

Now they are asking token

to generate token → remote repository → settings → Developer settings → personal access token → Generate personal access token

(profile)

→ for any public repository it asks password & username.

→ when we use ssh it doesn't ask for password.

→ how to login from system a to b without asking password

↓

generate the ssh keys

↓

two types

↓

id_rsa.pub

↓

public key

↓

for login of another persons.

id_rsa

↓

private key

↓

should not share to anyone

~~ssh-keygen -t rsa~~

~~In linux~~

system a

ssh-keygen -t rsa

↓

In linux system there is file called ~/.ssh

↓

here we can find keys

↓

id_rsa

id_rsa.pub

system b: In home directory

~/.ssh

↓

there is file called authorized keys

↓

we have to update keys from system a.

open Git bash

↓

ssh-keygen.exe -t rsa

↓

It will generate keys.

ls ~/.ssh/

↓

we can see keys

id_rsa id_rsa.pub

ls ~/.ssh/id_rsa.pub

↓

open & copy the key.

goto Github → settings → gpg & ssh → New ssh key

↓

paste the copied public key

↓

add ssh key

↙

now it doesn't ask for password.

git log --oneline | wc -l :-

to list no of commits.

in git hub → commits into

↓

+ & green → added lines

- & red → removed lines

In github to commit

↓

select file → edit → commit

→ To get the code from github to local repository

git pull <uri>

git stash :-

It is used to save current working directory temporarily we use git stash.

If we want to leave one branch temporarily

↓

we will not add & commit

↓

If we use git stash it save temporarily and we can resume later.

git stash list :-

It display any existing stash in our repository.

git stash :-

It will save temporarily

we cannot see changes at cat filename, git status.

we can only see at git stash list.

→ git stash apply :-

now we can see changes with ^we have done before at cat filename, git status.

git stash apply id of stash :-

to if we have multiple.

stashes and add one stash

git stash drop :- to

git stash drop :- to remove a stash.

git stash pop :- to apply and remove a stash at same time.

---

pull & fetch

merge & rebase

reset & revert