Tech Dev-Ops(B)2.2

TASK -1

Submitted by:

Neema Abdulkader

Employee ID: TEN/DE179

SECTION 1

Basically when we talk about devops we have to understand what a software development methodology is so in general any organization, how the software is developed let us say there is a requirement and that has to be converted to a project and software so for that there are stages:
i.e. we gather the 'requirements' from the customer so that is requirements analysis genes this is done by the ‘line of business people or the managers’.
Then we do the ‘design’ in design. It is of 2 types: high level design, low level design, so that is generally done by the architect.
So when we say design, it will be something like low level, high level, design where we basically decide which is the OS we want to support, which is the programming language we want to support, which is the scripting language we want to use.
Next is coding, which is generally done by the developers.
And when it comes to low level designing what all the components we should have how the projects should look like that kind of things.
In the cording developers as per the design code based on the programming language.
Test is of in different types:
Unit test, functionality test, system test, acceptance test, user acceptance test so once everything is low the you push the code to the production
Production: This is a regularly a software development methodology
Deployment: Something like, let a say there is an application in can be anything it’s a city backup or hdfc or Amazon which is relevancy on a server which is inside a data Centre so there is someone who is maintain their is a production team there will be development team there is a test team there's a design team so when something has to be done let us say there is a new feature has to be added to the website there is a design team who the design this development team who does the coding we there is the testing who test and there is a production team who pushes it into the server
All these phases are followed, but that is like not in 6 months of time but we can wait a few minutes for the development to happen. How is this happening? This is happening because of this end to end automation.
So basically it is like the deployment have to be done very faster in a very quicker fashion so that it has to be done accurately so that is the reason why we have this end to end automation so everything has to be automated
E.g.: Let us take an example of there is a
Development team
Test team
Product team
So generally let us say there is a development coding in the sense there have been multiple are working on a single project they store their software on a frame like that can be svn.git tar. or any other version in the system i.e. we don't store our software on a laptop, we stored a centralized server so that even if the laptop crashes you have code on the centralized server.

We basically develop the software and then we push it to the centralized server. So multiple people will be using the centralized server for storing the source code.
* End to end automation is called **continuous integration.**
**tools we use for this are:
Jenkins - which is continuous integration server. Using which we can achieve this End to end automation.
So this is a kind of framework wherein we need to add other tools like version control tool or build tool.

Dev-Ops

This is a set of principles or methodology where in the development and operation team collaborate to form a new model called Dev-Ops.
What does the developmenteres do?
They basically focus on coding, when we say development team it is like:

Analyst/architect
Developers
Testers
Build team

all these are the part of development sphere

The Moto of this team is to develop code. So they don't used to bother about stability. That they do not bother about the high availability.
When its in production:

Release engineering
Linux admin
Database admin/system admin

operation team

This team, they don't bother about te coding, only bother about the system as to be always up. I.e., if the any application is down like direct business loss to the client. So who will be questioned here the person who will be questioned is the system admin.
These are 2 different sets collaborate together hat is called Dev-Ops.
This system admin generally, scared of question the changes to the servers because they always bother about the high availability. High availability in the sense always the server the machine has to be available always the application which is running on a server has to be available i.e., if you open any site that has to available to the customer. If by any chance if the server or the application goes down, it is a directly business loss to the customer.
Ex: I am a system admin and there are 100 machines in my lab.
There is software x running with the version 1.0.
Now, there is a new version developed in the market which is 2.0. So I need to upgrade all my machines to 2.0
So how do I apply manually?
So i need to login to each machine and upgrade there will be some procedures to upgrade and then i need to come back like 1 then 2 then 3 likewise..
If 1 machine is taking 10 min, for 100machine it will take 1000min.

So, let us say if there is 1 lakh server. Then how do we upgrade this software companies like Google /Facebook etc. all these companies have servers more than 10 lakh.
So simply updating a software or applying a patch or creating a user for anything, so we used to login to each machine and do it. So that is like a TDS job.
How do we automate this / how do we enhance this?
For that we have tools. So what do we do is basicall6y have tools like one of them.
We configure a server.
Let us have 100 machines and 1 of them is a server so we called as server/ master and remaining 99 servers we configure as client/ agent.
Then we need to update the software. So for that I will not go manually.
I will write a script and this will not be like java, python programming language. This will be a symbol script that is like general English.
Syntax like:

- install this
- start the service

This will be executed and it will be automatically executed in all the machine.
Here we are not doing manually, but here we are doing automatically using a script.
That is, in Dev-Ops,
In production, operation guy will also think like the developer.
How is it possible?
Instead of doing manually here we are writing script and it is the one-time job. For the first time we will write the script. Next time for the same reason instead of writing the script we will call this script does the job.
So here system admin is coding and he is maintaining code. This is nothing but operation guy is thinking like developer is Dev-Ops.

- Version control tool- Git/github
- Continuous integration- jenkins
- Configuration management- chef/puppet/ansible
- Monitoring - nagios
- Development environment - vagrant
- Linux
- Shell script
- Build tools- ant/maven
- Artifacts - nexus/artifactory
- Virtualization - docker/containers
- AWS - cloud
- python

Monitoring:

When we say there are 100 of server and we need to monitor them. So because of any reason if the CPU increases on the machine, the machine will go down and again the application will go down so continuously we used to monitor.
Tools like : Nagios which does the monitoring
So nagios is a kind of a graphical user interface we can add allover servers to it and it will show that CPU stage. And whatever the requirement on the geographical console
Development environment:
Vagrant tool
Linux command
3 step commands to get environment

1. We have to download something on the windows laptop or Linux or mac
2. Install this tool that will create an environment.
3. Internally we download a Linux image on laptop on top of that it will create a vm, then we can login to the vm and use it.

The concept is called virtualization.
Workflow:
Linux
Shell script
Version control tool
Built tool
Artifact
Continuous integration
Configuration management
Monitoring
Development
Virtualization
AWS
Python
Operating System
Os is a set of software. Which is like an interface between the user and the hardware.
So if I want to communicate the hardware I need to communicate through a layer of os.
We use primarily,
Windows
Linux
Mac etc.
When we talk about sever, it is mostly Linux os.
LINUX (Linux forwards)
Also developed kernel.
Many of the organization tool this kernel and they develop their own utilities, applications, commands around that and they call it as a **distribution**

Type of Linux distribution:

- RHEL- redhat enter prize Linux - current version 7
- SLES- sase Linux enterprise Linux- 12 Novell
- Ubuntu- 17.10 debian
- Centos - 7
- Fedora - 25

Features

1. Open source
2. Free
3. Platform independent

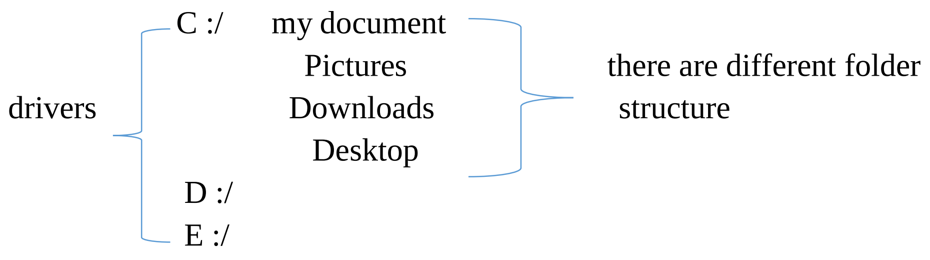
Difference between windows and Linux

Windows	**Linux**
User level- desktop	Server edition
	Case sensitive

SECTION 2
Architecture of Linux:

- Basic part is hardware
- The first layer is hardware
- Hardware in the scene, it include it as a server.
- When we say server it is like there are hardware vendors who manufacturer server and who sells servers, so that server has a processor, RAM, hard disc it has lot of external hard disc. It has the adaptors, the fiber channel adaptor to connect to the storage. It has Ethernet adaptors to connect to the network. On top of the hardware we have kernel. Kernel is the interface between shell and hardware. This is the main component of Linux.
- It is responsible hardware management/ process management/ CPU management/ memory management. i.e., any hardware device add and any hardware device removed it is the kernel responsibility to detect.
- Kernel is the one which detect new hardware.
- Next part is shell, where the user interacted with the machine. Shell is the interface between user and kernel.
- In case of Linux we have cli where we type the command shell is a terminal or Linux also have the graphical user interface but when we talk about server we mostly use shell.
- In case of windows, shell is icon which are used to interact those are the shells in windows.

Let us say if i want to instruct a machine to perform something. For that i type the command on the shell. Shell interact to the kernel and kernel does kernel interact to the hardware. Ie, how we gets the work done.
On top of the shell there is utilities commands /applications which are running under the shell.
Directory structure
Directory structure of windows:



A folder is called a directory in Linux. Folder is nothing but, collection of files.
Directory structure of Linux:
/----------/home => home directory of the user
/var=> log files   [application.log - for debugging]
/etc => configuration files
/temp => storing the temporary files

/opt => third party software
/lib=>storing library files
/root => home directory of the root
/bin =>storing binary files
/dev => when we say, we added a device to a Linux machine. It can be external hard drive or USB. That by default shows up in /dev.

Var

Logfile:

Is the storage is full or CPU is full etc. are debugged in log file in var directory.
Log file can be for applications and also OS related information's.

etc

What are the configuration file?
Every application make some configuration if you specify the user name and the password or you want to specify the directory or you want to specify the coding , which the application are.
So those kinds of setting generally do in /etc.
In Puppet:
/etc/puppet.conf => configure the file for puppet.

temp

Generally we copy some of the files to temp and as part of writing some script. We want to store some status which is not made after executing the script.
We store that in temp files

/opt

Like java etc installed stored in opt directory.

/lib

Some files which are related to libraries lib 64, 32 bit, 64 bit all those things stored in /lib. Libraries basically needed for the application installation, configuration execution.

/root

It is the administrator for Linux. It has the highest privilege unrestricted access.
How to login to a machine?
Putty => to login to remote machine in windows.
Google→ download putty→ alternative binary files→ 64 bit putty exe→open exe→provide hostname or ip address→select ssh/telnet→username and password→then we will be inside the Linux machine terminal

In terminal

Basic commands are:

ls=>command
   => Display list of files
ls -1=> display long list format
ls -1/=> it will give the directory structure
Reverse timestamp:

ls -lrt

ls -lrt/=> ls- longlist revers timestamp.

Clear => screen will be clear

mkdir=>to create / make directory

mkdir test => will create a directory called test

pwd=> present working directory

cd=> change directory

Cd test=> will change the directory to test

rm => delete /remove the file/directory

rm -rf/a=> everything will be deleted

r=> recursive

f=> force

Most dangerous command in Linux:

rm -rf/a

If you delete a file there is nothing like recycle bin in Linux. So file is gone permanently, you cannot get back once you delete a file.

rm -rf/=> machine will be crude up and we cannot get that we have to reinstall.

cp=>copy is the command to copy the file.

In this way we can take the backup

cat=> to list out the contents of a file.

cd..=> One level up to previous directory

SECTION 3

Git

Download git→windows→64 bit git for windows setups.
As like that oracle and vagrant
You may have to reboot your machine once you install all these.
How do we enable virtualization on top of the laptop?
Clear all your application
Reboot your machine
While rebooting your machine→ esc (continuously)
Till you go into→Bios menu and there is an option called virtualization technology vt -x →there you will have 2 option
Deisabled/enabled→ enabled→FIO(save and continue)

Basic Commands

ls =>List
ls -lrt=> long list
rm=>remove
rmdir=>remove directory
rm -rf=>forcibly, recursively
mkdir=>create director
cd=> change directory
cd..=> one level up
****
/a =>pwd
cd /b/c/d/f
Pwd
cd../../.. => 3 level up
cd- => go back to previous directory
ls -lrt =>actually gives 9 columns
ls -lrt/=>get all the directories in the /
/=>is like a highest point in Linux
r=> only read permission
chmod=> change mode(everyone has access)
+=> for added permissions
-=> remove permissions
cat/etc/*release=> it will generally give the information about the version
=> it will show Linux flavor/version

Let us say there is a Remote Linux machine and I want to login from here. So what are the commands you are to use and what is the tool we need to use, so we used a putty tool to login to a remote machine from a Windows laptop. download putty and give the username and then give IP and then give password so it just give you hostname and IP address in the box and then will you

select Telnet and ssh and then it gives you the prompt for entering username and then we enter username and password will be inside the mission if we give the correct password. so we use two protocols, one is Telnet another is ssh so these are 2 protocols used for logging into the machine.
Telnet results on a port 23
Ssh results on a port 22
But there will be a demon associated with it.
What is the demon?
What is the process
What is the program
Example
Let's write a java program or C program or C ++ program so we execute that in a Linux machine. So depending on the programming languages compiled as on a Linux machine you have written a *helloworld.Java* program. So what will be there inside the program will be something like *system.print.telnet=helloworld*. Then once we have written this program we have to compile it. we are compiling using *Javac helloworld.Java*, what happens if we compile this will actually provide a certain rate of a binary call *helloworld.class*. Compilation then we execute using *Java helloworld*, so this will provide output called helloworld to print hello world using Java.

Example
Program is taking 2 hours to execute so when we start a program it actually starts the process. So the program under execution is called the **process.** You started the program and executing a program that will be a printing something, it is executing something, it is creating some files, removing some files, they have some features associated with program so people will be able to login to that program are some website is running so actually when a program is running on a Linux machine internally there is a process associated with in it. The process is nothing but the program under execution. How do we see the process is *ps* is the command to list out the process. *ps -ef* will give the list of all the processes running on the Linux machine and so this process here ssh or Telnet are two programs which are already running on the machine so they will be running 24/7 in 365 days. So then only we will be able to login to a Linux machine so this will actually not run in the terminal in the foreground they will run in the background, so the process running in the background is called a **demon**. so ssh Telnet are two protocols to login to a machine, if not able to login to a machine using ssh to tell that there might be a network issue are there might be some problem with respect to Telnet and ssh, maybe the process not running, maybe the process not listening on the port, maybe there is a problem with the demon, there might be some problem in the configuration file, so these are there in which comes up if you want to debug.
If you use Telnet and the machine name it will be prompted to give username and password and if you enter the password that password can be hacked easily by anybody but if you use ssh the password which you enter will be encrypted and sent to the reason why it's always preferable to use ssh.

Git

Git is like a terminal, something like putty

Step

Open git bash→open a terminal.

Instalation media for Linux => redhat -7-Sp1-×86_64.lSO

oracle VB => is a kind of hypervisor

It is a vertualization enable

Which is beeing on top of the window laptop. that will vertualize the environment

- it will give a way for us to run multiple virtual machine on top of the windows

Vagrant => preins talled image everything is done you just have to download the image and get the UM in 3 steps

1. open git bush - open a terminal
2. create directory

   =>mkdir /vagrant 1

   cd/vagrant 1
3. vagrant init => this will create something called vagrant file
4. modify vagrant file to a specify OS

   config .um.box => ubontu / xenial64
5. vagrant ssh => you'll login to the machine

SECTION 4

cat file name => it will open the file and show the contents of the file so if I want to display only 1st 10 lines of the
file =>

head -10 file name

display the last 10 lines of the file
=> tail -10
Pipe operation:

Let's have one command and l want to redirect output of the command as input to another command. Let's have command one and I have command two so command one give some output that output I want to give as that the second command so that the second command works on the first command output and it gives another output from the command one that is called pipe operation and this is the symbol /:|

Wc => wood count
Wc => line number
Wc => character number
Wc -w => word number
Search for particular word / pattern =>
Grep => global regular expression print so this searcher
Grep - I => ignore the care
-w => search for particular word
- n => line number
-An=> print n number of lines after occurrence
-Bn => print n number of lines before occurrence

There is an application which is running on Linux machine and that application is logging some information in that log file and how do we debug that?
Vi or
Cat/var/log/massage
This will display whole note of content
This is like a dynamic
Tail - f /var/log/message
Lai/=>shows the log file dynamically

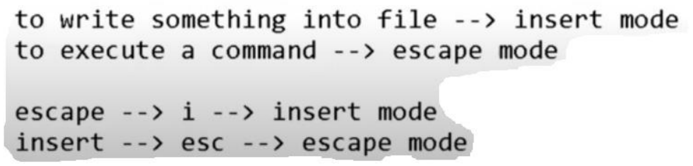
There are some things called editors. So it is useful to write something to work on files to manipulate files. That you want to store something on a file so I just took note or notepad++ to write something and save it so that is where you can edit it to you can delete it you can change something you can replace a word all those things you can do using editor's. So notepad-plus-plus is an editor, the same way in Linux we have several editors **vi, vim, nano**. So these are the primary editors we have in Linux. If you want to open a file and write something to it to open any of these files. We can also open *cat*, that will display the entire thing on the console but if you want to read something for readability we can use vi or vim or nano. So this is basically vim is an improved version of Vi. nano is like a recently developed and it is useful for the beginners to start. So vi and vim have a lot of comments, so nano has Limited functionality basically for the beginners to start with. So vi has a lot of comments which we can use to copy, align, cut a line or search for something so all these things can be used in vi and vim. So the recommended one is vim. So for a developer or for a programmer, it's always recommended to use vim because vim has something called correct Syntax. So you write something that will show in colors something like {you are the places where it is started where it is ended and when you write a program you have to follow the syntax. The syntax to be highlighted is colored with color when you use vim.

vi

In vi there are two modes:

- escape mode/command mode
- insert mode

So to write something into the file to edit the file we have to be in insert mode. To execute a command on the file we have to be in escape mode. But Switch between these two if you press **i** from Escape mode will go to insert mode and when you are in insert mode if you press escape will go to escape mode.



```
to write something into file --> insert mode
to execute a command --> escape mode

escape --> i --> insert mode
insert --> esc --> escape mode
```

To write some text into the vi editor with just enter open the file then press **i** to insert something then write the text and then go to Escape Mode then save it. So while saving we have some command. *:wq* is to save and quit. *:w* is to just save it will not come out, wq will come out by saving. So *:q* just quit, *:q!* will quit without saving. In the same way there are other commands written. Let's say I want to copy one line and paste it, so *yy* is to copy; *p* paste; *dd* cut.

nny will copy n number of lines; ndd will cut n number of lines and; p to paste.

Multiple files

Vim -o file1 file2 -will open 2 files vertically

Vim -O file1 file2 -will open 2 files horizontally

Control + w -switch the files

SECTION 5

How do we install an application on a windows machine? You want to install sometime party software on top of the Windows server download *.exe* and we just double click on it that will install. The same way how to install an application on third party software online. So a package you need to install on a Linux machine, so different distributions follow different methodologies that so different commands are used by different distributions. Here redhat/suse/Fedora/centos so they use something called rpm redhat package manager, for managing the packages like if you want to install, Remove a package, check the package version in all these things we can do using rpm command. Here the extension is:

.rpm

Where as in Ubuntu it uses tool something called dpkg

Here the extension is:

.dpkg

How to use it?

rpm -ivh chef.rpm   => is to install chef
rpm -qa => to list all the packages installed
rpm -e chef => to remove the package

dpkg -i chef.dpkg   => is to install chef
dpkg -l => to list all the packages installed
dpkg -r chef => to remove the package

These are some of the commands used regularly to install some devops tools on top of the Linux machine. In Windows we just download the package from the net and just double click on it and it will install the package, it will install third party software. In the case of Linux there are some comments that will do the job. What is the rpm for redhat/suse/Fedora/centos; dpkg for Ubuntu. So rpm -ivh that indicates that i= install v= vergos (means the messages will be shown on the screen) h = hash (it show the percentage of completion writer for installing the package to show 10% computer 20% can be like that will show the percentage in terms of hashes(#)). rpm -qa = query all (means display all the packages which are installed on the machine) rpm -e = erase (it will erase/remove the package). Then on Ubuntu dpkg -i = install dpkg -l = list all the packages dpkg -r = remove, so these are the primary commands which are used in package management.

man pages => To check the usage of a command

man ls => This will show information about the ls command, what is the description of what it does and what all the options it has ls have many options and many arguments for all those arguments it will show. So generally they will be installed by default but in distribution they are not available.

rmp --help => to check syntax

wget <url> => Downloads the package to Linux machine

So how do we download the package?

1. Download to Windows machine, you want to install some package we need to download it. To download we can download through Windows machine and from there copied to Linux machine.
    a. Using tool - winscp => to copy from windows to Linux
2. Or then we can directly download to Linux machine. To download this, we should have an internet connection.
   Using tool -wget

Command to check ip address of the machine:
Ifconfig
Command to check host name of the machine:
uname -na
Or
Hostname
How do we see all the users in the Linux machine?
Command:
Cat /etc/passwd
This contains all the information about the users.
How to create a user?
user will be associated with home directory, once you login you have to login into the home directory so we need to create the user and home directory and there is the shell default shell which user login so there are different shells available:
ksh
sh
Bash shell etc
Bash is the most advanced of all. So shell is something where we type the comments, like its interface between the kernel and the user. Here we need to specify which shell we have to login to, by default some of the Unix operating systems will provide ksh and most of the Linux operating systems shall provide Bash as the default shell. So we need to provide a shell and then there is an option to specify some comments. It is optional, all these things will be specified, a user will be created, a home directory will be created and that user will be associated with a particular shell and when we login with that particular user to the machine you will be inside your home directory.
Command
useradd -d /home/user1 -s “/bin/bash” -c “comment” -m user1
How do we check the user is created?
Command
Cat /etc/passwd
How to set the password?
passwd user1
How do we login to a particular machine from other machine?
Command:

```
Ssh -l user1 ip/hostname
```

How do you switch within the user?

Command:

```
su -root
su -user
```

SECTION 6
Advanced Commands

If you want to get the IP address of a machine, what we do is just type *ifconfig -a* it will list the IP addresses. They can have multiple IP addresses, one can be private another can be public. So the IP 10.0.0.125 is public IP. How to do; how to assign public IP and private IP when you are seeing a vagrant but the command *ifconfig* –a *which* will list all the interfaces or adapters. It has one IP address associated with. Every machine will have a loopback IP that is localhost IP. Ever in windows machine will have an IP called 127.0.0.1, this is the localhost IP. Using this IP anybody can connect to this machine and if you want to find out the hostname command is *hostname* this will give the hostname of the machine. Generally the hostname will be something like *server1.in.ibm.com* or *server1.us.ibm.com*. We can launch multiple machines or we can launch with a single vagrant file as well. Here server1 is called a short name; server1.in.ibm.com is called a fully qualified domain name. We can connect a machine using this fully qualified domain name (FQDN). When we are assigning IP, when we are trying to connect to a mission we have to use the full name. With this full name the machine will be reachable. *ifconfig -a* command to find out IP address; hostname is to find out the machine name.

Example

If there are two machines, server1 and server2 because of some reason you are not able to connect to server2 now I want to start debugging. Like to find whether server2 is upper running or is there any problem, why I am not able to reach server1. From server1 *ssh -1 root server2*, it says connection refused or timed out etc. So how will I debug,

1. To is ping
   ping server2.in.ibm.com
   Or the IP address
   ping 10.0.1.2 => this will actually check whether the remote machine is alive or not.
   Alive in the sense it is up and running or not, whether the network connectivity on the other machine is up or not. Generally what you have to do is, send packets to the machine and check the acknowledgement whether all these packets received properly or not, so that is pink just to check whether the remote machine is alive or not.

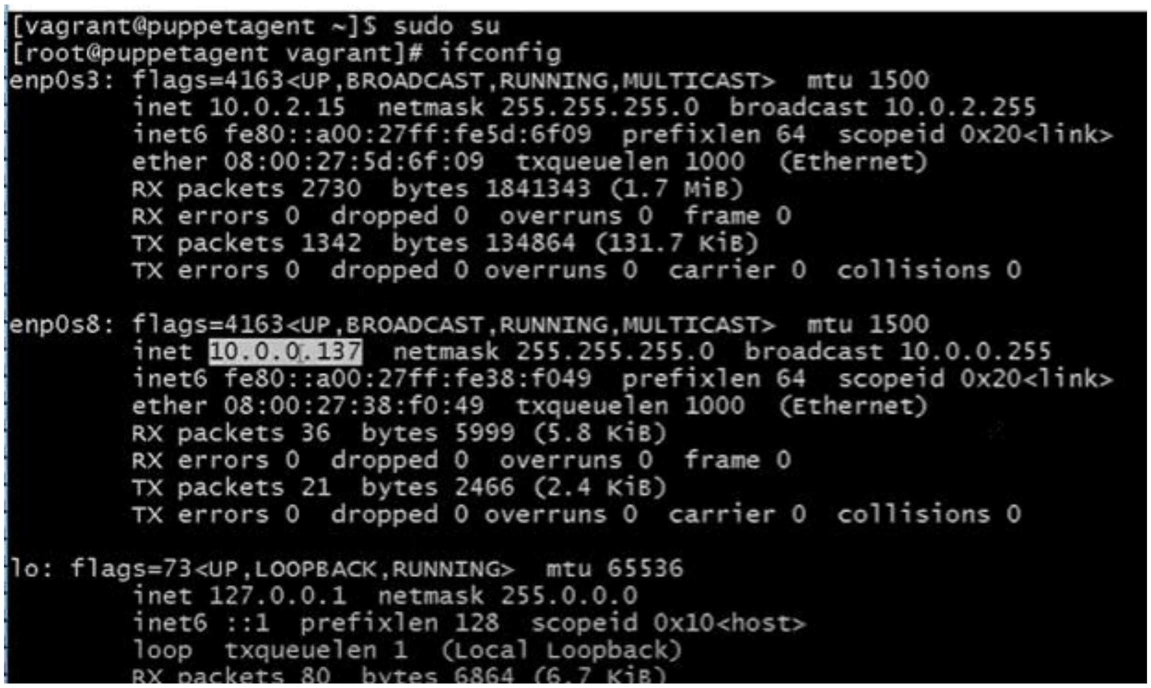
2. In every application which is listening, which runs on a particular machine list on a port. That ports we can find out using:
   netstat -na
   This command lists out the ports on which applications are listening. Every application will listen on a port.
   There is a Jenkins installed by default it will be on port 8080.
   To install Nexus by default it will be running on port 8081, it will change the port. Basically a port is like a logical connection, so when we send out requests to the particular server those requests will be meeting the server on this particular port. So this is a number, so

there will be one application running on each port and if you just want to find out whether an application is running or not we can just say *netstat -an*. If you just give *netstat -an* it will give the information about the ports.

So I have two machines. Log Into one of the machine.

IP address is

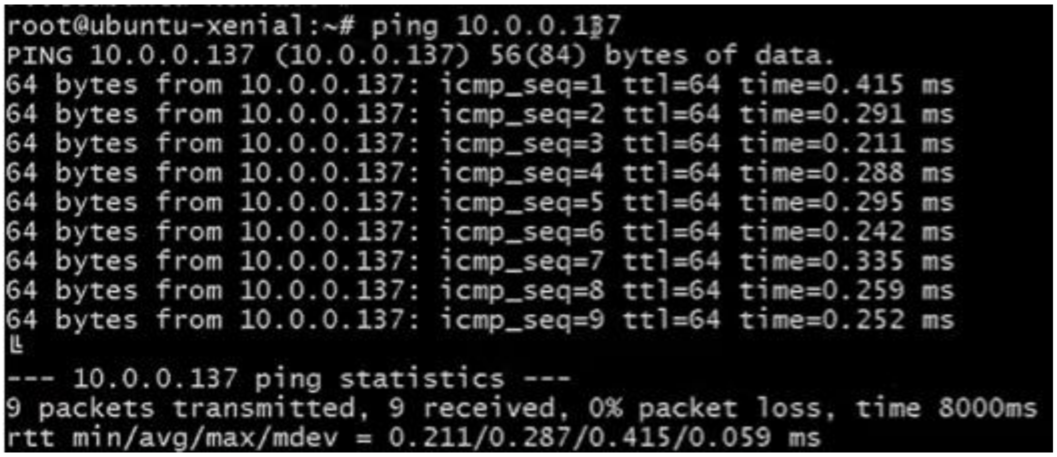
From 1st machine



```
[vagrant@puppetagent ~]$ sudo su
[root@puppetagent vagrant]# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe5d:6f09  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:5d:6f:09  txqueuelen 1000  (Ethernet)
        RX packets 2730  bytes 1841343 (1.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1342  bytes 134864 (131.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.137  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 fe80::a00:27ff:fe38:f049  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:38:f0:49  txqueuelen 1000  (Ethernet)
        RX packets 36  bytes 5999 (5.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 21  bytes 2466 (2.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 80  bytes 6864 (6.7 KiB)
```

Ping this from another machine.

I want to see whether the machine is alive or not



```
root@ubuntu-xenial:~# ping 10.0.0.137
PING 10.0.0.137 (10.0.0.137) 56(84) bytes of data.
64 bytes from 10.0.0.137: icmp_seq=1 ttl=64 time=0.415 ms
64 bytes from 10.0.0.137: icmp_seq=2 ttl=64 time=0.291 ms
64 bytes from 10.0.0.137: icmp_seq=3 ttl=64 time=0.211 ms
64 bytes from 10.0.0.137: icmp_seq=4 ttl=64 time=0.288 ms
64 bytes from 10.0.0.137: icmp_seq=5 ttl=64 time=0.295 ms
64 bytes from 10.0.0.137: icmp_seq=6 ttl=64 time=0.242 ms
64 bytes from 10.0.0.137: icmp_seq=7 ttl=64 time=0.335 ms
64 bytes from 10.0.0.137: icmp_seq=8 ttl=64 time=0.259 ms
64 bytes from 10.0.0.137: icmp_seq=9 ttl=64 time=0.252 ms
^C
--- 10.0.0.137 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8000ms
rtt min/avg/max/mdev = 0.211/0.287/0.415/0.059 ms
```

If you just type ping, it will ping the other machine and when you press *ctrl c* it will come out of the trap. so you can see 9 packets transmitted 9 packets received 0% packet loss this indicates that we have sent some packets we have received acknowledgement and there is no data loss. If the other machine is not pinging you'll get a hundred percent packet loss, this is the proper IP.

The trace route command is used to find out if there is server1 and server2 and when we send a packet from server1 to server2 it goes through a lot of other devices. May be server1 is connected to network switch, so from there it goes to a router and from there it goes to another network switch2 and then from there may be its connected to another router or from there it may be connected to another switch, like this it goes through several network devices and something is not pinging and this machine is not going from server1 and I want to find out the exact packet flow how the transfer is happening like from server1 if i say ping server2 it goes to network switch1, router switch2 and all these things. So there might be some problem at some level, maybe network switch1 is not responding; Router has a problem; switch2 has a problem so to find out that I want to trace the route. So trace route and the IP is the command in network infrastructure if you just type the IP, what happens is it shows all the IP addresses in which the packet travels from here to here and we can see all the devices which are involved in this packet flow. Generally in a data Centre there will be a lot of servers, a lot of network switches, a lot of devices, routers etc, so when we type trace route it exactly shows in which direction the packet flows. If it is stuck after network switch1, which indicates that there is some problem in the network switch1. Maybe the switch is struck or there is any software problem, so based on that we can debug where the network issue is. So this will be useful to find out if there is a network issue. In a typical data center environment if you just type trace route IP there will be various devices, so all those devices will be listed and if it is struck somewhere at some point of time that indicates that the device has a problem.
There is a comment called *top,* this is to find out the CPU utilization on a machine. Just type top it will give the CPU utilization in percentage, we can see the pointy the percentage and use the memory percentage
So CPU consumed by each process and memory consumed by each process.
Why do we need to take care of this because we have to check if any process is consuming more than 80% of the CPU for 90% of the CPU, if any process consuming more CPU then the machine will become slow and there might be some software issue in the process in the application which is causing this? Maybe some infinite loop and maybe some memory leak that we need to find out and that we need to open a ticket or bug to the developer to check. Ideally the CPU utilization should not be more than 80% or 70%, if it is continuously increasing means there might be some memory leak issue that is causing the problem. The CPU utilization increases or it reaches 80% or 90% the machine may crash the same as the memory percentage. That's why we need to continuously check what is the CPU utilization, for that we use the comment called *top*
How to install a package on Ubuntu? What is the extension of a package? Which package do we need to download?
I want to download *chef.*
What is the extension of the package on Ubuntu? How does it look like?
chef -12.1.x86_64.deb This is the extension of the package in case of ubuntu.rpf linux.
So if you want to install a single package *dpkg -i chef.deb* or *rpm -ivh chef.rpm* are the commands which we use.
There is something called dependencies in Linux,

Chef.rpm is dependent on a packet called ruby.
First ruby has to be installed to install chef. Ruby is a dependency for chef. So without installing Ruby we cannot install chef.
Tools to install packages take care of the dependencies
To install packages and dependencies as well. To check that or do that we have several comments for
centos/redhat/fedora we have a utility called *yum* and for ubuntu we have a utility called *apt -get*, so this comes by default with the system that we just need to configure the repository. These are the repository tools to install the packages and their dependencies.
Syntax:
yum install chef
Or
apt -get install chef
From where it downloads packages?
There has to be some site or there has to be some repository, some centralized server from where our machine connects and downloads the packages and installs them on the local machine. I have server1 vagrant installed on the machine and when I say *yum install -y chef* it actually connects to a machine centralized server where all the chef packages are stored and from there it downloads first then installed on the machine. So where the information about you go and download from the site is mentioned for server1, so that is called a repository configuration. What is the repository configuration for rhel/centos/fedora so the file is the */etc/yum.repos.d (*in this directory we tell like these are websites we have to look for this package) and for Ubuntu the file is */etc/apt/sources.list apt-get install ansible* it is the apt source list it looks for the website where ansible depository is restored and it will check for the dependencies automatically and then it will install the dependencies one by one. so apt-get install ansible will and ask for a confirmation you want to install ansible or not, all the packets or not. If I want to automate that *apt -get install -y ansible* this will without interaction without confirmation it will directly ask to install.

You need to have an internet connection to download the packages from the internet, from the repository. If you don't have an internet connection, we can configure the repository using iso, which like *iso* is an **installation media** for Linux machines. So this is like a bundle of all the packages. This we can download and we can use that to install a repository and install the package.
Example
You are under corporate Firewall security and you cannot access the internet from the machine, so you still want to use *apt get* or *yum*. In that case we can configure the repository locally and install all the packages using *iso* or if you have the internet connection then by default there will be some repository that is configured and using those repositories we can install using apt-get or Yum install.

Shell properties:

If you have a very big command and if you want to go to the beginning or end you can give the control+A => to go to beginning

control+E =>to go to end

These are some of the options which shell provides, bash shell has some advanced features using which we can go to the beginning or go to the end.

control+w => to remove a word

control+u => to clear the line

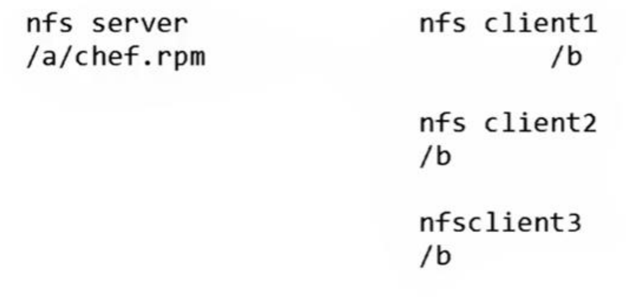
tab => auto fill

control+R => reverse search

SECTION 7

NFS network file sharing

Implementing NFS protocol:



```
nfs server              nfs client1
/a/chef.rpm                     /b

                        nfs client2
                        /b

                        nfsclient3
                        /b
```

So basically what we do is we configure one as an NFS server and we configure the other as an NFS client. From the NFS server we create a directory /a and from the NFS client we create directory /b. Now /a will mount to /b. Mounts are attached, mounting is nothing but attaching a file system to the client. So we can have one server and we can have multiple clients, the /a can be attached to /b /c from different machines.

Example NFS client1 is there the same way I have another machine which is NFS client2, it can be any directory and NFS client3, these are the different machines. So I have three machines and I have created three directories and I want to update some software. For that I just copy my software in nfs server *chef.rmp* so automatically this chef.rpm will also be available in /b on NFS client1 /b on NFS client2 /b on NFS client3, so that is attaching. Basically we are sharing a directory called /a with different machines.

How do we share so there are some steps for?

Stage NFS server configuration:

Identify one machine as the NFS server. Then in that machine we need to export the directory (share the directory)

Step 1. Yum install nfs*

Apt-get install nfs*

Service nfs start => start the nfs service

Service nfs status => check the status

Step 2. Create a directory and share it
So how we share it is: created a directory /a and then we need to update a file called /etc /exportfs, here we give options like share directory/a and you can specify as like read only read and write to all the Machines, can also restrict the Machines as I want to give it to everyone so I want to read only access. Now I create a directory, then I just have to restart.
Mkdir /a
/etc/exports
/a ro*
Exportfs => show the shared directories

On the client:

On the client we have to attach the file system attached directly from the server so Mount is a command then is the server IP or the hostname. let server IP and the directory which is shared on the server, directory is /a and where do you want to Mount it. Let's say I create a directory called mkdir /b and then I attach this to my machine using Mount IP address of the server that it will be shared with and where the directory has to be shared. So /a is shared to /b whatever is available in /a will also be available /b.
Step 1. Mkdir /b
Step 2. Mount 10.0.0.1:/a /b

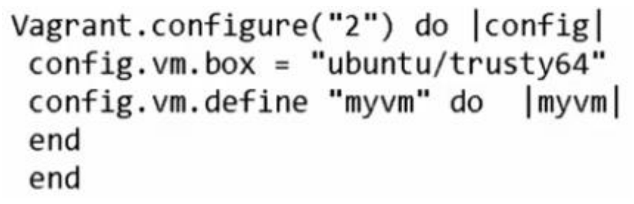
Umount

Command
umount /defs => to detach
This is temporary, if you reboot the machine the mount points will go off. But to make it permanent you just have to add this to */etc/fstab*

How did we virtualize the hardware?

Virtual machine →centos/ubuntu
Hypervisor → oracle virtualbox
Windows laptop → hardware
In windows laptop without using vagrant:
Create a template for the vm
Install vm using the iso
We have to download the iso→ it is the installation media
Installation → select OS
username, password
Locations
Directories (how do you want to partition the directories and that the language the time zone etc.)
Boot disc → to restart the machine of the installation is completed.

So in this way a lot of processes are involved with the installation.
To avoid that we can use Vagrant.

Vagrant →To download the preinstalled image.
When you say vagrant up →from machine it actually pushes the image from Cloud to the local machine.
So whatever the image lets it be Ubuntu/ centos that preinstall image will be downloaded to the laptop and once it is downloaded it uses oracle virtual box to create a vm. Then it brings up the machine the VM with a pre-installed image. So internal it is reading a vagrant file, in the current directory. In the vagrant file, we can give the CPU needed the memory unit and what is the operating system we need and also we can share what is a directory with a Windows machine. A directory can be shared from the Windows machine to Linux machine.
Vagrant file:

```
Vagrant.configure("2") do |config|
 config.vm.box = "ubuntu/trusty64"
 config.vm.define "myvm" do  |myvm|
 end
 end
```
Download an image and bring it up using vagrant up.
It downloads the particular image and vm is up.
Then, login to the vm using vagrant ssh.
Once you login, you can be logged in as a ubuntu user for the ubuntu machines, for centos the default user will be vagrant.
If you want to login to root,
Sudo su- => change the login to root
Once you have the root access, you can install the packages, try nfs, ifconfig etc.
If in 2 server:

1. Mkdir /vagrant1 /vagrant2
2. Cd /vagrant1; vagrant ssh
    i. Vi vagrantfile
3. Vagrant up
4. Cd /vagrant2; vagrant ssh
    i. Vi vagrantfile
5. Vagrant up

We can also create multiple vagrant vms using the single vagrant file. Here we mention vm1 the properties vm2 the properties, then when we say that the vagrant up, it creates two vms. Multiple VMS 3 or 4 vms also we can create using the single vagrant file. It creates vms one by one.
By default there is an interface attached to the VM that is called NAT network address translation. That is like a private IP so by default when you create a vagrant machine and give *ifconfig* that will be one adaptor with one IP that is private IP. So that we can communicate between the vms. If you have 2 vms they can communicate between each other.

In case of public ip:

```
vagrant.configure(“2”) do |config|
Config.vm.box = “ubuntu/trusty64”
Config.vm.network “public_network”, bridge: “Intel(R) Dual Band Wireless-AC 7265 #2”
Config.vm.define “mymv” do |myvm|
End
End
```

Here host is windows
Guest os ubuntu vm

2 ways to get bridge:

1. Open oracle virtual box
    a. Select a vm
    b. Go to network settings
    c. Select bridged if
2. Command to get the bridge name

SECTION 8
Shell scripting:
Why is shell scripting needed?
Shell scripting is needed to automate the tasks. If you are working on a Linux machine and doing things every day and you are doing that manually repeatedly and to save the time and to avoid the manual intervention so what we do is, we do to automate. So you are expecting rf commands manually or contouring and starting something
There are a lot of steps that have to be done manually and that can be automated using Shell Script. Simply we put all the commands together in your file and execute that is nothing but Shell Script.
Generally shell script ends with .sh
In general syntax
Sample.sh → shell script
Sample.py → python script
Sample.pl → perl script
Sample.rb → ruby script
Sample.c → c program
Sample.cpp → cpp
Sample.java → java

What is the first line shell script?
Sample.sh → first line
There are 2 types of program, when you compile/write a program there are 2 things:
a. Compiler - Entire program is executed at once
b. Interpreter - line by line execution
In programming languages like c,c++ or java compile the program first. It converts that to a machine level code, then we exclude machine level code to get the required output. The compiler reads a program from top to bottom once and it compiles. So whereas the interpreter executes the program line by line.
Compilers have an object level code which is generated when we execute the program.
In the case of scripting languages we don't have any object level code. Instead of it we directly execute and we get the output.
#! /usr/bin/bash → first line of the shell script. => This is actually an interpreter.
#! /usr/bin/bash => called she-bang line / magic line
This interpreter location changes based on the distribution:
Some of the distribution use
#!/bin/bash
#!/usr/bin/bash
How to execute a shell script?
./sample.sh → to execute script
Add 2 numbers and print output on the screen:

First it will ask:

```
echo “enter the first number”
read a;
echo “enter the second number”;
read b;
sum = ‘expr $a + $b’;
echo $sum
```

The non-interactive way of programming, we just execute the script and we get the output. Whatever values are required for the script will pass on the command line itself, they are called command line arguments.

How do we access these command line arguments inside the script?

That is, inside the script we can access the command line arguments using dollar ($) values.

$1 =>will print the 1
$2 => 2
$n => will print nth argument
$* => will give all arguments
$# => will give number of arguments/ count of command line arguments
$$ => PID
$@ => will also give all arguments
$0 => will give the file name

SECTION 9

Linux shell scripting:

Validate number of arguments:

```
If no.of args >=2
        Go ahead
        Else
        echo “please enter valid no.of args”
```

$# → to get no.of args

How do I validate the number of arguments printed?

Basically we can use something called loops/decision control loops.

Loops:

If

While

For

if loop syntax:

```
if [condition]
then
--------------------
else
---------------------
fi
```

1→

```
if [$# -gt 2]
then
sum = ‘expr $1 + $2’
echo $sum
Else
echo “please enter valid no.of args”
exit 1;
fi
```

If you want to compare numbers:

Operators used:

gt → greater than

ge →greater than or equal to

le → less than or equal to

lt →less than

eq → equal to

ne → not equal to

Checking the return code of a command or script:

echo $? →return code of the command/script

0 → pass
Other than 0 → fail

Redirection operators:

> → redirect to a file
>> →append operator/added at the end
ls -lrt > file →ls - lrt output is redirected to a file
ie, any commands output are redirected to a file using redirection operator(ie, greater than(>) operator)

While:

Syntax:

```
initialization
while [condition]
do
-------------------
increment/decrement
done
```

1→

```
i=0
while [$i -lt 10]
do
echo $i
i = ‘expr $i+1’
done
```

i=0 =>0 lt 10 →0
i=1 =>1 lt 10 →1
i=2 =>2 lt 10 →2
----
i=9 =>9 lt 10 →9
i=10 =>10 lt 10 →exit

How to debug the shell script?

There are 2 ways:

1. set -x →give this command at the beginning
2. bash -vx sample.sh

for loop:

Syntax:

1→

```
for i in 1 2 3 4
do
```

```
echo $i
done
2→
for ((i=0;i<10;i++))
{
echo $i
}
```

SECTION 10
Functions:
Basically functions are required for reusability. We define it once and use it multiple times that is called **reusability.**
Using the function we will get the **modularity**. The program looks simple, easily manageable.
Shell scripting syntax
Defining:
function function_name()
Calling**:**
function_name;
expr → to perform arithmetic operations
Lets I want to develop a **menu** and I want to give option to user to select something like
want to add two numbers
want to subtract
want to multiply
want to perform a division
and if we enter any of them and ask to enter the numbers. Here I want to develop something like a menu where users can select their input based on their choice, then we want to process them. How do we do that?
There is something called a **case statement**, this is to develop the menus of user choices.
Syntax :
case $n in
To develop menu for the user:
echo "enter your choice
1. addition
2. subtraction
3. multiplication
4. division"

Case $n in:
1. echo "enter 2 numbers"

sed - strem line editor
This is to find and replace the pattern.
Syntax:
sed 's/abc/xyz' file
sed by default will not change the file content. The changes will display on the screen but the actual fill will not be changed. So it is temporary.
Syntax:
sed 's/abc/xyz/g' file
If you want to make the change to be permanent:
Syntax:

```
sed -i ‘s/abc/xyz/g’ file
```

SECTION 11
sed:
sed ‘s/abc/xyz/g’ → replaces abc with xyz in all occurrences
sed -i ‘s/abc/xyz/g’ → permanent
sed 2d file → deletes the 2nd line
sed nd file → deletes nth line from the file
AWK:
Basically to extract the rows and columns.
This is like a programming language. There are a lot of use cases which we can use. Also add the numbers, add the rows, add the columns, and perform arithmetic operations and all can be used using the AWK. In daily usage to extract columns or rows from a file or from an output we can use AWK.
So again we have to redirect output of one command as input to this comment and this comment will process the input and it'll provide the output.
I want to extract the ninth column from a ls -lrt output, so:
ls -lrt | awk {‘print $9’} →this will extract 9th column from ls -lrt output
AWK is to extract a row or column from output. So
This output can be from a command ls -lrt | awk----
Or the output can be from a file cat file | awk----
REGULAR EXPRESSION:
Regular expression which we can use in commands that will display a proper output/ required output.
Wild characters/meta characters

***** → this will match 0 or more occurrences
^ → carrot/beginning of line
$ → ending
? → Single character

BOOT PROCESS:

What happens when you boot a machine?
Whenever you power on any Linux machine the first operation which happens is

1. Post → this power on self-test

Post, which is a power on self-test. That is the phase which happens for the first time, this is basically to check whether all the peripheral devices like monitor, mouse, keyboard are working fine and not. Sometimes when you see power on a machine the keyboard is not connected and the mouse is not connected. ie,signal to all the devices which are connected to the machine and it gets acknowledgement if they are working fine, then it causes and if there is any error it will give the error and you also sometimes see something RAM is not working or some devices not working some disk is not connected. So those are all errors from the posts stage, which is an overall self-test. It basically sends signals to all the devices which are connected to the machine and it will send them. This is like a small piece of software which is embedded into a chip that will basically Check send some signals to the hardware devices and check whether all are working fine or not so that is post. Post is successful, it goes to bios which is a basic input output system.

2. Bios →basic input output system

   Bios is like the system settings. You can also see, you can go to the boot menu and check what all are options available in the BIOS menu. So for the first time if you want to virtualize your hardware we need to enable virtualization capability, so that is actually there in bios so go to bios settings and enable virtualization capability by default disable and you need to enable it then only the virtualization work.
3. Mbr → master boot record
4. Boot loader
5. Kernel
6. /etc/inittab

The process ID of the first process to be started is *init* and process id is 1, then it starts all other processes based on the file call etc/inittab. So / etc/inittab has a lot of information like what services have to start by default, then all the processes running on the Linux machine will be started. Just enter reboot and see what is happening on the machine some of the distributions will have different architecture.

who -r => run level

fsck => file system check

Fsck will do file system check and it'll also automatically correct some of the errors related to the blocks of the memory. If something is corrupted we can run this. This will repair the things if there is any problem with the files. Some files are corrupted, some memory is not proper, some blocks are not proper, all those things will be corrected using fsck.

You want to transfer huge files from Linux machine 1 to Linux machine 2, so what will we do?

/a/

1000 files

Cd /a

tar -cvf sample.tar => will compress all files, create a single file

zip/gzip => reduce the size of file

gzip -9sample.tar =>sample.tar.gz

scp=> another machine

Sample.tar.gz

Gunzip sample.tar.gz => sample .tar

tar -xvf sample.tar => all 1000 files

Cksum→ to check the checksum

checksum you can find out whether the file is transferred properly or not. So how do we do it is basically we do cksum

cksum filename => on the first machine.

So this will produce some output of some numbers and when you copy and we go to the second machine, we also do the same checksum in the filename, these two numbers should match indicating that the file is transferred properly.

SECTION 12
ANSIBLE:

Ansible master (server) **Ansible Agents (client)**

ssh for communication
Ansible rpm has to be installed
push mechanism => we write scripts on master
Pushes changes to agents
playbooks => scripts to automate
YAML => yet another markup language
Entire ansible is written in python
Ansible is of 2 versions:

- Open source => manage upto 100 nodes/machine
- Enter prize => more than 100 nodes --its called ansible tower
  For debugging we should have to buy
  Enterprise version.

Installation

1. Install ansible on master
2. Enable password less authentication between the master and the agent
3. Write scripts/playbooks to automate
4. redhat /fedora /centos
   a. Epel repository
      i. Extra packages enterprise linux
5. Install this on the ansible master
   yum install ansible => installs ansible

By default ansible execution is parallel in nature.

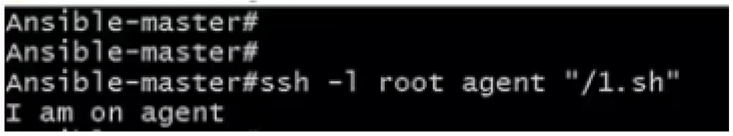
That is:

1. Install ansible
2. Add the agents to master
   /etc/ansible/hosts => inventory file information about the agents, master manages.
3. Passwordless authentication
   /etc/sysconfig/network-scripts/ifcfg-eth0
   ipaddress=
   bootproto= static
   Service network restart
   Ifconfig eth0 9.10.10.1 netmask 255.255.255.0 up

Ansible galaxy => repository of modules
Around 1000 modules are there in ansible.

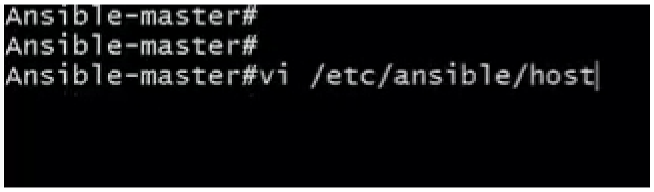
SECTION 13
Ansible
Passwordless authentication:

1. ssh-keys
   Generate using ssh-keygen
2. Copy the keys from master to agent
   ssh-copy-id -i /root /.ssh/id_rsa.pub root@client
   /root/.ssh/authorized_keys => when you execute the command the key will be added to that file.
3. Execute a script/command from master without logging into the agent

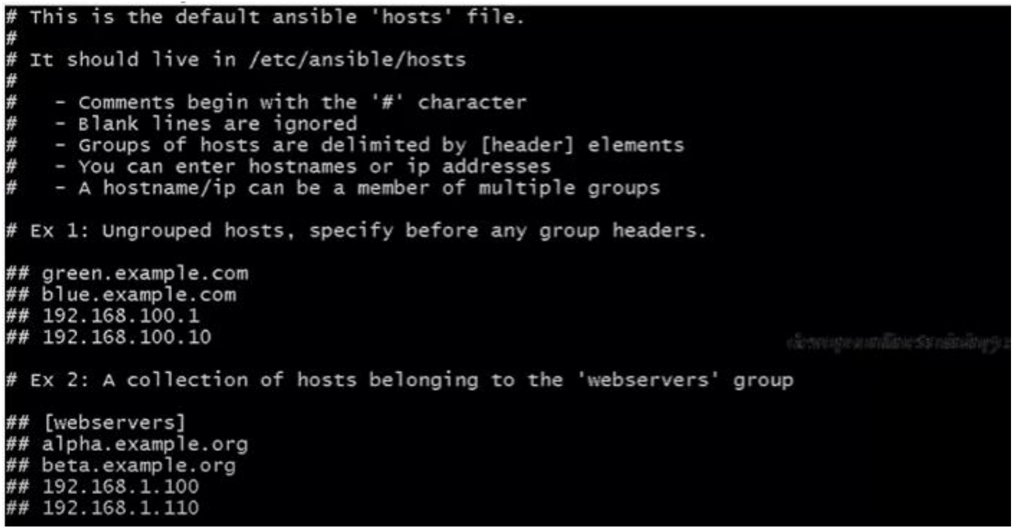


```
Ansible-master#
Ansible-master#
Ansible-master#ssh -l root agent "/1.sh"
I am on agent
```

Inventory file that we have to add in master

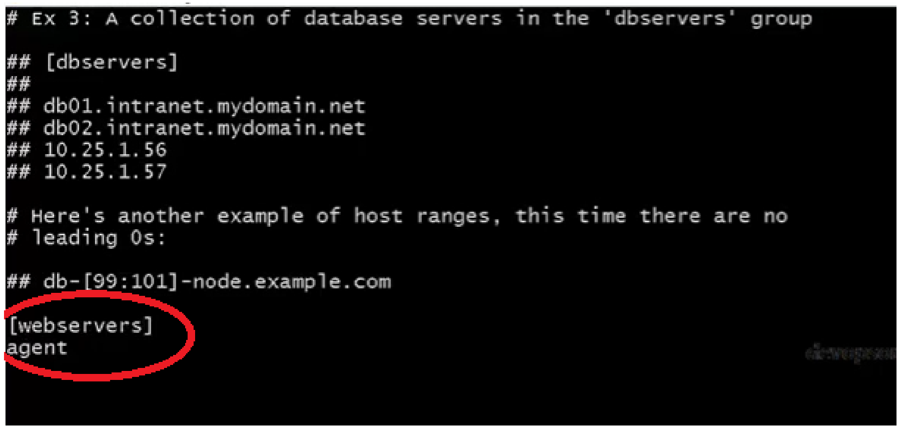


```
Ansible-master#
Ansible-master#
Ansible-master#vi /etc/ansible/host
```



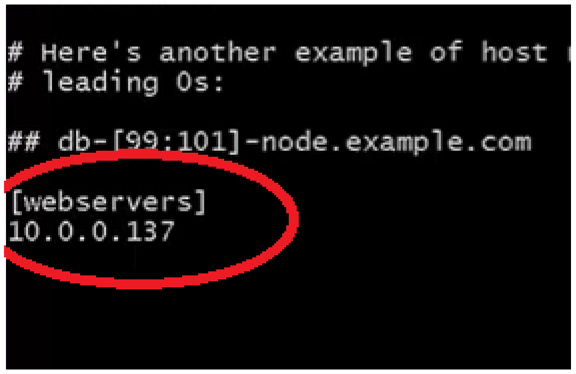
```
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - You can enter hostnames or ip addresses
#   - A hostname/ip can be a member of multiple groups

# Ex 1: Ungrouped hosts, specify before any group headers.

## green.example.com
## blue.example.com
## 192.168.100.1
## 192.168.100.10

# Ex 2: A collection of hosts belonging to the 'webservers' group

## [webservers]
## alpha.example.org
## beta.example.org
## 192.168.1.100
## 192.168.1.110
```

Go to the end and add a new server with a group.



```
# Ex 3: A collection of database servers in the 'dbservers' group

## [dbservers]
##
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# Here's another example of host ranges, this time there are no
# leading 0s:

## db-[99:101]-node.example.com

[webservers]
agent
```

Also



```
# Here's another example of host
# leading 0s:

## db-[99:101]-node.example.com

[webservers]
10.0.0.137
```

[webservers] =>group of many servers/ set of servers.

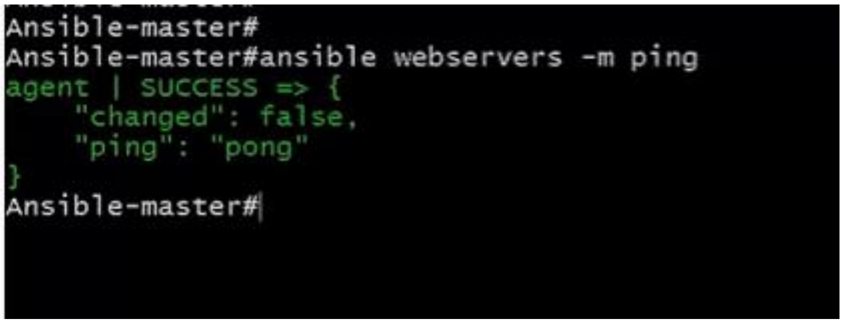
ie,

Web1

Web2

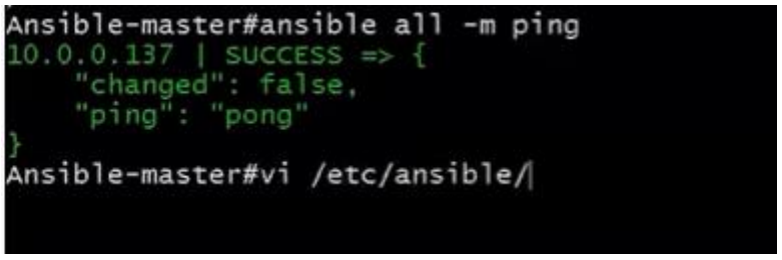
Web3

Ping ansible:



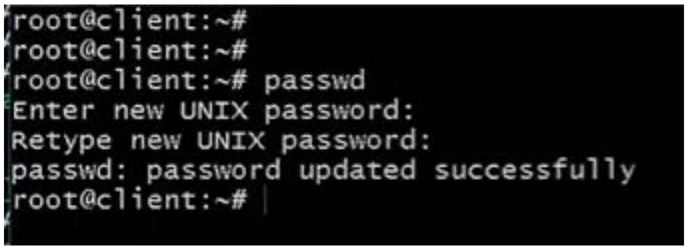
```
Ansible-master#
Ansible-master#ansible webservers -m ping
agent | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
Ansible-master#
```

Also



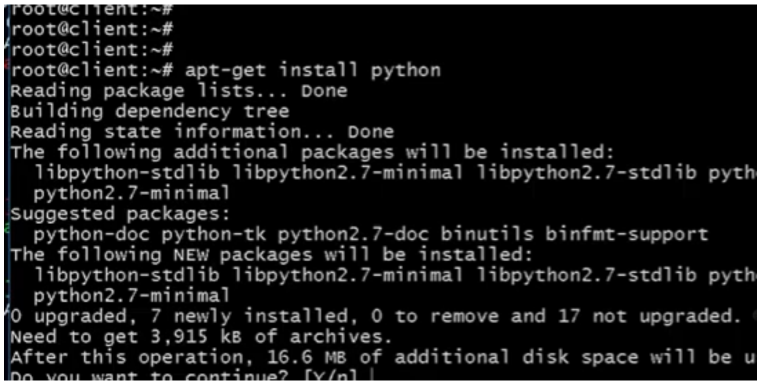
```
Ansible-master#ansible all -m ping
10.0.0.137 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
Ansible-master#vi /etc/ansible/
```

Set password



```
root@client:~#
root@client:~#
root@client:~# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@client:~#
```

Install python



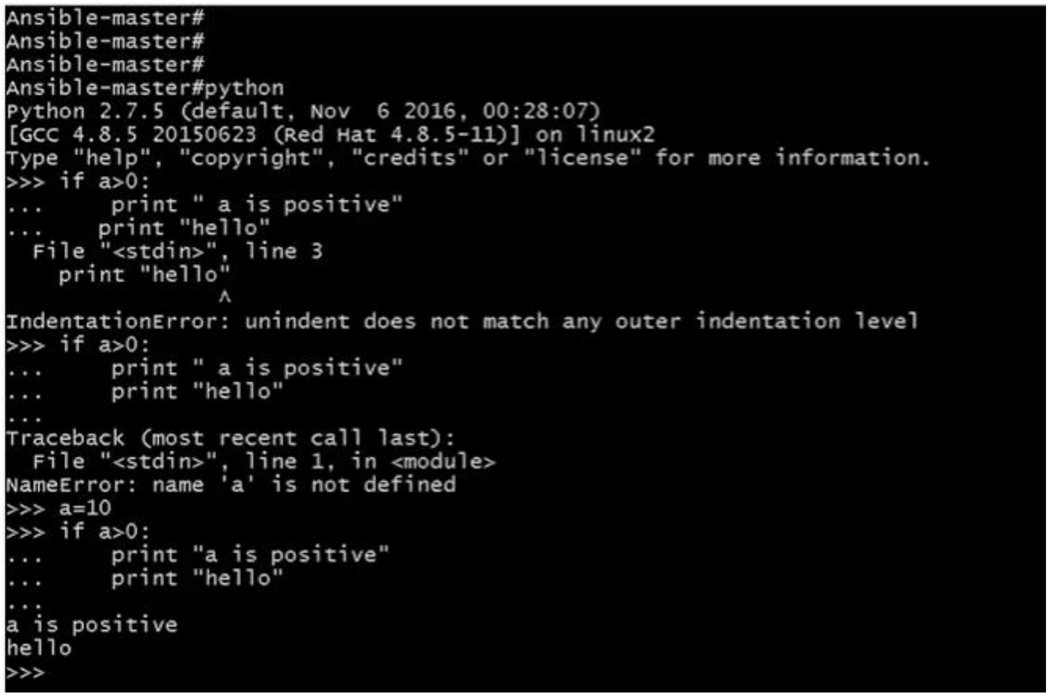
```
root@client:~#
root@client:~#
root@client:~#
root@client:~# apt-get install python
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libpython-stdlib libpython2.7-minimal libpython2.7-stdlib pyth
  python2.7-minimal
Suggested packages:
  python-doc python-tk python2.7-doc binutils binfmt-support
The following NEW packages will be installed:
  libpython-stdlib libpython2.7-minimal libpython2.7-stdlib pyth
  python2.7-minimal
0 upgraded, 7 newly installed, 0 to remove and 17 not upgraded.
Need to get 3,915 kB of archives.
After this operation, 16.6 MB of additional disk space will be u
Do you want to continue? [Y/n]
```

Indentation in python

If a>10:

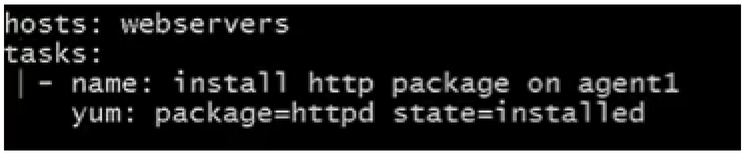
    Print "a is greater than 10"

Note: have to follow indentation strictly otherwise it shows error



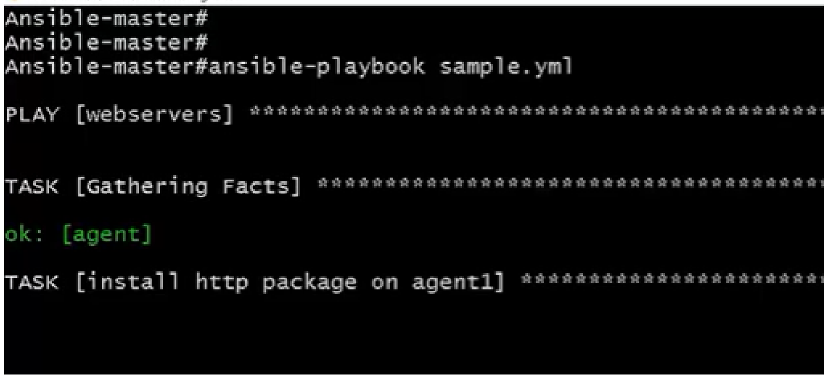
```
Ansible-master#
Ansible-master#
Ansible-master#
Ansible-master#python
Python 2.7.5 (default, Nov  6 2016, 00:28:07)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-11)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> if a>0:
...     print " a is positive"
...    print "hello"
  File "<stdin>", line 3
    print "hello"
                ^
IndentationError: unindent does not match any outer indentation level
>>> if a>0:
...     print " a is positive"
...     print "hello"
...
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'a' is not defined
>>> a=10
>>> if a>0:
...     print "a is positive"
...     print "hello"
...
a is positive
hello
>>>
```

Httpd install



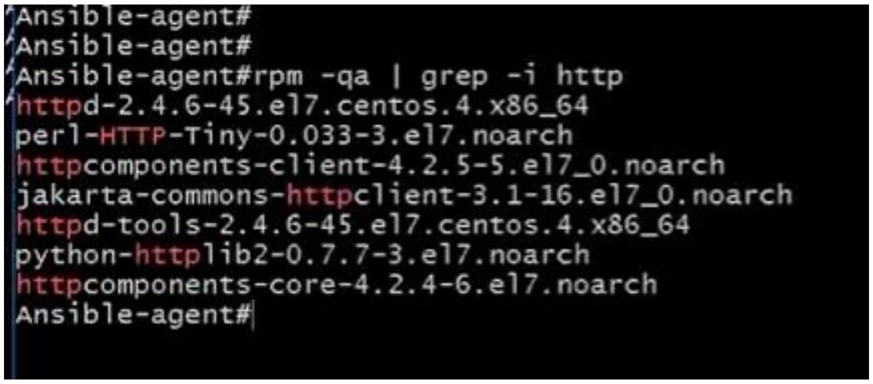
```
hosts: webservers
tasks:
 | - name: install http package on agent1
     yum: package=httpd state=installed
```

Execute playbook



```
Ansible-master#
Ansible-master#
Ansible-master#ansible-playbook sample.yml

PLAY [webservers] ***********************************************


TASK [Gathering Facts] ******************************************

ok: [agent]

TASK [install http package on agent1] ***************************
```

Check whether httpd is installed or not



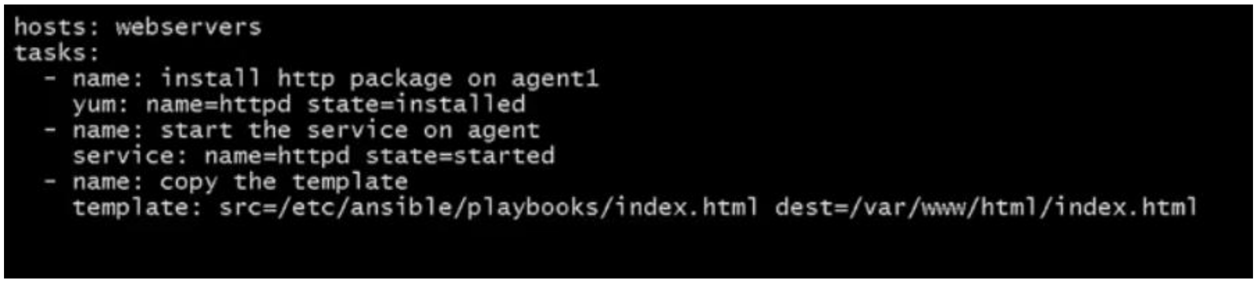
```
Ansible-agent#
Ansible-agent#
Ansible-agent#rpm -qa | grep -i http
httpd-2.4.6-45.el7.centos.4.x86_64
perl-HTTP-Tiny-0.033-3.el7.noarch
httpcomponents-client-4.2.5-5.el7_0.noarch
jakarta-commons-httpclient-3.1-16.el7_0.noarch
httpd-tools-2.4.6-45.el7.centos.4.x86_64
python-httplib2-0.7.7-3.el7.noarch
httpcomponents-core-4.2.4-6.el7.noarch
Ansible-agent#
```

SECTION 14

1. Http install the package
2. Start the service
3. Copy the configuration file => template module
4. Copy some image => copy module

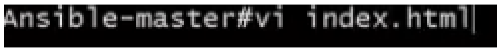
Start the service:

Give the task:



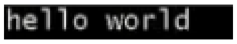
```
hosts: webservers
tasks:
  - name: install http package on agent1
    yum: name=httpd state=installed
  - name: start the service on agent
    service: name=httpd state=started
  - name: copy the template
    template: src=/etc/ansible/playbooks/index.html dest=/var/www/html/index.html
```

Copy a file



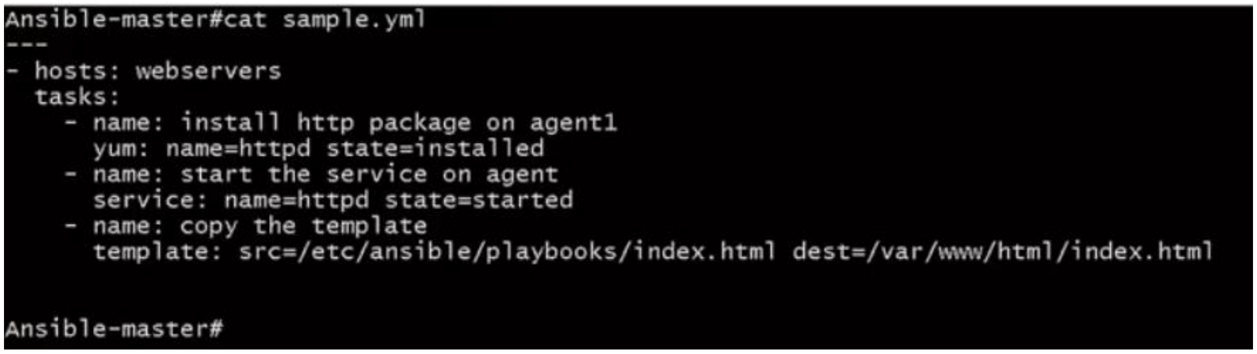
```
Ansible-master#vi  index.html
```

Copy template



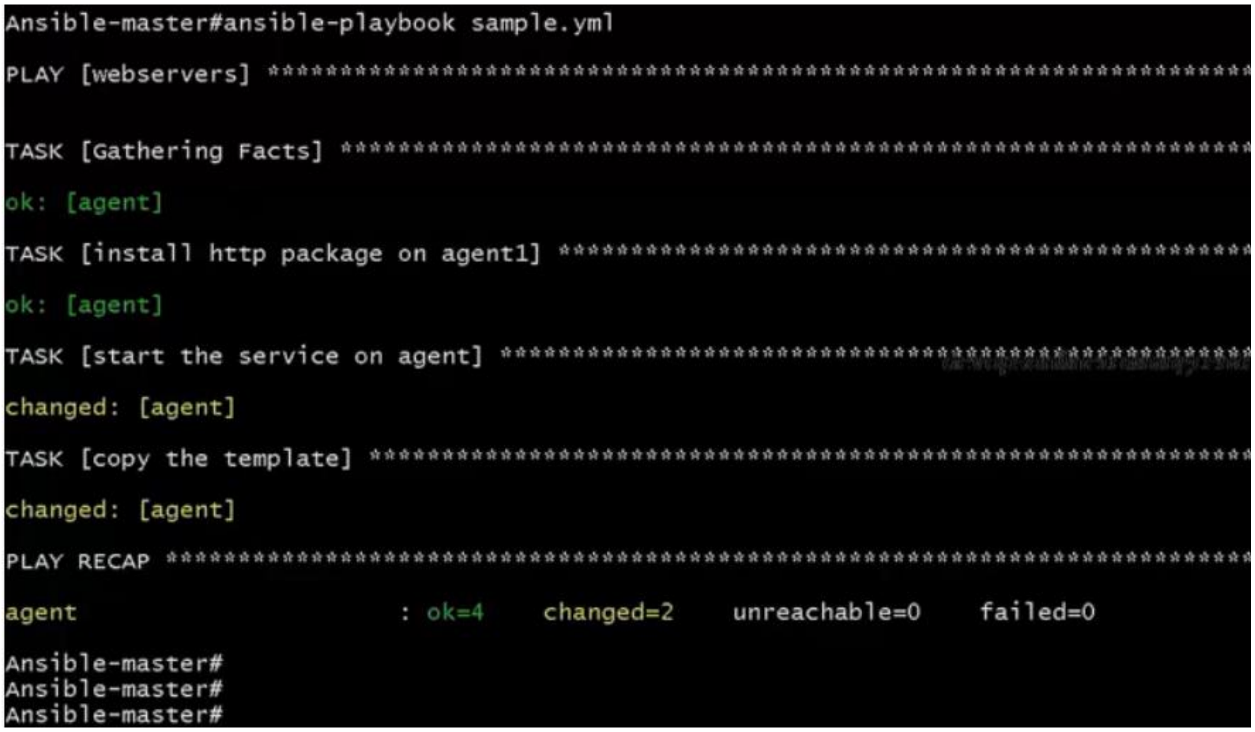
```
hello world
```

Copied to a remote machine



```
Ansible-master#cat sample.yml
---
- hosts: webservers
  tasks:
    - name: install http package on agent1
      yum: name=httpd state=installed
    - name: start the service on agent
      service: name=httpd state=started
    - name: copy the template
      template: src=/etc/ansible/playbooks/index.html dest=/var/www/html/index.html


Ansible-master#
```

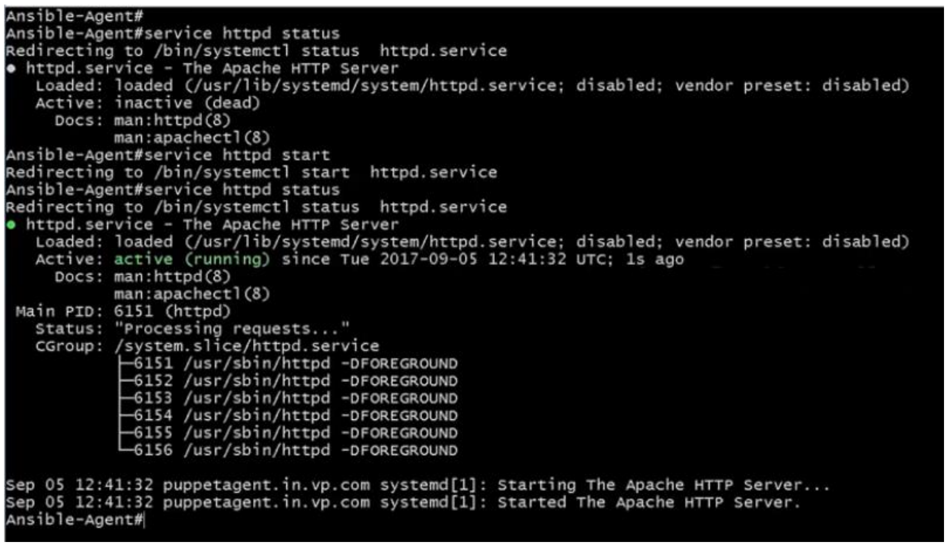
Run it:



```
Ansible-master#ansible-playbook sample.yml

PLAY [webservers] ***********************************************************************

TASK [Gathering Facts] ******************************************************************
ok: [agent]

TASK [install http package on agent1] ***************************************************
ok: [agent]

TASK [start the service on agent] *******************************************************
changed: [agent]

TASK [copy the template] ****************************************************************
changed: [agent]

PLAY RECAP ******************************************************************************
agent                      : ok=4    changed=2    unreachable=0    failed=0

Ansible-master#
Ansible-master#
Ansible-master#
```

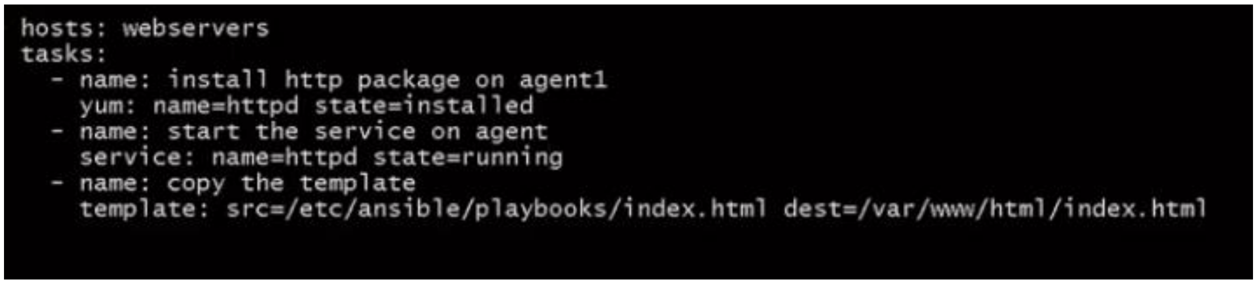
Service http status

Service http start



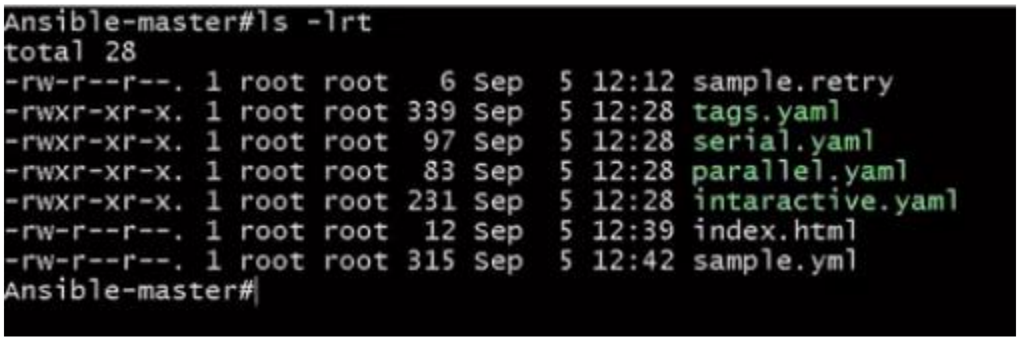
```
Ansible-Agent#
Ansible-Agent#service httpd status
Redirecting to /bin/systemctl status  httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:httpd(8)
           man:apachectl(8)
Ansible-Agent#service httpd start
Redirecting to /bin/systemctl start  httpd.service
Ansible-Agent#service httpd status
Redirecting to /bin/systemctl status  httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled)
   Active: active (running) since Tue 2017-09-05 12:41:32 UTC; 1s ago
     Docs: man:httpd(8)
           man:apachectl(8)
 Main PID: 6151 (httpd)
   Status: "Processing requests..."
   CGroup: /system.slice/httpd.service
           ├─6151 /usr/sbin/httpd -DFOREGROUND
           ├─6152 /usr/sbin/httpd -DFOREGROUND
           ├─6153 /usr/sbin/httpd -DFOREGROUND
           ├─6154 /usr/sbin/httpd -DFOREGROUND
           ├─6155 /usr/sbin/httpd -DFOREGROUND
           └─6156 /usr/sbin/httpd -DFOREGROUND

Sep 05 12:41:32 puppetagent.in.vp.com systemd[1]: Starting The Apache HTTP Server...
Sep 05 12:41:32 puppetagent.in.vp.com systemd[1]: Started The Apache HTTP Server.
Ansible-Agent#
```

Change the state into running



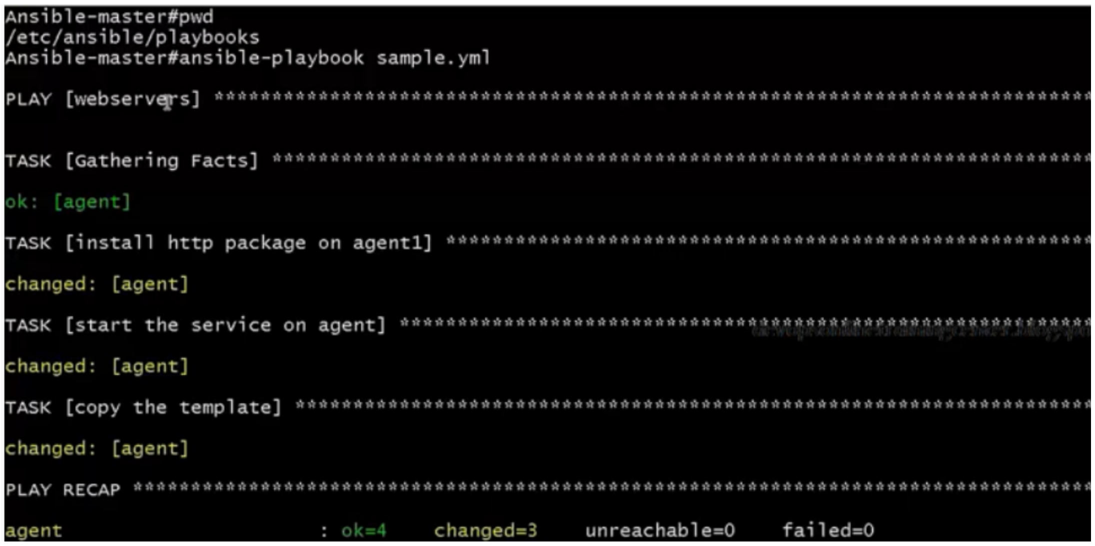
```
hosts: webservers
tasks:
  - name: install http package on agent1
    yum: name=httpd state=installed
  - name: start the service on agent
    service: name=httpd state=running
  - name: copy the template
    template: src=/etc/ansible/playbooks/index.html dest=/var/www/html/index.html
```

Copy to index.html



```
Ansible-master#ls -lrt
total 28
-rw-r--r--. 1 root root   6 Sep  5 12:12 sample.retry
-rwxr-xr-x. 1 root root 339 Sep  5 12:28 tags.yaml
-rwxr-xr-x. 1 root root  97 Sep  5 12:28 serial.yaml
-rwxr-xr-x. 1 root root  83 Sep  5 12:28 parallel.yaml
-rwxr-xr-x. 1 root root 231 Sep  5 12:28 intaractive.yaml
-rw-r--r--. 1 root root  12 Sep  5 12:39 index.html
-rw-r--r--. 1 root root 315 Sep  5 12:42 sample.yml
Ansible-master#
```

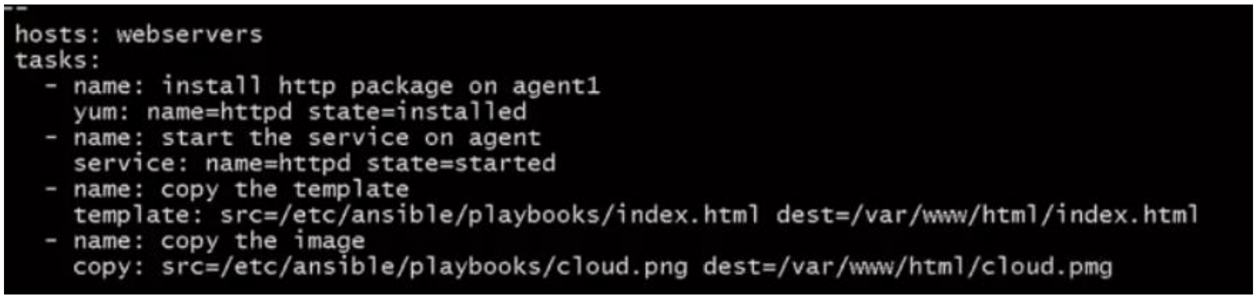
Run the process



```
Ansible-master#pwd
/etc/ansible/playbooks
Ansible-master#ansible-playbook sample.yml

PLAY [webservers] ***********************************************************************

TASK [Gathering Facts] ******************************************************************

ok: [agent]

TASK [install http package on agent1] ***************************************************

changed: [agent]

TASK [start the service on agent] *******************************************************

changed: [agent]

TASK [copy the template] ****************************************************************

changed: [agent]

PLAY RECAP ******************************************************************************

agent                      : ok=4    changed=3    unreachable=0    failed=0
```

Display image on website

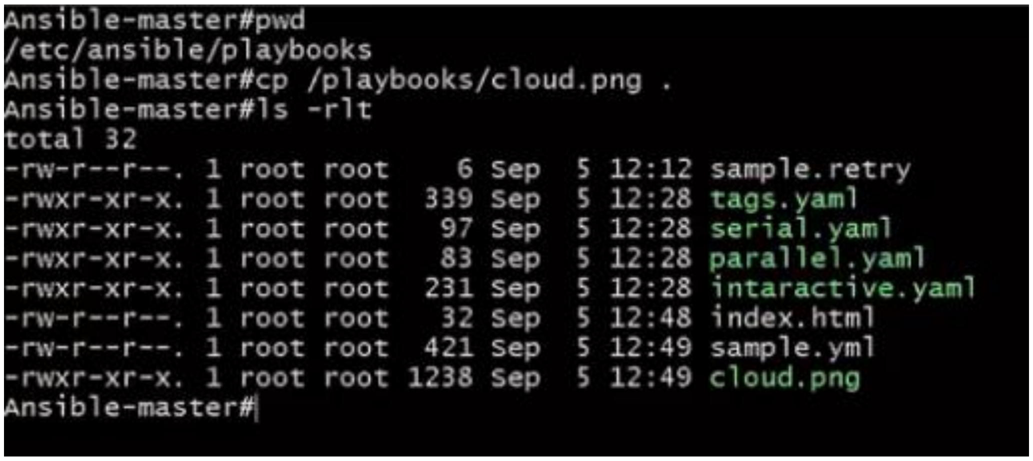
Copy the image



```
--
 hosts: webservers
 tasks:
   - name: install http package on agent1
     yum: name=httpd state=installed
   - name: start the service on agent
     service: name=httpd state=started
   - name: copy the template
     template: src=/etc/ansible/playbooks/index.html dest=/var/www/html/index.html
   - name: copy the image
     copy: src=/etc/ansible/playbooks/cloud.png dest=/var/www/html/cloud.pmg
```

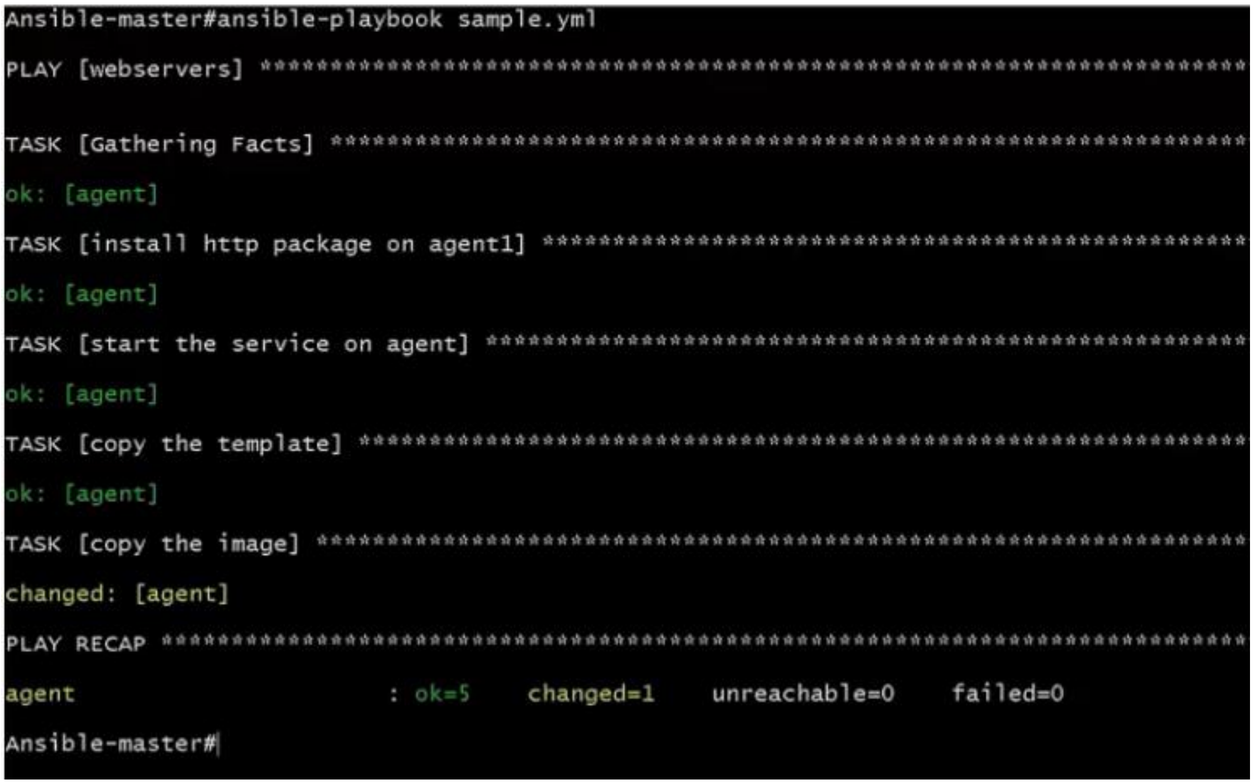
It will be available in the master.

Now it has to copy to the agent.



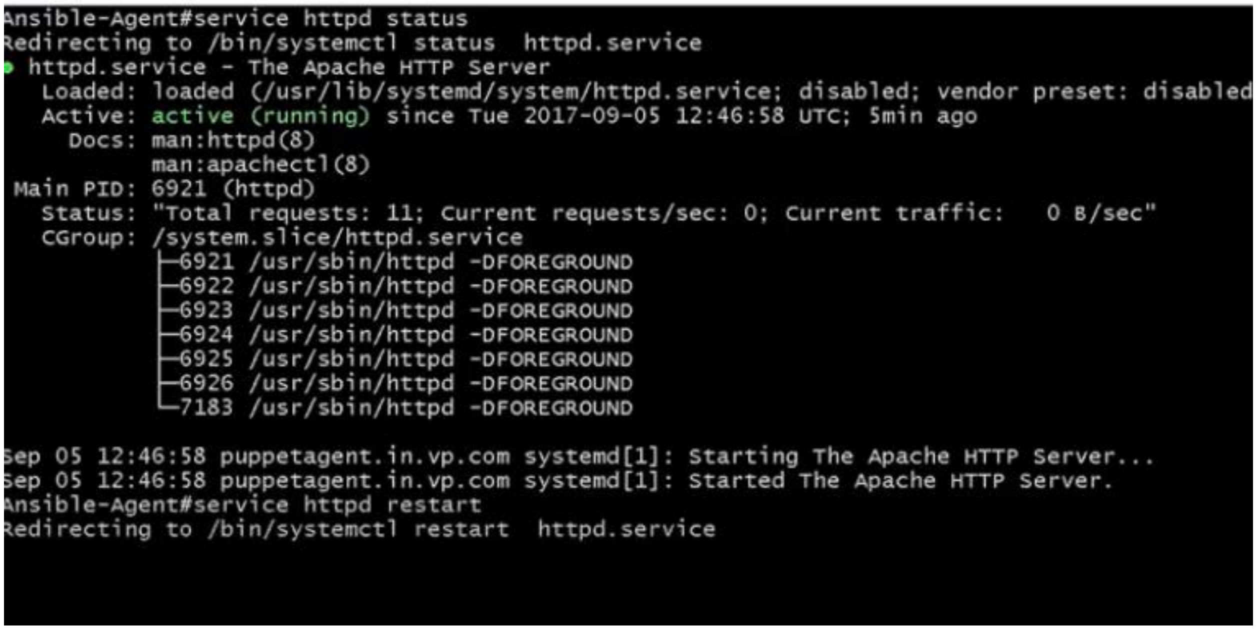
```
Ansible-master#pwd
/etc/ansible/playbooks
Ansible-master#cp /playbooks/cloud.png .
Ansible-master#ls -rlt
total 32
-rw-r--r--. 1 root root    6 Sep  5 12:12 sample.retry
-rwxr-xr-x. 1 root root  339 Sep  5 12:28 tags.yaml
-rwxr-xr-x. 1 root root   97 Sep  5 12:28 serial.yaml
-rwxr-xr-x. 1 root root   83 Sep  5 12:28 parallel.yaml
-rwxr-xr-x. 1 root root  231 Sep  5 12:28 intaractive.yaml
-rw-r--r--. 1 root root   32 Sep  5 12:48 index.html
-rw-r--r--. 1 root root  421 Sep  5 12:49 sample.yml
-rwxr-xr-x. 1 root root 1238 Sep  5 12:49 cloud.png
Ansible-master#
```

Refresh the page and run it



```
Ansible-master#ansible-playbook sample.yml

PLAY [webservers] ******************************************************************************

TASK [Gathering Facts] *************************************************************************
ok: [agent]

TASK [install http package on agent1] **********************************************************
ok: [agent]

TASK [start the service on agent] **************************************************************
ok: [agent]

TASK [copy the template] ***********************************************************************
ok: [agent]

TASK [copy the image] **************************************************************************
changed: [agent]

PLAY RECAP *************************************************************************************
agent                      : ok=5    changed=1    unreachable=0    failed=0

Ansible-master#
```

Check httpd status



```
nsible-Agent#service httpd status
edirecting to /bin/systemctl status  httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; vendor preset: disabled
   Active: active (running) since Tue 2017-09-05 12:46:58 UTC; 5min ago
     Docs: man:httpd(8)
           man:apachectl(8)
 Main PID: 6921 (httpd)
   Status: "Total requests: 11; Current requests/sec: 0; Current traffic:   0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─6921 /usr/sbin/httpd -DFOREGROUND
           ├─6922 /usr/sbin/httpd -DFOREGROUND
           ├─6923 /usr/sbin/httpd -DFOREGROUND
           ├─6924 /usr/sbin/httpd -DFOREGROUND
           ├─6925 /usr/sbin/httpd -DFOREGROUND
           ├─6926 /usr/sbin/httpd -DFOREGROUND
           └─7183 /usr/sbin/httpd -DFOREGROUND

ep 05 12:46:58 puppetagent.in.vp.com systemd[1]: Starting The Apache HTTP Server...
ep 05 12:46:58 puppetagent.in.vp.com systemd[1]: Started The Apache HTTP Server.
nsible-Agent#service httpd restart
edirecting to /bin/systemctl restart  httpd.service
```

Steps:

ssh

Install ssh

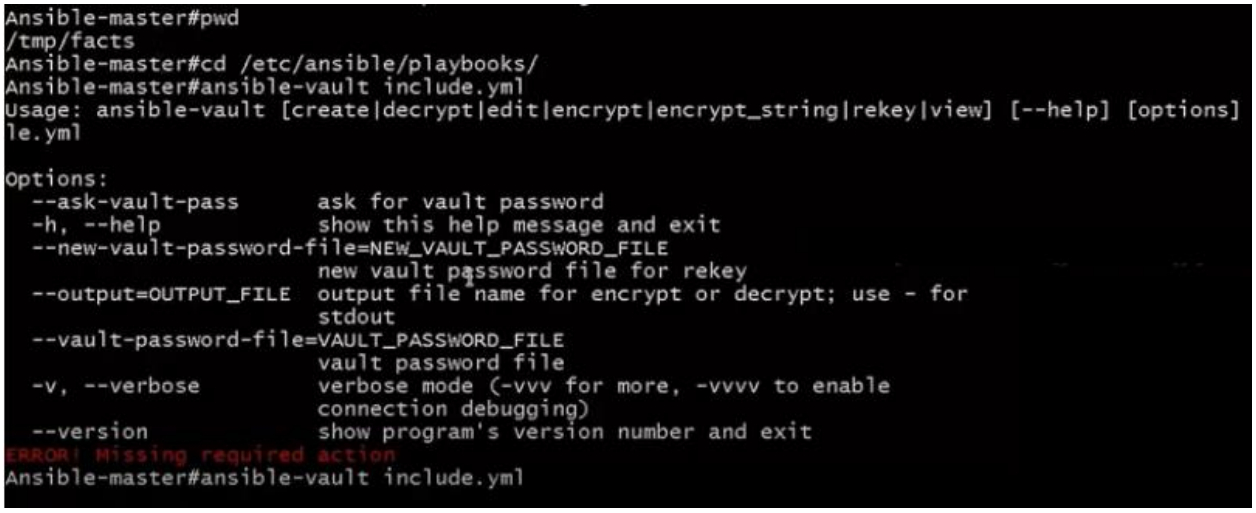
Start sshd

Copy /etc/ssh/sshd_config

Restart ssh

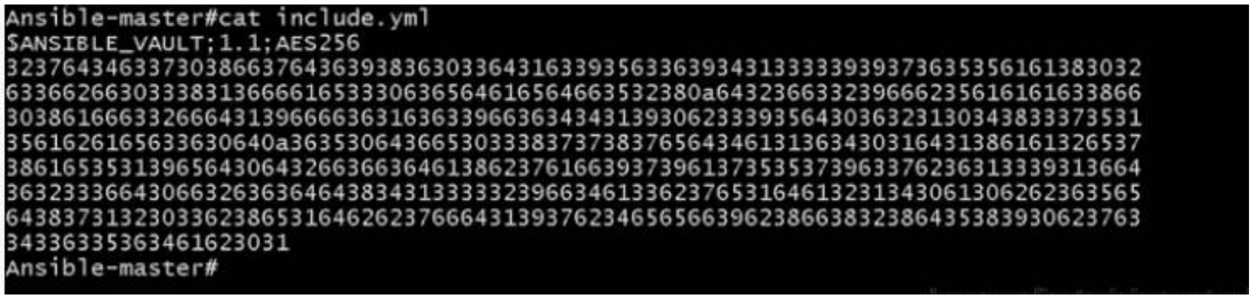
SECTION 15

you are using some confidential information in the playbook or you have written a Playbook and you don't want anybody else to see it or execute it so I want to restrict people from executing are seeing the Playbook it's so how we do that so basically want to protect the playbook so there is a component called ansible-vault this basically encrypts the Playbook with a password if someone wants to see they cannot see because the Playbook is encrypted and if someone to execute also it will ask the password so it will ask to set a password we can set the password so only that person who knows the password can be able to execute it others cannot executed it and while executing the Playbook it decrypts and executes on the fly this is called ansible vaults basically for security and if you have some confidential information in the Playbook we can restrict people from accessing it using ansible vault component.

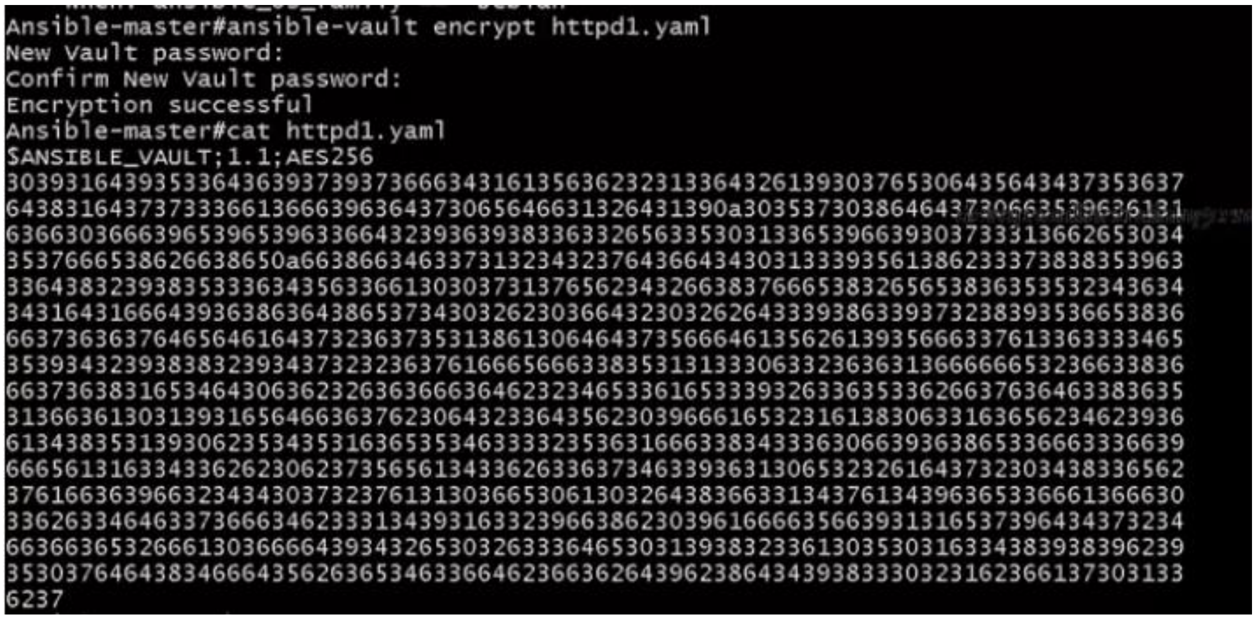


```
Ansible-master#pwd
/tmp/facts
Ansible-master#cd /etc/ansible/playbooks/
Ansible-master#ansible-vault include.yml
Usage: ansible-vault [create|decrypt|edit|encrypt|encrypt_string|rekey|view] [--help] [options]
le.yml

Options:
  --ask-vault-pass      ask for vault password
  -h, --help            show this help message and exit
  --new-vault-password-file=NEW_VAULT_PASSWORD_FILE
                        new vault password file for rekey
  --output=OUTPUT_FILE  output file name for encrypt or decrypt; use - for
                        stdout
  --vault-password-file=VAULT_PASSWORD_FILE
                        vault password file
  -v, --verbose         verbose mode (-vvv for more, -vvvv to enable
                        connection debugging)
  --version             show program's version number and exit
ERROR! Missing required action
Ansible-master#ansible-vault include.yml
```

If you open the playbook, it will be Encrypted



```
Ansible-master#cat include.yml
$ANSIBLE_VAULT;1.1;AES256
32376434633730386637643639383630336431633935633639343133333939373635356161383032
6336626630333831366661653330636564616564663532380a643236633239666235616161633866
30386166633266643139666636316363396636343431393062333935643036323130343833373531
3561626165633630640a363530643665303338373738376564346131363430316431386161326537
38616535313965643064326636636461386237616639373961373535373963376236313339313664
36323336643066326363646438343133333239663461336237653164613231343061306262363565
64383731323033623865316462623766643139376234656566396238663832386435383930623763
3433633536346162303 1
Ansible-master#
```



```
Ansible-master#ansible-vault encrypt httpd1.yaml
New Vault password:
Confirm New Vault password:
Encryption successful
Ansible-master#cat httpd1.yaml
$ANSIBLE_VAULT;1.1;AES256
30393164393533643639373937366634316135636232313364326139303765306435643437353637
6438316437373336613666396364373065646631326431390a3035373038646437306635396 36131
63663036663965396539633664323936393833633265633530313365396639303733313662653034
3537666538626638650a66386634633731323432376436643430313339356138623337383835396 3
33643832393835333634356336613030373137656234326638376665383265653836353532343634
34316431666439363863643865373430326230366432303262643339386339373238393536653836
66373636376465646164373236373531386130646437356664613562613935666337613363333465
35393432393838323934373232363761666566633835313133306332363631366666653236633836
66373638316534643063623263636663646232346533616533393263363533626637636463383635
31366361303139316564663637623064323364356230396661653231613830633163656234623936
61343835313930623534353163653534633332353631666338343336306639363865336663336639
66656131633433626230623735656134336263363734633936313065323261643732303438336562
37616636396632343430373237613130366530613032643836633134376134396365336661366630
33626334646337366634623331343931633239663862303961666635663931316537396434373234
66366365326661303666643934326530326333646530313938323361303530316334383938396239
35303764643834666435626365346336646236636264396238643439383330323162366137303133
6237
```

Commands:

ansible-vault encrypt sample.yml=> to encrypt

ansible-vault decrypt sample.yml=> to decrypt

ansible-playbook sample.yml --ask-vault-pass=> to execute

Common tasks to be performed

Approach 1

1.Webservers

- Install ssh
- Configure ssh
- Install python
- Install perl
- Http.yml code

2.dbservers

- Install ssh
- Config ssh
- Mysql code

3.Appservers

- Common task
- Application xyz

Approach 2

common.yml=> perform common tasks

1.webserver=> include common.yml
http.yml

2.dbserver=> include common.yml
mysql.yml

Approuch3
Webserver
http=>
vars/main.yml=> http_port=8080
main/task.yml=>{{http_port}}
templates/index.html
files/cloud.png

ANSIBLE ROLES

Webserver→role→install commontasks/ssh install http/configures
/etc/ansible/playbook/role1.yml

```
                    Hosts: webservers
                    Role: role1
/etc/ansible/roles/webservers/vars
                                  /main
                                  /templates
                                 /files
```

SECTION 16

Create a user on all agents:

Requirement:

Home directory

Username

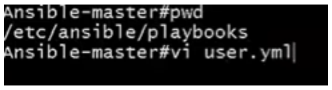
Group name

Password

Shell etc.

Process:

Step 1



```
Ansible-master#pwd
/etc/ansible/playbooks
Ansible-master#vi user.yml
```

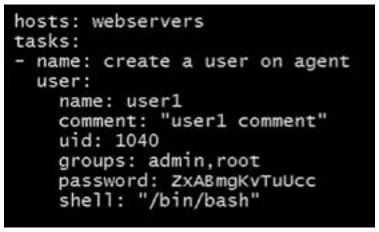
Step 2

Password creation

openssl password => to generate a encrypted password for user

Step 3

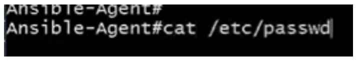
Add Host and task



```
hosts: webservers
tasks:
- name: create a user on agent
  user:
    name: user1
    comment: "user1 comment"
    uid: 1040
    groups: admin,root
    password: ZxABmgKvTuUcc
    shell: "/bin/bash"
```

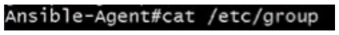
Step 4

List out all the users created in the linux machine



```
Ansible-Agent#
Ansible-Agent#cat /etc/passwd
```

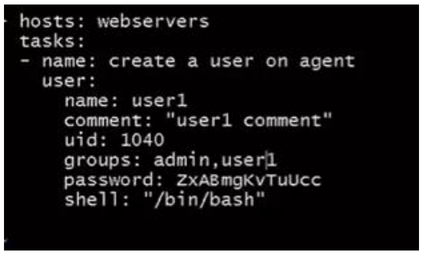
List all the groups created



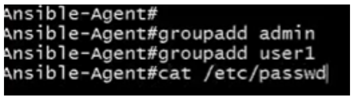
```
Ansible-Agent#cat /etc/group
```

Step 5

Create a group

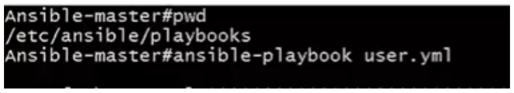


```
- hosts: webservers
  tasks:
  - name: create a user on agent
    user:
      name: user1
      comment: "user1 comment"
      uid: 1040
      groups: admin,user1
      password: ZxABmgKvTuUcc
      shell: "/bin/bash"
```



```
Ansible-Agent#
Ansible-Agent#groupadd admin
Ansible-Agent#groupadd user1
Ansible-Agent#cat /etc/passwd
```

Indentation



```
Ansible-master#pwd
/etc/ansible/playbooks
Ansible-master#ansible-playbook user.yml
```

Handlers:

Written a playbook that has:

Install http package

Start the service

Copy the template

Copy the file etc.

After coping jar file or var file some application specific data we need to restart the process. Whenever there is a change then only restart. For that we use handlers. It will only executed only if any change in content etc.

SECTION 17

Ansible

Ansible roles:

On the agents

ntp=> network time protocol

1) Visit galaxy.ansible.com
2) Go through the readme=> on which platform its supported
3) Ansible-galxy install<>=> /etc/ensible/roles
4) Playbook to associate role with hosts

   How do we write a role

```
/etc/ansible/roles/ntp/tasks/main.yml
                       vars/
                         defaults/
                         handlers/
                         templates/
                          files/
                         meta/=>readme
                          tests/
```

Source control/ version control systems:

Why do we need a version control system?

For to maintain the versions.

It can be for ppt, excel sheet etc

Track the changes

Version control systems:

Clearance

Svn

Git

Git/github:

Github is the centralized server where we store our source code.

Its an open source.

Git is just a tool to download the code and to upload the code.

90% of the open source code is on git.

Git bash is cli tool to download/upload code

Git=> help page

Git works on distributed

Whenever we upload/modify the code or file their version will be maintained as v1.0, v2.0

Git clone=> download entire code to laptop/desktop

Git add=> adds files to index area

Git commit=> adds to staging area

SECTION 19

Maven is a built tool which generates artifacts from the source code.it is used to build project. It depends on xml configuration which is project object model (POM).

Installation code for Maven is yum install apache-maven.

Artifact is a combined version of source code.

Phase is collection of goal in which task is performed.

"helloworld".jar is package in Maven. Clean, initialize, generate resource, built, Validate, test, package, deploy, process are some phases of build lifecycle of Maven. Deploy is used to store binaries.

Clean is used to clear repository.

If we execute any goal, it will be updated from beginning.

Plugin is a feature used to execute some plugin.

Ant and Maven is different because of the way in which we download dependencies. In ant we download dependencies manually and in Maven it is downloaded automatically. All downloads are kept in a local repository. There are two type of repository: Maven repository and local repository. Local repository is kept in maven home directory.

Maven code is wrote on POM.xml .It Include all dependencies. It is provided by developer. eg: <dependecies> ....</dependencies>

Maven is writen on basis of what all branches a project have. It is similar to Github repository. Some of them are Master, Development, and Feature. Feature branches can have multiple branches in which developer work on each day and pushed to feature branch; which is checked and merged to development branch. Master branch is stable used for deploying production.

Master branch will be a deployable software. it can have jar, war, ear, zip file format.

POM.xml is executed in two different ways. In Interactive way, when we execute we need to give projected ID, group id, version, artifactid etc.

In feature branch we will built and have joint test; while in develop we built, unit test, package, deploy, and perform testing. In master branch along with all this we do release which make sure development is fine. mvn release can update the versions.

SECTION 20

Jenkins is continous integration opensource server which devoleped by sun(employee of microsyste).
Jenkin is previosly named as hudson which is owned by oracle.
It is framework which integrate tools like github, maven, nexus, ansible e.t.c to deploy software.
It hase some feature(plugin). It has master slave architecture.
It can run in master mode and stand alone mode. it can be implimented in organization wide.
Jenkin master have access all its slave. Slave performs the task.
Jenkinks detects the change periodically and configure the changes.
By default it listens on port 8080, which we can change manually if needed. It is written in Java.
Jenkin have its Own repository
If all the basic configuration of jenkins is completed, It hook all too to jenkin product and execute based on build schedule.
With the command jenkin start it will start the service and by the command Jenkin Status we can check whether it is active or not.
Password for jenkin will be in our machine.
At fisrt it will show plug in to install in which we can install all suggested / frequently used pgin or we can install select plugin manually to install.
It will ask to provide essential information.
Manage jenkins is used to configure Jenkin. Congigue global security is to have security. we join Jenkin with organization authorized email-id. Matrix based security is used to have specific permission.
By defauly used id of jenkins will be jenkins and all command will be executed as Jenkin user.
We need to change bin/false to bin/bash if we nned to login as jenkin user.
There will be a file called /var/lib/jenkins/config.xml to configure file for jenkins. Here we can change user security to false which allow us to login without security configure.
Freestyle project allow us to integrationg source code managment and execute it. Build now will build the project. and we cann access it with console output.
We can configure the builded project and edit it. hence commands are given in command shell.
Plugin Manager show us the available plugin and its updates. If we need to download a plugin we need to restart Junkins. If we need to add plugin, We can access it in built step.
We can devolep menu's using Junkins which help us to build with parameter.
Build with parameter help us to give parameters which perform some operation.
Blue indicate sucess, redindicate Failure, whether(yellow) shows status.

SECTION 21

Build with parameter take the IP of machine and take the patch we want to apply and it takes distribution as input. We can write shell script based on requirment.

Any thing we can automated with Jenkins.

If devolepment team want to conduct some test cases, Instead of going to test team devolepment team can have their own setup for testing.

To add a slave to Jenkins; slave will conusertain the task : At first we ned to give authorization to slave to perform task. So we ned to setup ssh key between master and slave to perform task on slave by master. Slave and master could have a proper communication channel. It must generate key on agent and create the user jenkin on agent. Then setup key from master to agent.It help master to perform task without asking the password.

ssh-keygen to generate key and set password for jenkins user on slave.

ssh-copy-id to copyfrom master and agent.

useradd -d "/home/jenkins" -c "comment" -s "/bin/bash" -m jenkins is usedcto create user in jenkins

To setup slave in Jenkins, Go to manage jenkins and MAnage nodes. There we can add new node.Remote root dierectory is dierevtory of Jenkins user. Number of executer refers to number of shared processess on slave. Label refers to agent which project run the project. We need to launch it via SSH.

We need to specify all ssh while adding a slave and we can set primary key from jenkins master~/ssh.

Here when we create the project and we need to restrict where this project can be run, there we can see the slave added.

Master is actually dedicating the work and executing in slave. Slave report the status to master.

Jenkin will automatically start based on resources on slave and start job automatically.

It can communicate with Github by adding Url in source code managment- repository. It will download all the file from github to slave. Maven is used to convert theses to binary. For than we create another Maven project in Jenkins.

we can give maven goal such as mvn test, mvn compile, mvn package, mvn deploy, mvn install.

All binary of maven will be available in target. In POM.xml we can see various plugin, path, goals etc . we can specify module,dependencies of the project etc.

The downloaded binary will be uploaded to nexus artifactory and there we specify timestamp.

SECTION 22

To configure a slave on jenkins master we first create jenkins user on the slave and set the password to slave as root. Password must be less authentication on server and slave for Jenkin user.Then we go to manage nodes, new node select permenent agent and then select the option below.

Nexus is a tool for artifacts which we need to configure on machines. Nexus server will store artifacts.

Nexus will be a zipfile. It is installed on other machine. ./nexus start will start the nexus. By default it will be in port 8081.

nexus is devoleped by sonatype.

Since nexus is intalled seperately, so we need to transfer it into agent, fo that we use scp filename username@ip:/

tar -zxvf will extract nexus to local dierectory.

We craete a softlink which is like a short cut.

If we need to ren nexus as root we need to export the varibale shown there.

nexus is accessed with if config:portnumber/nexus/. By default the password will be admin123 and username will be admin.

In nexus we can store with timestamp, we can have clear dierectory structure.

partprobe is a command to select all devices. fdisk is utility. Using these we can create a apartition.

To configure the repository, From jenkin all the binary are dierectly stored in nexus server.

maven deploy artifacts to nexus.

The diffrence between snapshot and release is snopshot is created binary with devolep branch and release is which we create from master branch.

Syntax of snapshot will be application.version.timestamp-snapshot.zip

release syntax will be application.version.zip

release refers to final version.

we need to install plugin such as nexus and timestamp. Then we configure our project and select build timestamp and give date and time pattern.

artifact id will be build_timestamp.

Buildid refers to bulid number of the job.

We can use nexus artifact uploader and give nexus url. In credential we need to add username and password. we need to specify the path of workspace.

As soon as job/task is started, it will convert source code to binary , Maven is triggered, nexus is trigered to store binary on nexus repository.

Pooling is done by Jenkins and each task is done by jenkin sequentially.

SECTION 23

***** indicates that it will run on every minute and 5**** indicate that it will run for exeny 5 minute, second star denote the time in 24 hrs(* 13 * * * *). If there is a change in commands it will create a new job. It can be given in scedule on build triggers.

Workspace willbe Overwritten and Binaries will be available, because it is be on basis of timestamp.So if build goes wrong if can go and execute previous binary.

ansible plugin is a path to which playbook is present. We add credential on it.

While polling if it detects any changes, It creates a job in Jenkins and then download source code to jenkin workspace(slave). It call maven and based on goal it will execute. This goal will be converted to binaries. It is deployed to nexus. Copies of binary will also be kept to test node.

ansible master can be put in Jenkin slave. It will be copied to test node. ansible have the binaries.

Copy module will copy file from jenkin slave to ansible test node.

To copy the binaries to test node, we need to change the tast that is we need to mention copy the War file from server to test node and we need to mention the masters path. It will create a new project and we need to give project name as we given in Jenkins and target and the path.

We can remove nexus because here we dierectly take it from machine.

In Pipeline we write everything in code itself insted of using GUI. FOr this we use groovy scripting.

Everything that we do in Jenkin using plug in can also be done using code. It is pipeline. It is advanced problem on Jenkins.

In pipeline we first need to select the slave. The code are written in Jenkins pipeline script

Consider a sample code(groovy):

```
node('Slave1')
{
stage'github Download'
git url://http://...., CREDENTIAL: abc
stage'execut_test'
sh "mvn test"    (sh refers to execute shell command from groovy)
stage'Package'
sh "mvn package'  (allow diffrent stage and shows a pipeline)
stage 'deploy'
sh "mvn deploy'
stage 'executeansible'
sh "ansible-playbooks abc.yml'
}
```

SECTION 24

The problem with free style project is when Jenkins is scratched, We need to restore everything. although it will be a dificult task. Pipeline solves this issue.

Pipeline can maintain the code. Here we write code to perform the job. We can store thee code in repository, we can maintain and modify it.

There are diffrent stages in pipeline where in each pipeline, where in each stages like diffrent phases in project.

groovy codes are saved in source-code managment repository.

There is option to dierectly give pipeline script in Jenkins and there is also an option to give from pipeline in SCM, we need to write groovy in a file called Jenkinsfile. This jenkin file can be saved in github and we give repository information in pipeline

In groovy code with the node we can specify the branch where it is to be executed(master/slave). If we leave it empty jenkins will pickup automatically based on resource.

In pipeline we need to specify th branch While in multi branches pipeline in which jenkins automatically detects the branches present in the repository.

Mainly a Project will have Master,devolep and feature branches.

Devolep braches are kept in articfact FOr that we mvn clean, test, Package, deploy.

From the repository, we create a Jenkins file and we need to define a architecture.

Jenkins have some inbuilt variable such as build_id, build_url, build_status, build_branch.

eg :

```
node('slave100')
{
if ( env.BRANCH_NAME == 'master' )
   master_branch()
else if (env.BRANCH_NAME == 'devolep')
   devolep_branch()
else
    feature_branch()
```

each function is fefined in groovy

eg:

```
def master_branch()
{
stage('download code')
checkout scm    (to download the code)
.
.
.
}
```

notifyBuilt() is used to send a mail which contain succes/failure. emailext(), rublish reoports, html, junit are other plugin used to send mail which is published on jenkins after completion of task.
In multibranch pipeline we need to provide the code from source code managment(Scm) itself. That is we need to give github url and credential.
Multibranch pipeline will execute on the basis of branch. We need to make sure that every branch should have jenkins file.
If there happens any changes in any branches it will automatically built using poling.
If a failure happen, we can easily findout where it happens.
Checkpoint is used to show what is happening and approval is used to specify manual approval. It is only available in Multibrach pipelines.
eg for approval:
stage 'approval'
input 'Do you want to proceed'
There is a plugin named thinbackup which take job level back up. It is at job level. It is used when a job is deleted or modified by mistake. It will be available in manage jenkin if plugin is downloaded and activated. We need to specify the backup dierectory. It will create backup based on the timestamp.
Another method of taking of backup is taking the entire dierectories. If the master is scratched we need to built another server then copy this entire dierectory to new server. Then it will automatically restart all the job configurations.
eg: tar -cvf jenkins-20171909.tar /var/lib/jenkins   (it will create one file with this dierectory)
    gzip jenkins-20171909.tar --> jenkins-20171909.tar.gz (here we copy file to another server)
So incase if jenkin scratches we can install jenkins master again and we can copy this file and extract it.
To extract: gunzip jenkins-20171909.tar.gz
            tar -xvf jenkins-20171909.tar
Every job will have a config.xml. Jenkins store all information about that job in particular config.xml . We can use this also asa particular backup.

SECTION 25

In earlier we use to have on OS running on single machine. So there was under usage og memmory and CPU.

To run multiplication on a Single machine, VMWARE introduced virtual machine. Here on top of hardware, they created a software ehich is known as hypervisor. It virtualize the hardware to create multiple independent workstation. This is known as VM. SOme of example of Vm are VMWARE, xen e.t.c

In Docker we have hardware and on top of hardware we have OS, and on top of OS we have container. Instead of virtualizing the hardware we virtualize the OS here. In container application will run. Docker engine is responsible for sharing resource between container.

Main use case of Docker is when a devoper devoleped a feature and deployed it to customer. It can show an issue because of missing of binaries or libraries. To avoid this we can bundle all the binaries with fix and send it to the production. It is knoen as docker image.

Another use case is , If we need to transfer binary from machine 1 to machine 2 we upload the image to docker cloud and transfer it. Image contain all the fixes. Docker cloud is repository of images.

Container will only have limited set of file set which is required for application to run. Container can be considered as a light weight virtual machine.

Namespace isolation is used for isolating the resource between diffrent container.

Microservices are run on a container. Microservices are diffrent part of an application.

yum install docker* is command used to install docker. If it is intalled we can use commands like docker --help, docker --version command to help and see the version.

To create a container, we need to have an image. so we ese : docker pull ubuntu , which download image from docker cloud.

docker run will create a container on image. hence container is a process running from image. docker ps will give running container on machine.

docker attach/detach can connect to a container.

docker rmi<image id> is used to remove the image.

docker run ubuntu command will be dierectly attatched to terminal of docker container and docker run -d ubuntu will run outside shell which is a detatched mode.

service docker start is the command used to start the docker.

By default when we run a docker we need to attach a shell to it. If we dont attach anything it will not run. docker run -t -i is used to run ubuntu will run the container.

We can customize the container which means we can add and package/application/files e.t.c

docker run -d -t ubuntu:latest is used to run ubuntu will run the container on detach mode.

app-get install package_name is used to intall package in docker.

In attach mode if we comeout of container , by default container will stop.

ps-a is command used to show the stopped container.

SECTION 26

To create an image in docker we basically get the base Image using docker pull ubuntu:lastest.
To install Jenkins container in docker, we use the command docker run -i -t -p 8082:8080 Jenkins. here 8082 is host and 8080 represent container port, P represent the port. It will pull the container by default. Password will be provided there. To access it we need to use it https://dockerhost:8080. Anybody can access the container using ip-adderss of host. We need to give password in JEnkins. We donot connect dierect to container ip , we connect to host ip where container is installed. From this we do port forwarding which means redierecting obligation which is running on the container. Any body can access the container using the ip address of the dockerhost.
There can be mutiple container which can handle web traffic. Jenkin will run on container. All the logs will be redierected to host, not to container. We can create freestyle project on Jenkins.
If we need to install plugin here, It will be done on container.
All the susequent container will have ip-address starting from 172.17.0....
Similary we can also run databases. FOr that we use docker run --name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag , where my-secret-pw represent password which is set for mysql root user.
nginx is webserver similar to apache, It can handle web traffic more when we compare it to apache. It is also available in docker cloud. docker run
After getting the base image we need to costomize the page using apt-get update for updating or we can sinstall some packages or if need to do some configuration changes or if we need to update some file.
docker run nginix:latest is the command used to download nginix server.
Nginix is a free open source web server by apache.
Docker start container id is used to start the container. hence we can access it using ip.
docker run --name webserver1 -p 8083:80 nginx:latest. 80 is default port on which nginx runs. This should create container and incoming webtraffic is dierected from port 8083.
If there occurs a hardware problem and container on particular web server are scratched, We make sure that website is always running using cluster. It is handled by docker swarm.
Then we can create container ,If container is ready we need to create image out of the container again. This we can upload it to docker cloud such that anyone can download this image and create container out of it which have all the costomized packages.
We can create our own image and we can push it to the dockerhub. For that we need to signup to dockerhub. We can download the image using pull command in the dockerhub.

SECTION 27

To build a base image and configure and we create our own image and will push that image to docker hub.
we pull the image using docker pull ubuntu:latest.
Then we create container using docker run -t -i ubuntu:latest
To perfornrm costomization we use apt-get update/python/apache2.
New layer will be created for every installation.
For giving the information to customer, we need to convert container to image.
We then pushe image to docker hub. To check if it is done successgully we download the image again and we'll create the container out of it and check all the packages are present in it.
To create a new container withe name we run the command docker run --name ubuntu1 -i -t ubuntu:latest will create container named ubuntu1. Then we install ruby, apache, python e.t.c on the container.
By using container id we create image which is customized to upload to docker hub.
docker commit -m "ubuntucustomimage" -a "author_id" container id path_from_which _we_want_to_generate_containerid/customimage:v1
To push this image to docker hub, we need to login and push the image to docker hub, For that we use command docker push user_name/image_name:version of image(v1). Once it is pushed we remove it locally(docker rmi image_id) and download it(Pull) and create container out on this image(docker run -i -t path) and check whether the all packes exist or not.
If we need to configure the file again we need to do the all process again, So to automate this process we use Dockerfile using some keyword(FROM,RUN,CMD,EXPOSE,ADD..).It is the automated way of builting costomized image.
We use command vi Dockerfile to create a docker file, In that dockerfile we customize.
eg of Docker file:

```
From ubuntu:latest
RUN apt-get update && apt-get install -y ruby
RUN apt-get install -y python
RUN apt-get install -y apache2
RUN service start
```

dockerbuilt -t name_of_image . will run docker file, here . represent the dockerfile is in cwd.
To create cluster of docker hosts, suppose if we have two machine, If one goes down the another will have to take care of the process.
To create multiple container based on load we have docker swarm. To configure docker swarm we create cluster between 2 physical machine.
To create cluster between two machine, Suppose if one machine is manager node and other is worker node , we need to execute some command on manager and some on worker to create the cluster. Manager node will automatically create the container on worker based on load, If any worker goes down manager moves that load to another worker.
By using docker service command we can scaleup n number of container.

SECTION 28

Basically a cluster is a collection of 2 or more nodes or machines or server. It provides high availability, reliability and fail over.

High availability refers to something like less than 5 minute down time in a year. If one server goes down another server will take over the responsibility. This process is known as reliability.

There is a cluster software taking care of which node is down, checking of which node is up, checking of hard beat which send message and check which node is alive. High availability make sure that every node is up all the time.

In case of docker container we achieve high availability by docker swarm. HEre if one container goes down another comes automatically.

Suppose we create cluster of 3 physical machines, and within that cluster we create nginix webserver which should be up all the time. We scale up and check how many and where the containers are created. Scale up increase the infrastructure.

Here we shutdown one of the node and we check how containers are moving from one node to another node.

To see running containers on a node, we use dr.Pearson and check if it show more number of containers or not.

Manager node will contol all the container and workers.

export PS1=swarm manager will make manager

export PS1=Docker-worker1 will make worker1

export PS1=Docker-worker2 will make worker2

Manager can also run the work because the container also will run on the manager.

By default docker version will be 1.1.2 or more and they should be communicating with each other and they should be in same network and some port should be open.

Then initialize cluster on manager, we use the command: docker swarm init --listen-addr address:2377 --advertise-addr address. It will start broadcasting this address and generate a token. This token can be used on slaves to connect to manager.

To add worker to the manager node, we run the key on slave/worker node. Then we need to start it.

Then we create a service with nginix server using following command: docker service create--name website--publish 80:80 sexeyed/docker-swarm-walkthrough. It is a container image available on dockerhub that will be available for download and use it.

docker pull sexeyed/docker-swarm-walkthrough will pull the image which has weserver congigured in it and if i create a container out of this it will have webserver which is already configured.

Hence container is created in manager node, For accessing the container on manager by worker we can use ip address. We can access it because it also has the property of load balancing.

If we create 10 or more number of container in manager , It will automatically share the load: that is some container will automaticall run on workers node.

If master goes down we can promote slave as master .

To scaleup how many containers are created we use the command: docker service update --replicas 20 website.
docker node ls will check the status of node.
Load balancer is a software that redierects the traffic to diffrent instances running on multiple machines. Here DNS will do this work.
Some of the clustering services are mesos, google kubernetis.

SECTION 29

For a company they need hardware, OS, SOftware, human resources, power supply, AC, Space etc. This is known as on premise dta center.

Instead of this now we use cloud computing, Which have a cloud provider Which take care of every thing. Cloud computing is basically getting the resources from internet. As of our need when we request for n machine and their resources, we will get all resource automatically.

Cloud computing provide horizontal and vertical scalability.

Another feature of cloud computing is Pay as you use. That is it provide post paid payment.

Some of the cloud provider are AWS, AZZURE, google cloud, IBM soft layer, Rackspace e.t.c

There are mainly two type of cloud model, Public and private. Public cloud provide its cloud to every one . Some of the public cloud are AWS, azzure e.t.c . Private cloud provide its rosources to a particular organization only. Since ther have on premise data center. They use a software to manage every thing. Cisco quiker is a private cloud model.

Hybrid cloud is a cloud model which is a combination of public and private cloud. That is thay use public model for devolepment and testing porpose and use private model for production.

Community cloud is a cloud model which is setup only gor a particular community. We can only access to this only using the community's email and password.

Private model consist of openstack which have driver for network, driver for storage.

Various cloud services are IAAS (Infrastructure as a service), PAAS(Platform as a service), SAAS(Software as a service).

Suppose we dont have MSoffice in our system and we try to open a word doc in gmail, It opens because google offer SAAS.

Suppose devoleper devoleped a software in which they dont need to maintain software and hardware and they need to maintain source code. Google cloud engine provide run time for this, It can be considered as platform as a service.

IAAS provide virtual machine and we can maintain, built and create website in VM.

ec2 (Elastic cloud computing) is a basic service of AWS. Here we request for VM wich is known as instances using amazon website. For that we need to register and sign on by providing credit card details. Then we can use ec2 and based on usage they will charge. 720 hours per month they provide free service. They provide a key to access virtual machine and based on it, By using this we work on it.

An architect need to define the server,storage e.t.c and then we give permission for others to access it.

If we launch instance or install vagrant for first time, we have preinstalled image in cloud and when we need to vagrent up we pull the vagrant cloud and it will build image up. Aamzon machine Image is the Preinstalled image in aws. Based on OS we need we can select it.

We can have AMI ceated on our own. We can customize it. If I need a base image ubuntu and I did lot of configuration changes and If I need to spin up an instance, We can do it by AMI.

Tag in instance refers it is free.

Amazon will have data centers geographically located which is called regions /availability zones. If we want high input output performance, It will charged accordingly. Eventhough one region can have multiple availability zone. they are datacenter inside one region.
When we login to AWS, WE can see diffrent services(s3,ec2,dynamodb etc) based on requirmnet. If want high performance, we need advanced performance in storage, Eventhough it is charged. We can create multiple Vm/instances. We can use 30 gb harddisk for free. We can create new volume and it will be attatched as a disk. We can add tag to classify the VM. If we need to access webserver running on machine we give access to http/hhtps which refers to security.
If We need to know whether my instances is up or not, For that porpose we use ICMP. We specify the source for whem where all we can access the instances. Security group defines how do we use our VM.
When we launch aws, We can generate key-pair. Userid and password of ec2 vm will be same.
It is similar to vagrant vm and the diffrence is how do we access and and the security group.
Key will be in .pem format by default and it can be converted to .ppk . we can access gib=tbash using .pem format and putty by .ppk format.
We can convert .pem format to .ppk format using puttygen.
We can see the public ip address after creating the instances and by using it we login to VM using key-pair.
After initializing We can login using following command: ss -i path keypair.pem -l ec2-user ipaddress.
ubuntu will have ec2 instances called ubuntu and amazon linux ami/redhat will have ec2 user.
We can install, customize the packages/application in it using apt-get install ____.

SECTION 30

If we reboot the machine ip of machine will be changed.because it is based on dhcp protocol.

So for static ip we use, elastic ip. If we use more than 5 elestic ip, it will be charged.

We need to assosiate elastic ip with the instances. Then we can able to login with that ip which will not change after the reboot.

IAM refers to identity and access Management service, used for security porpose. Since an owner need to delegate work for workers of particular organization. when an owner creates the aws account he will have unrestricted root access. For giving permission to his worker/admin With restricted access IAM was introduced. Here each set of worker will only have access to limited set task which is assigned by root , worker will be restricted to perform other task.

Here when we create user by providing user name, Here we give AWS managemment console access and Provide password. If we gives the option for auto-generated password we must need to create a new password at next sign in. In permission, we create a new group and We give policy name amazonEC2FULLAccess. Then we assosiate the policy with a new group called linux admin group. Then we well get a new console for linuxadmin(worker). We can send the worker console and one tipe password with user name. When we login we need to create a new password.

We can create n number of user with a certain policy in which user cant access another policy on the same way.

Only root user will have access to all the policy.

We can create our own policy and we can create it from existing policy. we need to use json for creating policies.

R&D account is for testing and devolepment, Production account will aonly have restricted access.

If we need to setup an website, we need to copy the website and we need to make sure that http is opened in security in networking. we need to add http/https with port in inbound rules.

SECTION 31

Simple storage service S3 is one of the service from amazon which is Object oriented. Here we create containers/buckets.
If we need to host a website we need to buy a domain name from sites like godaddy.com .
Unlimited storage* refers that it provide unlimited storage space with condition that it sends a mail that if the storage space in exhausted. To avoid this AMazon comeup with service known as simple storage services.
S3 is a container servive, Eventhough if file in the container have some malware, It doesnt matter. S3 can be considered as conatiner of object. We can do worm operation (write once read many times) and it is usefull for this type of operation. S3 doesn't care about the size of the object.
We can create bucket from managment console from services. We can create bucket, upload files. We can access the file/object with the link provided by bucket.
If we have a server with limited capacity and if we need to extend it to 100 GB we use elastic blovk storage(EBS). EBS volume can be created in ec2 and this volume can attatched to disk. Instances and volume must be in same zone.
If we have multiple storage and if we need to share the storage we use elastic file system(efs).
To check number of disk available we use the command fdisk -l.
Volume can be attatched to single instances by giving the id of instances.
TO refresh the hardware configuration, we use the command partprobe. lsblk is used to see the volume available. we can create partition on the volume.
To extend the file system, we need to create a volume , then create a file system and Mount the file system.
By default if we need to copy the file to instance, we need to loging as ec2-user. An html page should be copied to var/www/html ; this method is used to deploy a website. We can costomize these website easily.
While creating an instance, In advanced detail,we can write a script to add some additional package when machine boots up.

SECTION 32

Auto scaling is used to scale up or scale out based on cpu/memory utilization. Based on user using the particalar website, It has to scaleup instances automatically. We can specify minimum&maximum number of instances. Instances will scale up based on the load. We can set alarm which checks the usage of memory and CPU.

To understood what kind of instances it has to spin up, That is aws what to know what kind of os,software,configuration It wants to use. We define these in a predifined launch configuration. SO based on this it spins up new instances.

We need to create auto scaling from auto scaling, We create a launch configuration and we need to select instances which we need to spin up.

Health check is length of time that auto scaling waits before checking an instances health status.

We can create auto scaling group as initial size. we set minimum and maximum instances, create alarm while creating auto scaling group.

scale out policy is used to grow number of instances. that is if cpu utilization reaches paricular percentage it spin up and start new instances. For that we create alarm

If cpu utilization is less than paricular percentage it destroy some instances. It is scale in policy.

Launch configuration is also created with auto scaling group.

Auto scaling group is attached to load balancer. Load balancer is a software to balance the load across the instances. It is similar to a traffic manager who redierects the traffic to instances based on some algorithm. We configure security group to http.

In load balancer we have something called classic load balancer, Application load balancer and network load balancer.

Classic load balancer is routing of incoming happens at the ip-layer. Here incoming web traffic will be redierected based on ip-address of machine.

In application load balancer routing happens at application level. It is context based routing.

Network load balancer is advanced version of classic load balancer. It can handle millions of webtraffic effectievly.

We can create load balancer publically and for internal to organization. Configure health check will make sure that monitor is up and running. Then we add ec2 instnces which is running.

Created load balancer will show the instances and its status.

While creating auto scaling group we can attach it to load balancer. all the incoming traffic will be rediereced to this instances.

SECTION 33

Curl is used to check whether application is running on particular instances or not. (Curl localhost:80/index.html) . 404 denote server is not healthy and 200 denote server is okey.

If 404 is shown it stops the service.

In healthy threshold it does 10 check in betwwen 10 second and it declares as healthy.

Load balancer will detect the status based on health checks , it will be redierected to instances and if CPU utilization goes beyond it, Auto scaling group will start the instances.

AMI can be considered as pre-defined or docker image In which we create instance from it. Then we login to these instance and we customize,configure the instance. WE can use this customized AMI in launch configuration to spin up the instance.

SECTION 34

RDS is relational database service in aws. It is fully managed in a sence that will not have access to underliner OS. It is highle available. Here we dont have to bother about OS level updates. They provide database like Mysql,oracle db , postgress sql, mongodb, aurora etc.
Aurora db provide 5 times performance more than mysql and it is more costless.
We can access the database from ec2 instance provided we have proper authentication.
If we want high availability, we want database deployed to multi available zone. Suppose if we have data center provided in multiple location and amoung them one of them is primary and another is stand by. Data replication happens betwwen these data bases. That is if primary database goes down, Stand by can act as primary database. It helps to maintain high availability and fault tolerence.
When we create data base it ask whether it is for devolepment porpose (free) or is it for production porpose(Charged).
Multivailability zone are supported only for production.
If we want to write into a database, istead of loading data base with the request we use multiple read replicas. There will be a schema available in database which is required for application to run.
To check a communication is established between ec2 and the database we install mysql/workbench on ec2. TO do that we use following command: mysql -h <database name> -u <> -p <>
We need to remember username and password while creating the database so that we can access it later.
It provide encrypted and backup option.
3306 in the default port in which my sql listen on.
We need to allow inboud security group (MYSQL) so that any instance can access the port using 3306.
MYsql -h <end point of RDS> -u admin -password is command to access database.
DNS (Domain name server) is basically to name website from godaddy/amazon etc. FOr that we need to but the domain and then we need to create ec2 instance and i need to create load balancer which has auto scaling in place. The we define inbound and outbound rule and givs access to http request.
Typically DNS helps to giva a name for the IP. We need to map ip address to domain name.
In amazon route53 we can create and choose a domain and it will be charged accordingly.
Every entries on DNS server will be replicated across all dns server in world and it can be accessible to evey one.

SECTION 35

While adding the server we need to make sure that hostname defined properly. Networking should be proper.

Command wich we use on chef client is chef client and command we use on chef dk is knife and command we user on chef server is chef-server-ct1.

We write the cook books on devolepment kit. In cookbook we use the command chef generate cookbook cookbook_name(apache) to generate a cook book. COmmand are written on JSOM format.

From devolepment kit if we execute knife node list, it will give node, if node is successfully node. There can be any number of node. OPen source chef supports till 25 nodes. After 25 nodes we need to buy node from chef.

Chef is based on pull mechanism. To upload cookbook to serever we use following ccommand: Knife cookbook upload cookbook_name.

Runlist is the assiciation betwwen server and the client. Knife run_list add <client> <cookbook_name> is used to assosiate. Then we execute chef client on the client. Node will collect all information on system.

If we need to add recipies for node, we can add it though command line or we can add ith through GUI.

When we execute chef client from the client, It synchronizes the changes or any cookbook is added will be executed on client.

If we have huge number of client, to automate the pull machanism we use crontab. crontab -l refers to list the jobs. crontab -e refers to edit. We need to create crontab on every client.

Cookbooks/apache/recipes/default.rb is them main file in which cookbook starts. To upload the file we use cookbook/apache(cookbook name)/metadata.rb . If we change the data here both the version will be available on here.

We can create the file with particular name, group, permission mode inside the cookbook. We can also define the same in a configuration file sothat we use some variables in cook book. It is written in a default file under the cookbook.

```
Cookbook: user node['user']
          group node['group']
default: default['user']="user123"
         default['group']="group123"
```

If we want to install any package, we perform

```
package 'httpd' do
action:install
end
service 'httpd' do
action: [:start, :enable]
end
```

(to copy file:)

```
template <dierectory'
source: index.html
end
```

To create a relation between cookbook and client we use the command: knife node run_list add client1 apache1 .

Suppose there are thousand client and If we need to create relation between client and cookbook, We create role which consist of all server. it perform 1 time job.

chef-client -i debug command will enable debug option.

All webserver can be combined and known as apache, db serber can be combined and known as my sql and app server can be combined and known as application.

Whenever there occur a change in a cookbook, At a particular time it automatically takeup by server and it will get executed.

A project will have production,staging,testing and environment. If a production get updated only after testing we make it public. So that updation will be restricted. So environment is to restrict cookbooko version based on environment. That is in production it will have the same version and in testing it will have updated version.

SECTION 36

We create role with the following command: knife role create from file <file_name>.
To add the client to server we use the command: knife bootstrap client2 username kay. This method is called bootstrap method.
Within the machine If we need to check If the browser is working fine, we use the command: curl -I <ip address>
We usually create the role from a file with the command: knife role create from <file_name>.
Chef-apply is used to execute cookbook directly on the client. chef-solo is used to execute on single node.
Hosted chef refers to that chef server will be on the cloud and that cloud will manage the clients.
If we need to edit on the cook book we use the command: export EDITOR=vi
To associate a node with the cookbook: knife run_list node cookbook. From GUI if we want to do the same, In the edit run list of node, We drag and drop node to cookbook.
knife ssh is the command used to execute a task on all nodes. If we need to do it on the client side we use the following command Knife ssh 'name=client' 'ls' .
Suppose we need to restart webserver on a particular group. we use the following command: knife ssh 'role=webserver''service httpd restart'
chef is used to execute a command on multiple machine, it can also be used as inventory.
If we need to check whether a particular package is installed, we use the command: knife search, internally search uses ohai component that get information and store on server. It will give OS information based on domain name/ platform.

SECTION 37

Chef supermarket is the repository of all the cookbooks. First we need to download the cookbook and then we need to upload the cookbook to server. After uploading then we need to create run list.

Chef can be created in different ways. One method is using with all 3 component and other is with cloud hosted, and other chef solo method.

We can directly execute the cookbook on client using the command: chef apply.

To create a user we create a user, we create a user file with following syntax:

Eg:

```
user 'userid' do
(attribute)
uid '111'
gid '12345'
homedir '/home/user1234'
pshell '/bin/bash'
password '' (we generate password in ssl- open ssl passwd)
end
```

If we need to create home directory, we need to make sure in GUI that manage home =true.

To execute a single recipe we use the following command: chef-apply file/recipes/default.rb

We create a group in chef to overcome the probllem of executing each group individually. First we create the group and then we create the user.

Syntax of this cook book:

```
group 'group111'; do
gid '12345'
end
```

Hence this group is linked with above user.

If a cookbook has a dependency on a recipe that is located in another cookbook, dependency must be declared in the metadata.rb file fo that cookbook using depends keyword.

Following recipie is included in a cookbook named my_app: include_recipe ' apache2::mod_ss1'

metadata.rb file for that cookbook would have: depends 'apache'

A recipe can be include one recipie from cookbook by using include recipe method. When a recipe is included the resource found in that recipe will be inserted at the point where include recipe keyword is located. Syntax: include_recipe 'recipe'

default

To upload a user we use following command: knife cookbook upload create user

Chef has 3 module of commands:

```
execute 'xyz'do
command "touch file123"
end
```

Cookbook_file is used to copy the non-text file. It uses the command cksum to copy the file.

To copy text file we use template. We use diff command for this purpose.
We can upload 1000 of cookbooks to a server and we can execute this easily, because it only show one cookbook. We use the following command for this purpose. Chef-client -o recipe [cookbook1].

SECTION 38

chef has if,when,group looping method.

If method:

if /etc/ssh/sshd_config file exists --> do this

if the file does not exist create it

Onlyif command is executed only if the file is present .

```
eg: template '/tmp/somefile123' do
        mode '0755'
        source 'somefile'
        only if 'test -f /etc/psswd'
        end
```

notif command is executed only if the file is not present.

```
eg: template '/tmp/somefile123' do
        mode '0755'
        source 'somefile'
        not_if do
           File.exist?('/etc/psswd')
    end
```

notify perform particular task on a file, if something happens. Notify keyword is used to restart the task immediatly or delayed.

If it is immediatly at first it copy the file, execute the command and then creates the group. If it is delayed it it copy the file and create the group and then it execute the command. Notify is usefull when we have large number of command.

eg:

```
template '/tmp/somefile123' do
        source 'server.conf'
        owner 'root'
        group 'root'
        mode '0755'
        source 'somefile'
        notify :run, 'execute[command]', :immediately
end
group 'group18890' do
  gid '1280'
end
execute 'command' do
commannd 'touch/file-test123'
action :nothing
end
```

The add cookbook to particular node: knife node run_list add nodename cookbook.

The command knife node edit nodename will vi editor the nodename where all the assosiation cookbook will prenent and we can edit this.

Suppose if there is a application file and suppose it is changed, we need to restart the system to get update the change, for that porpose we use template.

If we need to execute a shell script as part of cookbook execution, FOr this porpose we use a resource known as bash.

Eg:

```
bash 'shellscript_execute' /test112345
do
  user 'root'
  cwd '/tmp'
  code <<-EOH
  echo hi
  touch /test12345
  EOH
end
```

If want to install multi packages, we can give it in kind of loop. Given example will install particular package.

Syntax:

```
[package1,package2,package3].each do i
action: install
end
```

When loop is when a particular condition met, it will perform particular task.

component in chef is used to gather system variables for all the client. Component in cookbook is known as ohai profiler.

Cookbook can be downloaded from supermarket.chef.io . It is provided in several platform. Its server will be available in chef cloud. We can dounload the cookbook and Unzip the file.

ntp refers to network time protocol. Since ntp server will be synced to all clients. All tye of ntp operation is done be the cookbook.

By restarting ntp, all system will comes to same synchronization.

Chef cloud is available in aws which is known as ops.