

Ansible

Configuration Management Tool

Module 5: Ansible – A configuration Management (Duration-5 hrs)

➤ Introducing Ansible – A configuration management tool

o Basics / What Will Be Installed

o Understanding Ansible architecture

o Control Machine Requirements o Managed Node Requirements

➤ Inventory o Hosts and Groups

o Host Variables

o Group Variables

➤ Learn various ansible Modules

➤ How to use adhoc commands

o Parallelism and Shell Commands

o File Transfer o Managing Packages

o Users and Groups

o Deploying From Source Control

o Managing Service

➤ Introduction to YAML script

➤ Playbook o About Playbooks

o Playbook Language Example – YAML

o How to Write Playbooks

o Tasks in Playbooks

o Understanding about various tasks in playbook

o Introduction to Handlers and variables

o Learn about using handlers, variables in the playbook

o Become (Privilege Escalation)

➤ Roles

o Role Directory Structure

o Using Roles

- o Role Duplication and Execution
- o Role Default Variables
- o Role Dependencies
- o Role Search Path
- o Ansible Galaxy
 - Including and Importing
- o Includes vs. Imports
- o Importing Playbooks
- o Including and Importing Task Files
- o Including and Importing Roles
 - Writing a playbook to install and configure web servers and deploy an application
 - How to create Ansible Role and use it
 - Using an ansible role in playbook
 - How to use Ansible Galaxy to download roles.
 - Example – Install and use Jenkins roles from ansible galaxy

Ansible is simple open source IT engine which automates application deployment, intra service orchestration, cloud provisioning and many other IT tools.

Prerequisites

Before you start doing practice with various types of examples given in this tutorial, it is being assumed that you have hands-on experience with running commands into a Linux shell. This will help you the Ansible tasks in a better way.

Ansible is simple open source IT engine which automates application deployment, intra service orchestration, cloud provisioning and many other IT tools.

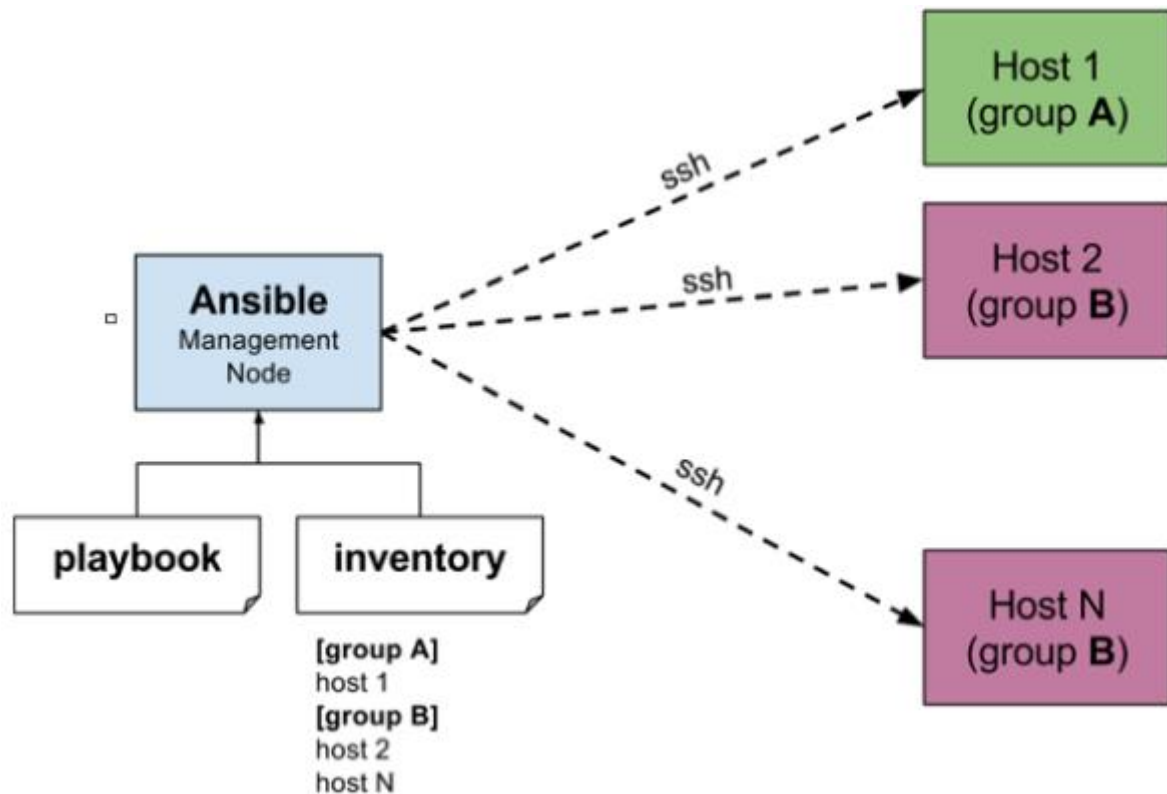
Ansible is easy to deploy because it does not use any agents or custom security infrastructure.

Ansible uses playbook to describe automation jobs, and playbook uses very simple language i.e. **YAML** (It's a human-readable data serialization language & is commonly used for configuration files, but could be used in many applications where data is being stored) which is very easy for humans to understand, read and write. Hence the advantage is that even the IT infrastructure support guys can read and understand the playbook and debug if needed (YAML – It is in human readable form).

Ansible is designed for multi-tier deployment. Ansible does not manage one system at time, it models IT infrastructure by describing all of your systems are interrelated. Ansible is completely agentless which means Ansible works by connecting your nodes through ssh (by

default). But if you want other method for connection like Kerberos, Ansible gives that option to you.

After connecting to your nodes, Ansible pushes small programs called as “Ansible Modules”. Ansible runs that modules on your nodes and removes them when finished. Ansible manages your inventory in simple text files (These are the hosts file). Ansible uses the hosts file where one can group the hosts and can control the actions on a specific group in the playbooks.



ANSIBLE INVENTORY HOST PATTERN

login as: ec2-user

Authenticating with public key "imported-openssh-key"

<https://aws.amazon.com/amazon-linux-2/>

```
[ec2-user@ip-172-31-36-113 ~]$ sudo su
```

```
[root@ip-172-31-36-113 ec2-user]# ls
```

```
[root@ip-172-31-36-113 ec2-user]# wget https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

```
[root@ip-172-31-36-113 ec2-user]# ls
```

```
epel-release-latest-7.noarch.rpm
```

```
[root@ip-172-31-36-113 ec2-user]# yum install epel-release-latest-7.noarch.rpm
```

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
```

```
Examining epel-release-latest-7.noarch.rpm: epel-release-7-14.noarch
```

```
Marking epel-release-latest-7.noarch.rpm to be installed
```

```
Resolving Dependencies
```

```
--> Running transaction check
```

```
---> Package epel-release.noarch 0:7-14 will be installed
```

```
--> Finished Dependency Resolution
```

```
amzn2-core/2/x86_64 | 3.7 kB 00:00
```

```
Dependencies Resolved
```

```
Running transaction
```

```
Installing : epel-release-7-14.noarch 1/1
```

```
Verifying : epel-release-7-14.noarch 1/1
```

```
Installed:
```

```
epel-release.noarch 0:7-14
```

```
Complete!
```

```
[root@ip-172-31-36-113 ec2-user]# yum update -y
```

Installation through YUM on AWS Machine as below

```
Install Packages GIT PYTHON ANSIBLE PYTHON_PIP OPENSSEL
```

```
[root@ip-172-31-36-113 ec2-user]# yum install git python python-level python-pip  
openssl ansible -y
```

Dependency Installed:

```
git-core.x86_64 0:2.37.1-1.amzn2.0.1
git-core-doc.noarch 0:2.37.1-1.amzn2.0.1
perl-Error.noarch 1:0.17020-2.amzn2
perl-Git.noarch 0:2.37.1-1.amzn2.0.1
perl-TermReadKey.x86_64 0:2.30-20.amzn2.0.2
python-paramiko.noarch 0:2.1.1-0.10.el7
```

complete!

Check the installed version

```
[root@ip-172-31-36-113 ec2-user]# python --version
```

Python 2.7.18

```
[root@ip-172-31-36-113 ec2-user]# python-level --version
```

bash: python-level: command not found

```
[root@ip-172-31-36-113 ec2-user]# git --version
```

git version 2.37.1

```
[root@ip-172-31-36-113 ec2-user]# ansible --version
```

ansible 2.9.27

config file = /etc/ansible/ansible.cfg

python version = 2.7.18 (default, May 25 2022, 14:30:51) [GCC 7.3.1 20180712 (Red Hat 7.3.1-15)]

```
[root@ip-172-31-36-113 ec2-user]# python --version
```

Python 2.7.18

NOW GO TO host file with the following command below to add IP of Ansible node for access.

```
[root@ip-172-31-36-113 ec2-user]# vi /etc/ansible/hosts
```

Update ansible Ansible cfg file with below command

Insert Node 1 & node 2 private ip of aws instance

Below

[Demo] [] USE THIS ALWAYS

IP node 1

IP node 2

Exit and save file

Agenda : to provide All Root privileges to ansible user

[root@ip-172-31-36-113 ec2-user]# vi /etc/ansible/ansible.cfg

Open file , search for

inventory = /etc/ansible/hosts

sudo-user = root

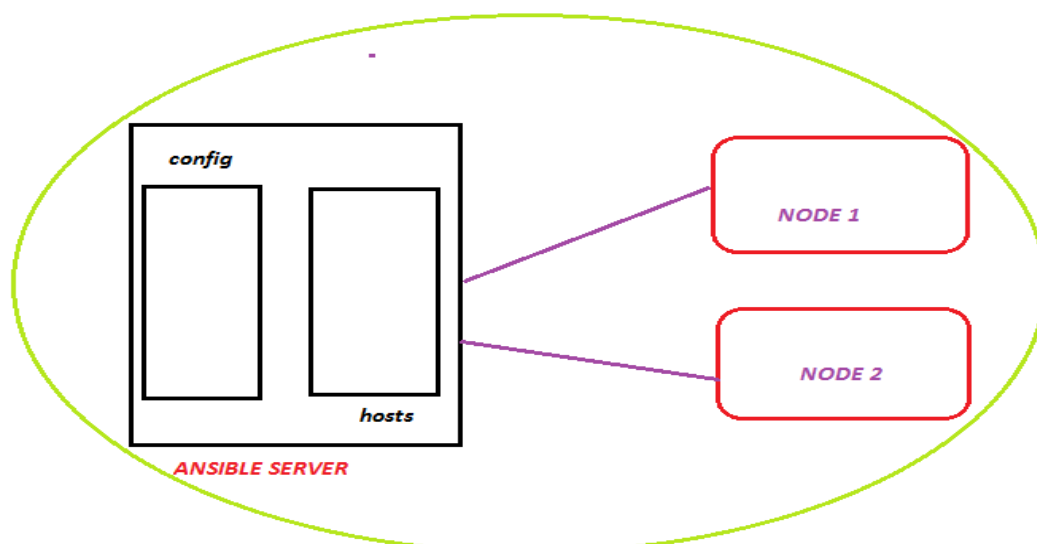
Uncomment above both with removing **#** as below

inventory = /etc/ansible/hosts

sudo-user = root

save & exit.

Create a Ansible user in Ansible serve machine and node1 , node2



Add user in the Ansible Mater and NODE1 Node2

[root@ip-172-31-36-113 ec2-user]# adduser ansible

```
[root@ip-172-31-36-113 ec2-user]# passwd ansible
```

Changing password for user ansible.

New password: technical

BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word

Retype new password: technical

passwd: all authentication tokens updated successfully.

*****Repeat above method in both the nodes and create ansible user

**** Switch to as User to Ansible USER in node1 & node 2

node 1

```
[root@ip-172-31-36-113 ec2-user]# adduser ansible
```

```
[root@ip-172-31-36-113 ec2-user]# passwd ansible
```

node 2

```
[root@ip-172-31-36-113 ec2-user]# adduser ansible
```

```
[root@ip-172-31-36-113 ec2-user]# passwd ansible
```

Move to Ansible server machine 1

1. Ansible server

```
[root@ip-172-31-36-113 ec2-user]# su - ansible
```

Ansible

Create a empty file

```
[ansible@ip-172-31-36-113 ~]$ touch file1
```

```
[ansible@ip-172-31-36-113 ~]$ ls
```

File1

2. Install Httpd packages Apache3

**** Use sudo as the root user with priveleges.

```
[ansible@ip-172-31-36-113 ~]$ sudo yum install httpd -y
```

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

- #1) Respect the privacy of others.
- #2) Think before you type.
- #3) With great power comes great responsibility.

[sudo] password for ansible:technical

ansible is not in the sudoers file. This incident will be reported.

```
[ansible@ip-172-31-36-113 ~]$ exit
```

Logout

Open Visudo

Open file and find as below

```
[root@ip-172-31-36-113 ec2-user]# visudo
```

Allow root to run any command anywhere without password always

DEFAULT – ROOT ALL= [ALL] ALL

ADD ANSIBLE ALL= [ALL] NOPASSWD: ALL

:wq exit

Update in

*****Repeat above method in both the nodes to provide Sudo privileges

Node1

```
[root@ip-172-31-36-113 ec2-user]# visudo
```

Node 2


```
[root@ip-172-31-36-113 ec2-user]# visudo
```

switch as ansible user with command below in all machine server and node1 node2

```
[root@ip-172-31-36-113 ec2-user]# su - ansible
```

****Install httpd package on server machine

```
[ansible@ip-172-31-36-113 ~]$ sudo yum install httpd -y
```

Loaded plugins: extras_suggestions, langpacks, priorities, update-motd

229 packages excluded due to repository priority protections

Complete!

Now take the access of ansible node 1 with private IP of Ansible

Copy private id of node 1 and node 2

```
[ansible@ip-172-31-36-113 ~]$ ssh 172.31.36.241
```

ERROR: The authenticity of host '172.31.36.241 (172.31.36.241)' can't be established.

Exit now to

```
[ansible@ip-172-31-36-113 ~]$ exit
```

We will open an default Ansible configuration file as config follows:

```
[root@ip-172-31-36-113 ec2-user]# vi /etc/ssh/sshd_config
```

Vi edit opens file ,find as below

-
1. # password authentication YES

Remove # to make it uncommnted now its

password authentication YES

2. password authentication no

add # commented to above line

save :wq and exit

Restart as to update service

```
[root@ip-172-31-36-113 ec2-user]# service sshd restart
```

Redirecting to /bin/systemctl restart sshd.service

AGENDA TO TAKE ACCESS OF NODE1 FROM ANSIBLE MC 1

```
[root@ip-172-31-36-113 ec2-user]# su - ansible
```

Last login: Sat Nov 5 06:45:18 UTC 2022 on pts/0

```
[ansible@ip-172-31-36-113 ~]$ ssh 172.31.36.241 (ip of node 1)
```

```
ansible@172.31.36.241's password: technical [as we set previous]
```

Last login: Sat Nov 5 06:45:41 2022

Now accessing as node 1 as connection established @ip-172-31-36-241

FILES CREATION

Lets create few empty files with touch or nano command

```
[ansible@ip-172-31-36-241 ~]$ touch file2 file3
```

```
[ansible@ip-172-31-36-241 ~]$ touch file6
```

```
[ansible@ip-172-31-36-241 ~]$ ls
```

file2 file3 file6

All files created and visible with ls

```
[ansible@ip-172-31-36-241 ~]$ touch filez
```

```
[ansible@ip-172-31-36-241 ~]$ exit
```

logout

Connection to 172.31.36.241 closed.

AGENDA TO TAKE ACCESS OF NODE2 FROM ANSIBLE MC 1

```
[ansible@ip-172-31-36-113 ~]$ ssh 172.31.33.44 (ip of node 2)
```

ansible@172.31.33.44's password: technical

Last login: Sat Nov 5 06:45:49 2022

[ansible@ ~]\$ ls

No existing file as not created . create some files.

FILES CREATION

[ansible@ip-172-31-33-44 ~]\$ touch fileb filec

[ansible@ip-172-31-33-44 ~]\$ ls

fileb filec

file creation succesfull ,then exit from node 2 access

[ansible@ip-172-31-33-44 ~]\$ exit

logout

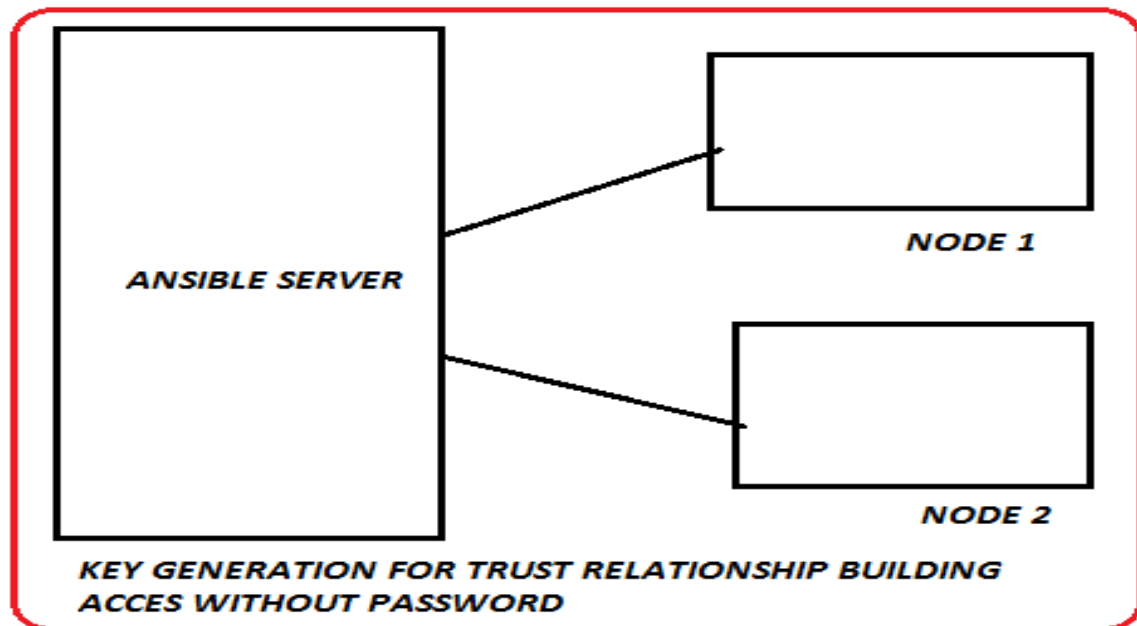
Connection to 172.31.33.44 closed.

TRUST RELATIONSHIP

To access all node ,everytime password is mandatory to connect with node1 nd node2 from ansible user

ROOT ---- ROOT

USER ---- USER



```
ansible@ip-172-31-36-113 ~]$ ssh-keygen
```

Generating public/private rsa key pair.

Enter file in which to save the key (/home/ansible/.ssh/id_rsa):

```
[ansible@ip-172-31-36-113 ~]$ ls -a
```

```
. .. .bash_history .bash_logout .bash_profile .bashrc file .ssh
```

CHANGE DIRECTORY

```
[ansible@ip-172-31-36-113 ~]$ cd .ssh/
```

```
[ansible@ip-172-31-36-113 .ssh]$ ls
```

```
id_rsa id_rsa.pub known_hosts
```

NOW NEED TO COPY PRIVATE KEY IN BOTH THE NODES

```
[ansible@ip-172-31-36-113 .ssh]$ ssh-copy-id ansible@172.31.36.241
```

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:  
"/home/ansible/.ssh/id_rsa.pub"
```

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any  
that are already installed
```

```
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now  
it is to install the new keys
```

```
ansible@172.31.36.241's password: TECHINCAL
```

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ansible@172.31.36.241'"
and check to make sure that only the key(s) you wanted were added.

ADD ANOTHER NODES IP NODE 2

```
[ansible@ip-172-31-36-113 .ssh]$ ssh-copy-id ansible@172.31.33.44
```

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:  
"/home/ansible/.ssh/id_rsa.pub"
```

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any  
that are already installed
```

```
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now  
it is to install the new keys
```

```
ansible@172.31.33.44's password:
```

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ansible@172.31.33.44'"
and check to make sure that only the key(s) you wanted were added.

```
[ansible@ip-172-31-36-113 .ssh]$ cd ..
```

```
EXIT
```

WE PROVIDE TRUST RELATIONSHIP

WE CAN ACCESS DIRECTLY CODE WITHOUT PASSWORD TO NODES

```
[ansible@ip-172-31-36-113 ~]$ ssh 172.31.36.241
```

```
Last login: Sat Nov  5 06:56:06 2022
```

```
logout
```

```
Connection to 172.31.36.241 closed.
```

```
[ansible@ip-172-31-36-113 ~]$ ssh 172.31.33.44
```

Last login: Sat Nov 5 06:58:37 2022

Its in accessible

```
[ansible@ip-172-31-33-44 ~]$ ls
```

fileb filec

```
[ansible@ip-172-31-33-44 ~]$ touch security
```

HOST PATTERN

Patterns: targeting hosts and groups₃

When you execute Ansible through an ad hoc command or by running a playbook, you must choose which managed nodes or groups you want to execute against. Patterns let you run commands and playbooks against specific hosts and/or groups in your inventory. An Ansible pattern can refer to a single host, an IP address, an inventory group, a set of groups, or all hosts in your inventory. Patterns are highly flexible - you can exclude or require subsets of hosts, use wildcards or regular expressions, and more. Ansible executes on all inventory hosts included in the pattern

You use a pattern almost any time you execute an ad hoc command or a playbook. The pattern is the only element of an ad hoc command that has no flag. It is usually the second element

WE HAVE 10 NODES CONNECTED WITH ANSIBLE SERVER OR TO THE GROUPS FOLLOWS THE SAME METHOD .

```
[ansible@ip-172-31-36-113 ~]$ ansible all --list-hosts
```

hosts (3):

(demo)

172.31.36.241

172.31.33.44

hosts (0):

```
[ansible@ip-172-31-36-113 ~]$ ansible all --list-hosts
```

hosts (3):

(demo)

172.31.36.241

172.31.33.44

NODE IN GROUP - DEMO

```
[root@ip-172-31-36-113 ec2-user]# ansible demo --list-hosts
```

hosts (2):

172.31.36.241

172.31.33.44

you can use subscripts to select individual hosts or ranges within the webservers group:

```
webservers[0]    # == cobweb
webservers[-1]   # == weber
webservers[0:2]  # == websockets[0],websockets[1]
                  # == cobweb,webbing
websockets[1:]   # == webbing,weber
websockets[:3]   # == cobweb,webbing,weber
```

0 IS 1

1 IS 2

```
[root@ip-172-31-36-113 ec2-user]# ansible demo[0] --list-hosts
```

hosts (1):

172.31.36.241

```
[root@ip-172-31-36-113 ec2-user]# ansible demo[0] --list-hosts
```

hosts (1):

172.31.36.241

```
[root@ip-172-31-36-113 ec2-user]# ansible demo[-1] --list-hosts
```

hosts (1):

172.31.33.44

```
[root@ip-172-31-36-113 ec2-user]# ansible demo[0:1] --list-hosts
```

hosts (2):

172.31.36.241

172.31.33.44

[root@ip-172-31-36-113 ec2-user]# su - ansible

Last login: Sat Nov 5 07:01:59 UTC 2022 on pts/0

[ansible@ip-172-31-36-113 ~]\$ ansible demo[0:1] --list-hosts

hosts (2):

172.31.36.241

172.31.33.44

END OF LECTURE

LEC-33

ANSIBLE AD HOC COMMANDS MODULES & PLAYBOOK

1. Ad- hoc commands (simple linux comands)
It has no idemputency in it
2. Modulus - Single line command
3. Playbooks - More than one modules

AD-HOC commands:

- Ad-hoc commands are commands which can be run individuals to perform quick functions.
- These commands are not used for configuration management & deployment as they are one time usage.
- These commands uses /usr/bin/ansible command line tool to automate a single task.

LAB

SWITCH TO ANSIBLE USER FROM MACHINE 1 ANSIBLE SERVER

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -a "ls"
```

[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at

/usr/bin/python, but future installation of another Python interpreter could change this. See

file6

filez

```
[ansible@ip-172-31-36-113 ~]$ touch file1
```

```
[ansible@ip-172-31-36-113 ~]$ ls
```

File

LIST OF FILE FROM DEMO = GROUP [NODE1 & NODE2] , A IS ARGUMENT

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -a "ls"
```

[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at

/usr/bin/python, but future installation of another Python interpreter could change this. See

https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more

information.

172.31.36.241 | CHANGED | rc=0 >>

file2

file3

file6

filez

```
[ansible@ip-172-31-36-113 ~]$ ansible all -a "ls"
```

[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at

/usr/bin/python, but future installation of another Python interpreter could change this. See

https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more

information.

172.31.33.44 | CHANGED | rc=0 >>

fileb

filec

security

[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python

172.31.36.241 | CHANGED | rc=0 >>

file2

file3

file6

filez

```
[ansible@ip-172-31-36-113 ~]$ ansible all -a "touch rajputfile"
```

[WARNING]: Consider using the file module with state=touch rather than running 'touch'. If you need

to use command because file is insufficient you can add 'warn: false' to this command task or set

172.31.36.241 | CHANGED | rc=0 >>

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -a "ls -al"
```

[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at

/usr/bin/python, but future installation of another Python interpreter could change this.
See

https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html
for more

information.

172.31.33.44 | CHANGED | rc=0 >>

total 16

drwx----- 4 ansible ansible 171 Nov 5 07:36 .

drwxr-xr-x 4 root root 37 Nov 5 06:30 ..

drwx----- 3 ansible ansible 17 Nov 5 07:26 .ansible

-rw----- 1 ansible ansible 82 Nov 5 07:09 .bash_history

-rw-r--r-- 1 ansible ansible 18 Jul 15 2020 .bash_logout

-rw-r--r-- 1 ansible ansible 193 Jul 15 2020 .bash_profile

-rw-r--r-- 1 ansible ansible 231 Jul 15 2020 .bashrc

-rw-rw-r-- 1 ansible ansible 0 Nov 5 06:58 fileb

-rw-rw-r-- 1 ansible ansible 0 Nov 5 06:58 filec

-rw-rw-r-- 1 ansible ansible 0 Nov 5 07:36 rajputfile

-rw-rw-r-- 1 ansible ansible 0 Nov 5 07:07 security

drwx----- 2 ansible ansible 29 Nov 5 07:05 .ssh

[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python
interpreter at

/usr/bin/python, but future installation of another Python interpreter could change this.
See

https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html
for more

information

[ansible@ip-172-31-36-113 ~]\$ ansible demo -a "touch rajputfile"

[WARNING]: Consider using the file module with state=touch rather than running

.

HTTPD PACKAGE INSTALLATION:

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -a "sudo yum install httpd -y"
```

[WARNING]: Consider using 'become', 'become_method', and 'become_user' rather than running sudo

Complete!

CHECK PACKAGE INSTALLED ON BOTH THE NODE I&2 WITH BELOW CMD

```
[ansible@ip-172-31-36-113 ~]$ which httpd
```

HTTPD PACKAGE REMOVAL:

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -ba "yum remove httpd -y"
```

[WARNING]: Consider using the yum module rather than running 'yum'. If you need to use command

Transaction Summary

Remove 1 Package (+1 Dependent package)

Dependency Removed:

mod_http2.x86_64 0:1.15.19-1.amzn2.0.1

Complete!

CHECK PACKAGE INSTALLED AND REMOVED ON BOTH THE NODE I&2 WITH BELOW CMD

```
[ansible@ip-172-31-36-113 ~]$ which httpd
```

ANSIBLE MODULES:

Introduction to modules

Ansible ships with a number of modules (module library) that can be executed on remote hosts or through, playbooks"

Your library of modules can reside on machine and there are no servers, demons database required

Modules (also referred to as “task plugins” or “library plugins”) are discrete units of code that can be used from the command line or in a playbook task. Ansible executes each module, usually on the remote target node, and collects return values.

You can execute modules from the command line:

You can execute modules from the command line:

```
ansible webservers -m service -a "name=httpd state=started"
ansible webservers -m ping
ansible webservers -m command -a "/sbin/reboot -t now"
```

From playbooks, Ansible modules are executed in a very similar way:

```
- name: reboot the servers
  action: command /sbin/reboot -t now
```

Which can be abbreviated to:

```
- name: reboot the servers
  command: /sbin/reboot -t now
```

Where ansible modules are stores:

****The default location for the inventory files are
/etc/ansible/hosts

Ansible Modules

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a " pkg=httpd state:
present"
```

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a "pkg=httpd state-
latest"
```

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a "pkg=httpd state=absent"
```

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a "pkg=httpd
state=started"
```

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m user -a "name = raj"
```

Yaml

Install = present / Uninstall= absent / Update= latest /Start=Started

```
[ansible@ip-172-31-36-113 ~]$ ansible demo-b-m Yum a "pkg=httpd state=present"
```

o/ p installed / node1 & node2 . check which httpd

o/p installed

-Repeat above command

already installed visible.

module has idempotency

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a "pkg=httpd state=latest"
```

updated

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -b -m yum -a "pkg=httpd state-absent"
```

Removes from group demo sever Node1 & node2

```
[ansible@ip-172-31-36-113 ~]$ sudo service httpd status
```

op/inactive

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -b -m service -a "name=httpd state=Started"
```

started.

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -b -m -user -a "name=asim"
```

Created in all server. checked.

```
[ansible@ip-172-31-36-113 ~]$ cat /etc/passwd.
```

Node1 [ip] touch copiedfromserver. (create file)

Is –

file 1 Copiedfromserver.

```
ansible@ip-172-31-36-113 ~]$ ansible demo -b -m copy -a "** Src file4 dest=/temp"
```

dest = destination

```
[ansible@ip-172-31-36-113 ~]$ ansible demo [-1] -b -m copy -a "src= technical  
Guftugu dest=/tmp"
```

Ansible Modules idempotency is present and it will not repeat same commands.

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -m setup.
```

** Setup run while execution to check Current Configuration of linked server

```
[ansible@ip-172-31-36-113 ~]$ ansible demo -m setup -a "filter=*ipv4*"
```

display all ip address of Node 2 & Node 1

LEC-34

**Ansible YAML PLAYBOOK,VARIABLES,HANDLERS &
LOOPS**

YAML

Ansible are written in YAML format It is human readable data serialization language.

Commonly used for configuration files. you can code

- Playbook a like a file where Consist of vars, tasks handlers, files templates and roles.

Each Playbook is composed. of one or more 'modules' in a list . Modules is a collection of Configuration files.

Playbooks are divided into many Sectors like:

1. Target section. Defines the host against when playbooks taske has to be executed.
2. Variable section : Defines variably
3. Task section: list of all modules need to run in an order

Playbooks

Target
Variable
Task
Handler
Condition
vars

What is YAML, and how does it benefits?

YAML is a data serialization format that stands for **YAML ain't Markup language**.

The main advantage of using YAML is readability and writability. If you have a configuration file that needs to be easier for humans to read, it's better to use YAML. YAML is not a complete substitution of JSON as JSON and XML have their places too; nevertheless, it's useful learning YAML.

Another benefit of YAML is its support of various data types like cases, arrays, dictionaries, lists, and scalars. It has good support for the most popular languages like JavaScript, Python, Ruby, Java, etc.

YAML only supports spaces, and it is case sensitive as well as space sensitive. Tabs are not accepted universally. A YAML file has `.yaml` extension.

Data serialization

Whenever you want to send some data structure or an object across computer networks, say the Internet, you have to turn it into a special format to read it and store it. The process is commonly known as serialization and is of enormous importance on the web. A common usage example of serialization is when reading data from databases and transferring it across the web.

Some serialization formations include JSON, YAML, XML.

In this article, we talk about YAML, and at the end of the article, you'll be able to work your way through YAML and have a clear introduction to YAML.

YAML PLAYBOOK

In this chapter, we will learn about Playbooks in Ansible.

Playbooks are the files where Ansible code is written. Playbooks are written in YAML format. YAML stands for Yet Another Markup Language. **Playbooks** are one of the core features of Ansible and tell Ansible what to execute. They are like a to-do list for Ansible that contains a list of tasks.

Playbooks contain the steps which the user wants to execute on a particular machine. Playbooks are run sequentially. Playbooks are the building blocks for all the use cases of Ansible.

key-value pair

YAML uses simple key-value pair to represent the data. The dictionary is represented in key: value pair.

Note – There should be space between : and value.

Example: A student record

```
--- #Optional YAML start syntax
james:
  name: james john
  rollNo: 34
  div: B
  sex: male
... #Optional YAML end syntax
```

```
ansible@ip-172-31-36-113:~
19 package(s) needed for security, out of 25 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-36-113 ~]$ sudo su
[root@ip-172-31-36-113 ec2-user]# su - ansible
Last login: Sat Nov  5 08:49:09 UTC 2022 on pts/2
[ansible@ip-172-31-36-113 ~]$ ls
copiedfromserver  file1  technicalguftugu
[ansible@ip-172-31-36-113 ~]$ vi target.yml
[ansible@ip-172-31-36-113 ~]$ ansible-playbook target.yml
```

```
[ansible@ip-172-31-36-113 ~]$ vi target.yml
```

```

-- # My First Testing YAML Playbook
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  gather_facts: yes

```

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook target.yml
```

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook target.yml
```

```
PLAY [demo] *****

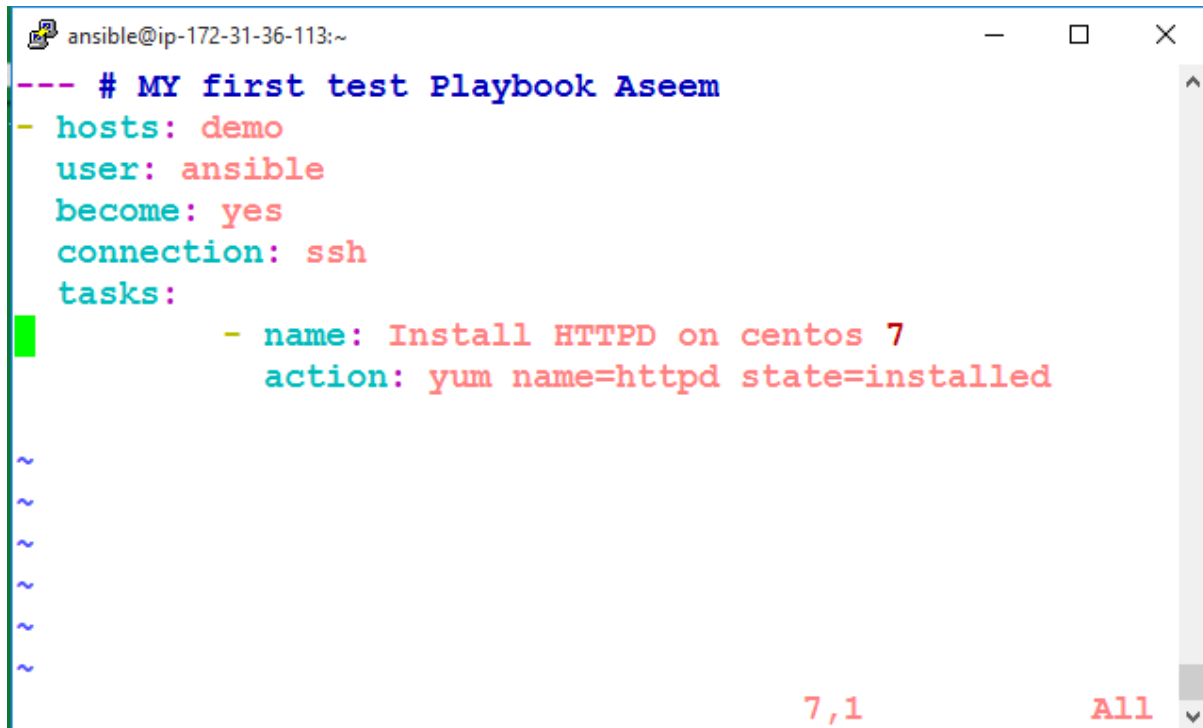
TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference\_appendices/interpreter\_discovery.html for more
information.
ok: [172.31.33.44]
[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference\_appendices/interpreter\_discovery.html for more
information.
ok: [172.31.36.241]

PLAY RECAP *****
172.31.33.44      : ok=1    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0
172.31.36.241    : ok=1    changed=0    unreachable=0    failed=0    skipped=0    resc
ued=0    ignored=0

[ansible@ip-172-31-36-113 ~]$ vi target.yml
[ansible@ip-172-31-36-113 ~]$
```

2. Create one more Yaml playbooks

\$ Vi task.yml



```
--- # MY first test Playbook Aseem
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  tasks:
    - name: Install HTTPD on centos 7
      action: yum name=httpd state=installed
```

Host : demo is group name mention in node1 node2

User : ansible user name ansible determined earlier

Become : sudo privileges given

Connection: connection access through **ssh**

Task: work mentioned in playbook , httpd to be installed

[ansible@ip-172-31-36-113 ~]\$ ansible-playbook task.yml

Commands run and install httpd on Node1 and node 2

Check httpd installed or not

[ansible@ip-172-31-36-113 ~]\$ which httpd

/usr/sbin/httpd

Installed as playbook run successfully.

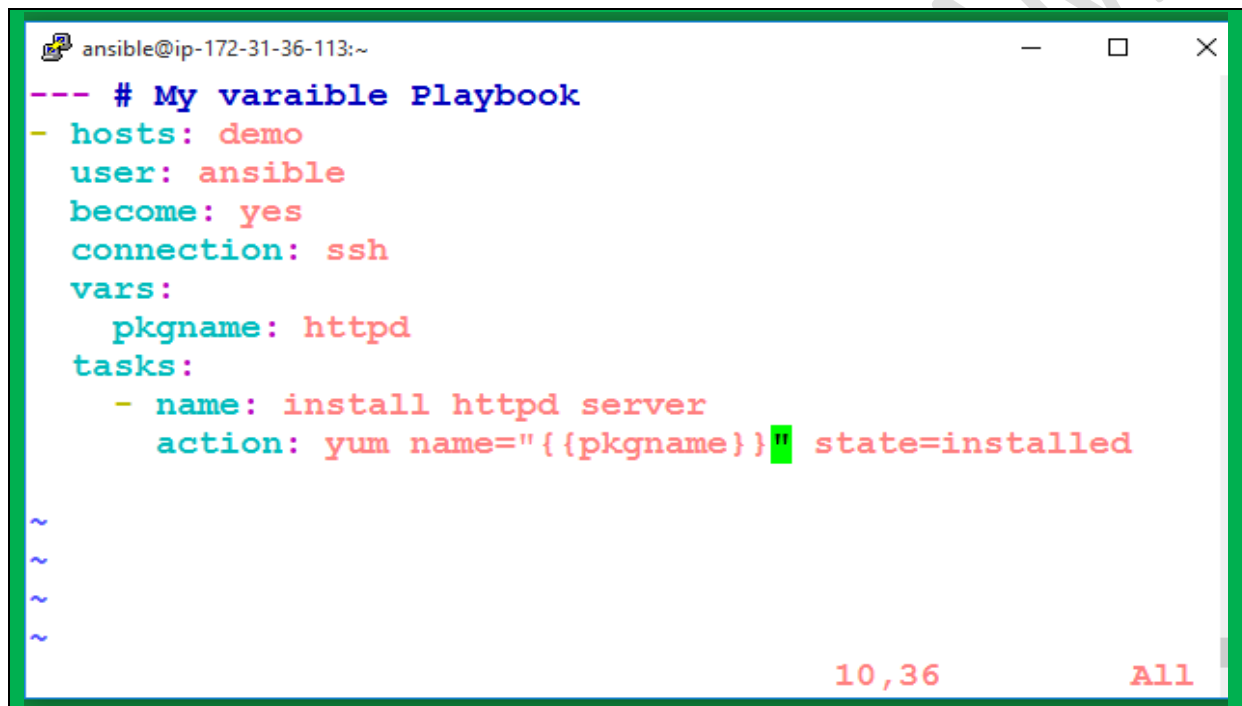
Remove httpd packages from node 1 and 2

Variables

Create one yml as

```
[ansible@ip-172-31-36-113 ~]$ vi vars.yml
```

Vi editor opens



```
ansible@ip-172-31-36-113:~  
--- # My variable Playbook  
- hosts: demo  
  user: ansible  
  become: yes  
  connection: ssh  
  vars:  
    pkgname: httpd  
  tasks:  
    - name: install httpd server  
      action: yum name="{{pkgname}}" state=installed  
~  
~  
~  
~  
10,36 All
```

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook vars.yml
```

Playbooks executes and run .

```
ansible@ip-172-31-36-113:~
TASK [install httpd server] *****
ok: [172.31.36.241]
changed: [172.31.33.44]

PLAY RECAP *****
172.31.33.44      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.36.241    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ansible@ip-172-31-36-113 ~]$ vi vars.yml
[ansible@ip-172-31-36-113 ~]$ ansible-playbook vars.yml

PLAY [demo] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.36.241]
[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.33.44]

TASK [install httpd server] *****
ok: [172.31.33.44]
ok: [172.31.36.241]

PLAY RECAP *****
172.31.33.44      : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.36.241    : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ansible@ip-172-31-36-113 ~]$
```

Httpd package installed on all nodes

`~]$` which httpd

`/usr/sbin/httpd`

Installed as playbook run successfully.

```
[ansible@ip-172-31-36-113 ~]$ ls
copiedfromserver  file1  target.yml  task.yml  technicalguftugu  vars.yml
[ansible@ip-172-31-36-113 ~]$
```

HANDLERS AND DRY RUN PLAYBOOK

DELETE httpd packages from nodes 1 & 2

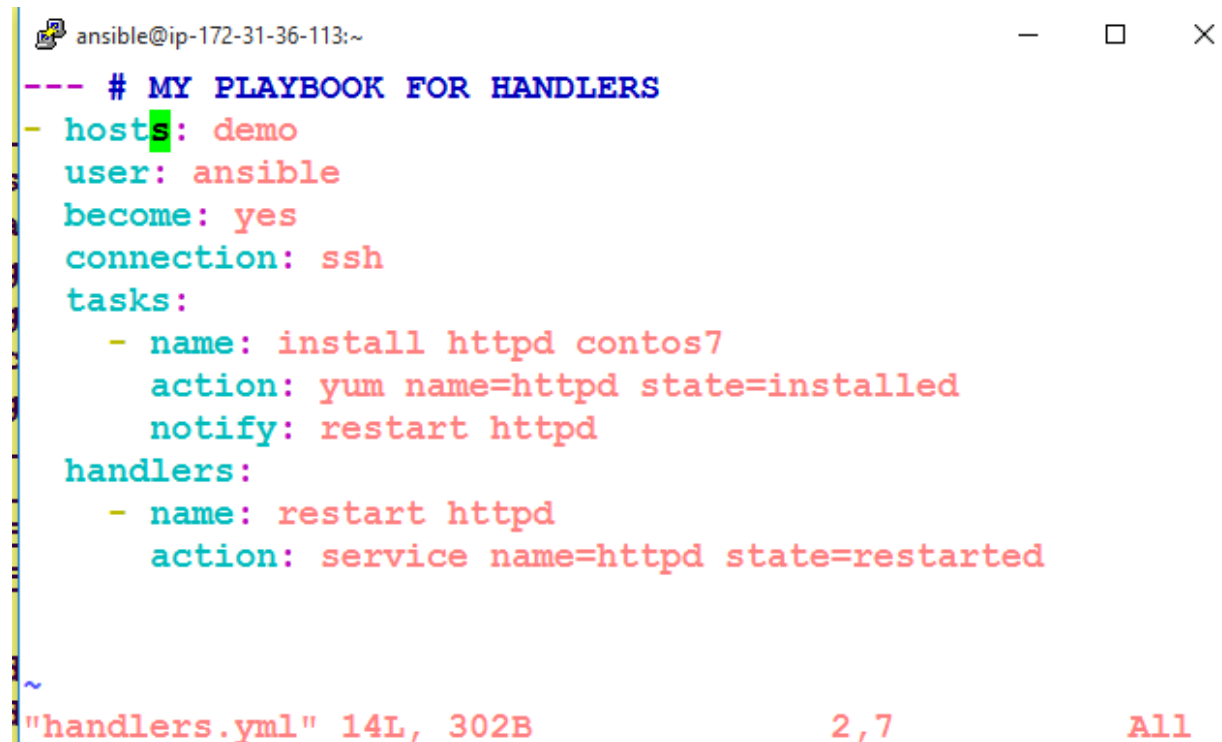
Agenda

Write a playbook with below execution in it

1. To know connected ips in a demo group
2. Install httpd packages in demo group (node 1 and node 2)
3. Start httpd service

Write a playbook with name

```
[ansible@ip-172-31-36-113 ~]$ vi handlers.yml
```



```
ansible@ip-172-31-36-113:~
--- # MY PLAYBOOK FOR HANDLERS
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  tasks:
    - name: install httpd contos7
      action: yum name=httpd state=installed
      notify: restart httpd
  handlers:
    - name: restart httpd
      action: service name=httpd state=restarted

~
"handlers.yml" 14L, 302B          2,7          All
```

Dry run to know errors

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook handlers.yml --check
```

No errors

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook handlers.yml
```

Playbooks executes and run .

```
ansible@ip-172-31-36-113:~$ ansible-playbook handlers.yml
172.31.36.241 : ok=3 changed=2 unreach=0 failed=0 skipped=0 rescued=0 ignored=0
=0

[ansible@ip-172-31-36-113 ~]$ vi handlers.yml
[ansible@ip-172-31-36-113 ~]$ ansible-playbook handlers.yml

PLAY [demo] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.33.44]
[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python interpreter at /usr/bin/python, but future
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.36.241]

TASK [install httpd contos7] *****
changed: [172.31.36.241]
changed: [172.31.33.44]

RUNNING HANDLER [restart httpd] *****
changed: [172.31.36.241]
changed: [172.31.33.44]

PLAY RECAP *****
172.31.33.44 : ok=3 changed=2 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
172.31.36.241 : ok=3 changed=2 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

[ansible@ip-172-31-36-113 ~]$
```

Check httpd packages in node1 and node2 is installed or not.

Httpd package installed on all nodes

~]\$ which httpd

/usr/sbin/httpd

~]\$ sudo service httpd status

It is in running state

```
Complete!
[ansible@ip-172-31-36-241 ~]$ which httpd
/usr/sbin/httpd
[ansible@ip-172-31-36-241 ~]$ which httpd
/usr/sbin/httpd
[ansible@ip-172-31-36-241 ~]$ sudo service http status
Redirecting to /bin/systemctl status http.service
Unit http.service could not be found.
[ansible@ip-172-31-36-241 ~]$ sudo 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 prese
t: disabled)
   Active: active (running) since Thu 2022-11-10 09:01:48 UTC; 8min ago
     Docs: man:httpd.service(8)
  Main PID: 6651 (httpd)
    Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes se
rved/sec: 0 B/sec"
    CGroup: /system.slice/httpd.service
            └─6651 /usr/sbin/httpd -DFOREGROUND
            └─6653 /usr/sbin/httpd -DFOREGROUND
            └─6654 /usr/sbin/httpd -DFOREGROUND
            └─6655 /usr/sbin/httpd -DFOREGROUND
            └─6656 /usr/sbin/httpd -DFOREGROUND
            └─6657 /usr/sbin/httpd -DFOREGROUND

Nov 10 09:01:48 ip-172-31-36-241.ap-south-1.compute.internal systemd[1]: Star...
Nov 10 09:01:48 ip-172-31-36-241.ap-south-1.compute.internal systemd[1]: Star...
Hint: Some lines were ellipsized, use -l to show in full.
[ansible@ip-172-31-36-241 ~]$
```

Agenda completed

ANSIBLE LOOPS IN PLAYBOOK

Agenda

Write a playbook with below execution in it

1. To know connected ips in a demo group
2. Create 4 user (node 1 and node 2)

Write a playbook with name

```
[ansible@ip-172-31-36-113 ~]$ vi loops.yml
```


ansible@ip-172-31-36-113:~

```
--- # MY LOOPS PLAYBOOK
- hosts: demo
  become: yes
  connection: ssh
  tasks:
    - name: add list of users in my nodes
      user: name="{{item}}" state=present
      with_items:
        - Aseemakram
        - amaira
        - afroz
        - sachin

~
~
"loops.yml" 13L, 277B                               12,13
```

Created user 4 . view file with Cat command

ansible@ip-172-31-36-113:~

```
cat: loopd.yml: No such file or directory
[ansible@ip-172-31-36-113 ~]$ cat loops.yml
--- # MY LOOPS PLAYBOOK
- hosts: demo
  become: yes
  connection: ssh
  tasks:
    - name: add list of users in my nodes
      user: name="{{item}}" state=present
      with_items:
        - Aseemakram
        - amaira
        - afroz
        - sachin

[ansible@ip-172-31-36-113 ~]$
```

User created in all nodes

Check in nodes

```
[ansible@ip-172-31-36-113 ~]$ cat /etc/passwd
```

User displays in both the nodes.

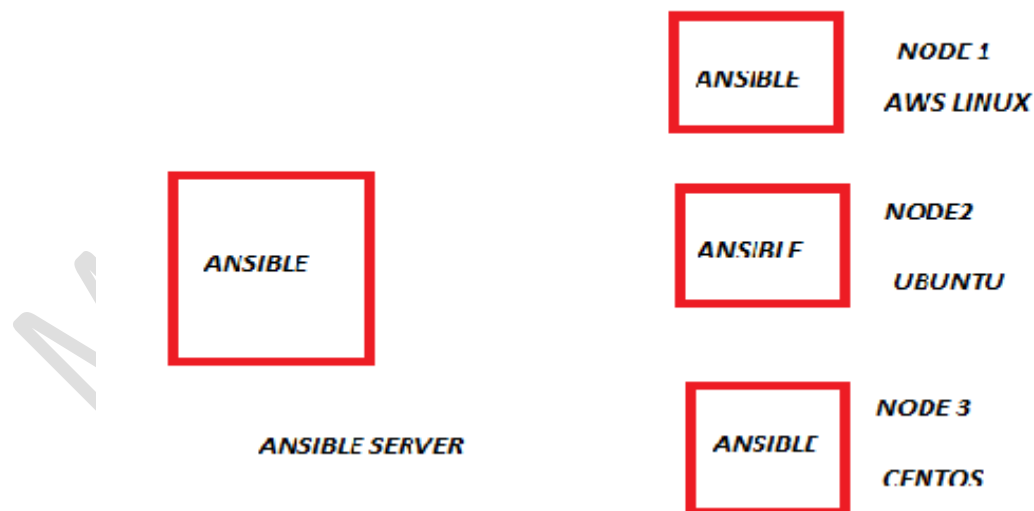
```
ec2-user:x:1000:1000:EC2 Default User:/home/ec2-user:/bin/bash
ansible:x:1001:1001::/home/ansible:/bin/bash
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
asim:x:1002:1002::/home/asim:/bin/bash
Aseemakram:x:1003:1003::/home/Aseemakram:/bin/bash
amaira:x:1004:1004::/home/amaira:/bin/bash
afroz:x:1005:1005::/home/afroz:/bin/bash
sachin:x:1006:1006::/home/sachin:/bin/bash
[ansible@ip-172-31-33-44 ~]$
```

LEC – 35 ANSIBLE CONDITIONS VAULT AND ROLES

In different scenario we place different condition to perform the task accordingly.

Scenario 1 . Node with different o.s are linked in the Ansible server e.g below:

Ansible server



WHEN STATEMENT IN PLAYBOOK

Sometimes you want to skip a particular command on particular node .

Agenda:

1. To retrieve existing nodes IP gathering facts
2. To install apache package on node 1 & node 2 ansible

Debian family = apache and RedHat family = HTTPD package

Remove httpd package from node if existing

```
[ansible@ip-172-31-36-241 ~]$ sudo yum remove httpd
```

```
[ansible@ip-172-31-36-241 ~]$ which httpd
```

/usr/bin/which: no httpd in

(/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/ansible/.local/bin:/home/ansible/bin)

Playbook as below :

```
[ansible@ip-172-31-36-113 ~]$ vi condition.yml
```

```
--- # condition playbook (demo is group yes is sudo, ansible is user )
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  tasks:
    - name: Install apache server on Debian
      command: apt-get -y install apache2
      when: ansible_os_family == "Debian"
    - name: Install apache for redhat
      command: yum -y install httpd
      when: ansible_os_family == "RedHat"
```

```
[ansible@ip-172-31-36-113 ~]$ ansible-playbook condition.yml
```

```
PLAY [demo] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information.
ok: [172.31.33.44]
[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information.
ok: [172.31.36.241]

TASK [Install apache server on Debian] *****
skipping: [172.31.36.241]
skipping: [172.31.33.44]

TASK [Install apache for redhat] *****
[WARNING]: Consider using the yum module rather than running 'yum'. If you need to use command
because yum is insufficient you can add 'warn: false' to this command task or set
'command_warnings=False' in ansible.cfg to get rid of this message.
changed: [172.31.33.44]
changed: [172.31.36.241]

PLAY RECAP *****
172.31.33.44      : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
172.31.36.241    : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0
ignored=0
```

Installed .

VAULT IN PLAYBOOK

- Ansible allows keeping sensitive data such as password or key in encrypted files, rather than a plain text in your playbook.

AGENDA:

- To create a new encrypted playbook command:

```
[ansible@ip-172-31-36-113 ~]$ ansible-vault create vault.yml
```

```
New Vault password: technical
```

```
-- # My First Testing YAML Playbook
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  gather_facts: yes
```

```
[ansible@ip-172-31-36-113 ~]$ vi vault.yml
```

```
$ANSIBLE_VAULT;1.1;AES256
38653861336566343330626238373633373731326162373165653837346536383061373134653832
3937396232303433383439366630386632313137653236330a346336383531376635353765623132
37353933663161626536383933383366353331326434613466346363626135306239393134613031
3236666539393436310a3833333633866373738613230323038656465346438336538346538336466
35386638323830316235343631353933663236653637383839643861643332663061393432613637
33643465646363376338326233356234393837386461356531366534336230373734613466626565
61376535393964363534623436336235373335306239396630386561353862643035626463633530
36653230353963323066313661386462376131396331383437613163666164656234656261626161
32653864303931646135636163666130393234643663623465666333353562323762313461316530
3438326437396134366239646635323439393939306534613432
~
```

It is encrypted with password protected to edit or view

- To Edit or view encrypted playbook command:

```
[ansible@ip-172-31-36-113 ~]$ ansible-vault edit vault.yml
```

```
Password : technical
```

- To change password of encrypted playbook command:

```
[ansible@ip-172-31-36-113 ~]$ ansible-vault rekey vault.yml
```

```
Password : technical
```

```
Newpassword: technical1
```

- To encrypt existing playbook command:

```
[ansible@ip-172-31-36-113 ~]$ ansible-vault encrypt target.yml
```

```
New Vault password:
```

```
Confirm New Vault password:
```

Encryption successful

- To decrypt existing playbook command:

```
[ansible@ip-172-31-36-113 ~]$ ansible-vault decrypt target.yml
```

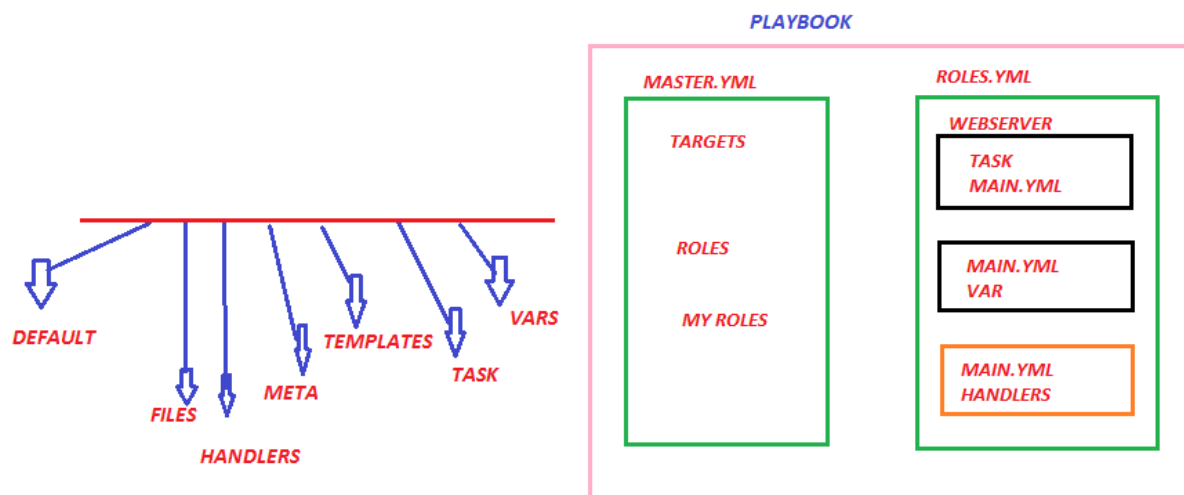
Vault password:

Decryption successful

ROLES

Roles let you automatically load related vars, files, tasks, handlers, and other Ansible artifacts based on a known file structure. After you group your content in roles, you can easily reuse them and share them with other users.

Its better for organizing tasks & encapsulating data needed to accomplish those tasks.



DEFAULT: It stores the data about roles/application. Default variable ex; if you want to run port 80 or 8080 then variables has to be defines in the path.

FILES: It contains files needed to be transferred to the remote vm (static files)

Handlers: They are triggered or task . we can segregate all the handlers required in the playbook.

META: This directory contains files that establish roles dependences . Author name , size, times etc. Its basically data of data.

TASK: It contains all the task that is normally in this playbook. Ex: packages , files etc

VARS: Variables for the roles can be specified in this directory and used in configuration file. Both vars and default entire variables.

```
roles/
  common/                # this hierarchy represents a "role"
    tasks/               #
      main.yml           # <-- tasks file can include smaller files if warranted
    handlers/            #
      main.yml           # <-- handlers file
    templates/           # <-- files for use with the template resource
      ntp.conf.j2        # <----- templates end in .j2
    files/               #
      bar.txt            # <-- files for use with the copy resource
      foo.sh             # <-- script files for use with the script resource
    vars/                #
      main.yml           # <-- variables associated with this role
    defaults/            #
      main.yml           # <-- default lower priority variables for this role
    meta/                #
      main.yml           # <-- role dependencies
    library/             # roles can also include custom modules
    module_utils/        # roles can also include custom module_utils
    lookup_plugins/      # or other types of plugins, like lookup in this case

  webtier/               # same kind of structure as "common" was above, done for the webtier role
  monitoring/            # ""
  fooapp/                # ""
```

LAB

Install tree

```
[ansible@ip-172-31-36-113 ~]$ sudo yum install tree
```

Loaded plugins: extras_suggestions, langpacks, priorities, update-motd

Complete!

```
[ansible@ip-172-31-36-113 ~]$ tree
```

```
.
├── condition.yml
├── copiedfromserver
├── file1
├── handlers.yml
├── loops.yml
├── playbook
│   └── roles
│       └── webserver
│           └── tasks
├── target.yml
├── task.yml
├── technicalguftugu
├── vars.yml
└── vault.yml
```

```
[ansible@ip-172-31-36-113 ~]$ cd playbook/
```

```
[ansible@ip-172-31-36-113 playbook]$ tree
```

```
.
├── roles
│   └── webserver
│       └── tasks
```

```
[ansible@ip-172-31-36-113 playbook]$ touch roles/webserver/tasks/main.yml
```

```
[ansible@ip-172-31-36-113 playbook]$ vi roles/webserver/tasks/main.yml
```

```
~ ansible@ip-172-31-36-113 /playbook
```

```
- name: install apache on RedHat
  yum: pkg=httpd state=latest
```

```
~
```

```
:wq
```

```
[ansible@ip-172-31-36-113 playbook]$ tree
```

```
.
├── master.yml
├── roles
│   └── webserver
│       └── tasks
│           └── main.yml
```

```
3 directories, 2 files
```

```
[ansible@ip-172-31-36-113 playbook]$ vi master.yml
```

```
--- # master playbook for webserver
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  roles:
    - webserver
```

```
~
```

```
:wq
```

```
[ansible@ip-172-31-36-113 playbook]$ ansible-playbook master.yml
```

```

    unreachable=0    failed=0    skipped=0    rescue
    d=0      ignored=0

[ansible@ip-172-31-36-113 playbook]$ ansible-playbook master.yml

PLAY [demo] *****

TASK [Gathering Facts] *****
[WARNING]: Platform linux on host 172.31.36.241 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information.
ok: [172.31.36.241]
[WARNING]: Platform linux on host 172.31.33.44 is using the discovered Python interpreter at
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information.
ok: [172.31.33.44]

TASK [webserver : install apache on RedHat] *****
ok: [172.31.36.241]
ok: [172.31.33.44]

PLAY RECAP *****
172.31.33.44           : ok=2    changed=0    unreachable=0    failed=0    skipped=0    re
    ignored=0
172.31.36.241         : ok=2    changed=0    unreachable=0    failed=0    skipped=0    re
    ignored=0

```

THANK YOU

MOHD ASEEM AKRAM

DEVOPS ENGINNER

9666192490

RESOURCES :

1. Bhupinder Rajput info@technicalguftgu.in

2. https://docs.ansible.com/ansible/2.9/modules/list_of_all_modules.html