**All DevOps Commands**

**Linux:**

* **Create a directory:** mkdir dirname
* **dir**
* **Delete a dir:** rm –rf dirname
* **Rename a dir:** mv olddir newdir
* **Copy a dir:** cp srcfile /dist
* **Move a dir:** mv filename /dest
* **Create a file:** vi/touch filename
* **Delete a file:** rm filename
* **Copy a file:** cp filename /dest
* **Move a file:** mv srcfile /dest
* **Rename a file:** mv oldfilename newfilename
* **Switch one dir to another:** cd dirname or cd /opt
* **Difference b/w two files:** diff file1 file2
* **Display content of the file:** cat filename
* **Display the list of files/dir:** ls, ls –a(hidden files), ls –l, ls –latr, ls -latrh
* **Redirecting or Appending commands:**

cat file1 > file2 (copying files from file1 to file2)

cat file1 >> file2 (copying multiple time from file1 to file2)

* **Create a tar file, untar and display content and untar:**

tar –cvf name.tar filename (creating a tar file)

tar –xvf name.tar (untaring the tar file)

tar –tvf name.tar (display the content and untar the tar file)

* **Create a zip file and unzip:**

zip name.zip filename

unzip name.zip

* **Grep commands:**

grep –i “exp” filename (displays the exp in the file)

grep –v “exp” filename (ignores the exp and displays remaining content)

grep –r “exp” \* (displays the exp available in dir)

* **Sed commands:** sed –i (s/oldname/newname/g) filename (Stream Editor)
* **Yum commands:**

yum update

yum install

yum remove

* **Change the permissions of the file/dir:**

chmod –R 400 dirname

chmod 400 filename

* **Create a User:** useradd username / passwd username
* **Create a Group:** groupadd groupname
* **Switch to user:** su - username
* **Get the details of Single User:** cat /etc/passwd | grep username
* **Display the list of users:** cat /etc/passwd
* **Display the list of groups:** cat /etc/group
* **Move users into the group:** usermod –a –G groupname username
* **Delete user:** userdel username
* **Delete group:** groupdel groupname
* **Change the Ownership of the dir:** chown –R name:name dirname
* **Change the ownership of the file:** chown name:name filename
* **Find a file/dir:**

find / -name filename

find / -opt filename

* **Create a log file:** vi name.log
* **Check the log files:**

head -19 name.log

tail –f name.log

tail -20f name.log

* **Print the file details in column wise:** awk “{print $1, $7}” filename
* **Check the IP address:**  if config
* **Check Disk space:** df –Th **/**  df -h
* **Check Directory space:** du –sh dirname
* **Check OS version:** uname –a **/**  cat /\*-release
* **Check Memory/RAM:** free -m
* **If the memory is full clean the memory:** sync ; echo 2 > /proc/sys/vm/drop\_caches
* **Check who logined in:** who
* **Check the details of running time:** uptime
* **Check the details of CPU usage:** top
* **If the usage is high kill it:** kill -9 commitid
* **Download a file from internet:** wget url-link
* **Verify/connect to internet:** ping google.com
* **Get the details of site address:** nslookup google.com
* **Connect port with Domain name/ IP address:** telnet google.com/ipaddress
* **Check History:** history
* **Check Process ID status:** ps –ef | grep proceesor-name
* **Check the Port status:** netstat –anp | grep port-number
* **Create a Pub/Pvt key:** ssh-keygen
* **Connect one linux machine to another:** ssh username@ip
* **Send a file from one linux to another:** scp –r dirname username@ip:path
* **What are run levels:**

***Git:***

* **Difference b/w SVN and Git?**

SVN is a Centrailised Version where we have only One Repository whereas Git is a

Distributed Version System where it has Two Repositories Local and Remote.

* **Difference b/w Git hub and Git lab?**
* **How many repositories are there in Git:** Two Repositories (Remote and Local)
* **How to Clone a repository:** git clone url-link
* **How to verify user:** git config –global user.name “name”
* **How to verify email:** git config –global user.email “email”
* **Display the list of users:** git config –global --list
* **How to edit the list of users:** git config –global --edit
* **Create a file, send it to staging area, local repo and remote repo?**

touch filename (Working Stage)

git add . (Staging Stage)

git commit –m “exp” (Local Repository)

git push (Remote Repository)

* **Check status:**  git status
* **Check logs, types of logs:**  git log / git log –oneline **/** git reflog
* **Display single file details:** git show –p commit-id
* **Move the file from Local repo to Staging area:** git reset –soft commit-id (down)
* **Move the file from Staging to Working:** git reset –mixed commit-id (down)
* **Remove the file from Local repo?**
* **Get the files from Remote repo to Local repo?**
* **Get the files from Local to Working?**
* **Get the files from Remote to Working?**
* **Edit/remove the file from Remote?**
* **Remove the file from Local and then in Remote repo?**
* **Rename the file from Local and then in Remote repo?**
* **Create a Branch: git branch branchname**
* **Switch into branch: git checkout branchname**
* **Create a branch and auto switch: git checkout –b branchname**
* **Check the list of the branches: git branch**
* **Merge a branch with another branch: git merge branchname**
* **Merge a single file in another branch: git cherry-pick filename**
* **Delete a Branch: git branch –D branchname**
* **Delete a branch from Remote area: git push origin –delete branchname**
* **Display all the files in Stash: git stash list**
* **Save a file in Stash: git stash save”exp**”
* **Display the details of the file in stash: git show –p stashid**
* **Get the file from stash to staging area cut &copy: git apply/pop stashid**
* **Delete a file from stash?**
* **Create a Ignore file?**
* **Create local repository?**

**Jenkins:**

**Docker:**

* **Install Docker:**
* **How to check docker version:**
* **How to start/stop/status docker:** systemctl start/stop/status docker
* **Download a docker image:** docker pull imagename
* **Check the image is downloaded or not:** docker images
* **Display the list of images:** docker images
* **Create a Container from image:**  docker run –itd --name myname imagename
* **Check the list of running containers:** docker ps
* **Check the list of all the containers:** docker ps –a
* **How to start/stop the container:** docker start/stop contid
* **Go inside the container:** docker exec –it contid /bin/bash
* **Check logs of the container:** docker logs contid
* **Delete a container:** docker rm –f contid
* **Delete a image:** docker rmi imagename

**Ansible:**

***Terraform:***

* **Check terraform version:** terraform version
* **How to initiate terraform:** terraform init
* **How to validate terraform:** terraform validate
* **Check any changes are made or not:** terraform plan
* **Display lines in a format:** terraform fmt
* **Execute a terraform file:** terraform apply
* **Delete a terraform file:** terraform destroy
* **If any changes that are made in terraform file are stored in:** terraform.tfstate
* **Where terraform plugins will be stored:** (.terraform) file plugins will be stored
* **Where will be terraform save its STATE Files:** terraform.tfstate

We can also keep this file in a centralized location like S3-Bucket.

* **Where will be backup files kept:** In (.terraform.tfstate.backup) file
* **Modules:** **P**rovider, **R**esource, **V**ariable, **P**rovisioner.

**Provisioner:** File Provisioner is used to send files from Source to Destination. And Local-Exec and Remote-Exec is used for connecting SSH from One linux to another.

* **To execute a state.tfvars file:** terraform plan –var-file test.tfvars(Its .tfvars file)
* **To create a state.tfvars file into a another file:** terraform apply - -var.file test.tfvars –out myterraformtesting(file name)

***Kubernetes:***

* **From Master run the command that ensure all the nodes are running or not: kubectl get nodes**
* **How to get in details of pods that running in nodes and more info:** **kubectl get all –o wide**
* **To get the details of the Nodes:** **watch –n 1 kubectl get nodes**
* **To copy a yaml file link from github**: **kubectl apply –f https:// url link.yml**
* **To create a Name Space**: **kubectl create ns twitter**
* **To get the info of that Namespace**: **kubectl get all –n twitter**
* **Download the yaml file from github**: **wget https:// url link.yml**
* **To edit the downloaded file**: **vim**
* **To directly apply a file: kubectl apply –f filename.yml –n twitter** (if u doesn’t mention namespace it takes the default namespace name)
* **Display how many namespaces are running: kubectl get ns**
* **To delete the namespace: kubectl delete ns twitter**
* **To delete a pod:** **kubectl delete pod/nginx –n twitter**
* **To get the details of deploy in namespace:** **kubectl describe deploy -n twitter**
* **To get in detail description of container:**  **kubectl get all –n twitter –o wide**
* **To watch the nodes: watch n -1 kubectl get all –n twitter –o wide**
* **How to get the roll out status:** **kubectl rollout status deploy/nginx –n twitter**
* **How to get the details of roll out history: kubectl rollout history deploy/nginx –n twitter**
* **Now the rollout is in 3rd version how to get to 2nd version: kubectl rollout undo deployment nginx –to-revision=2 –n twitter**
* **How to scale up the pods:** **kubectl scale deployment/nginx --replicas=10 –n twitter**
* **How do you scale up pods through yaml file: vim deployment-twitter.yml (goto replicas and change the number)**

**Then: kubectl apply –f deployment-twitter.yml –n twitter**

* **How to add a new Node in middle: kubeadm token create --print-join-command (**give this command it will generate a token and execute that command in New Node**)**
* **How to intilize helm: helm init**
* **How to create a helm skeleton: helm create**
* **To get Helm version: helm version**
* **How to create a project in helm: helm create projectname**