

ANSIBLE

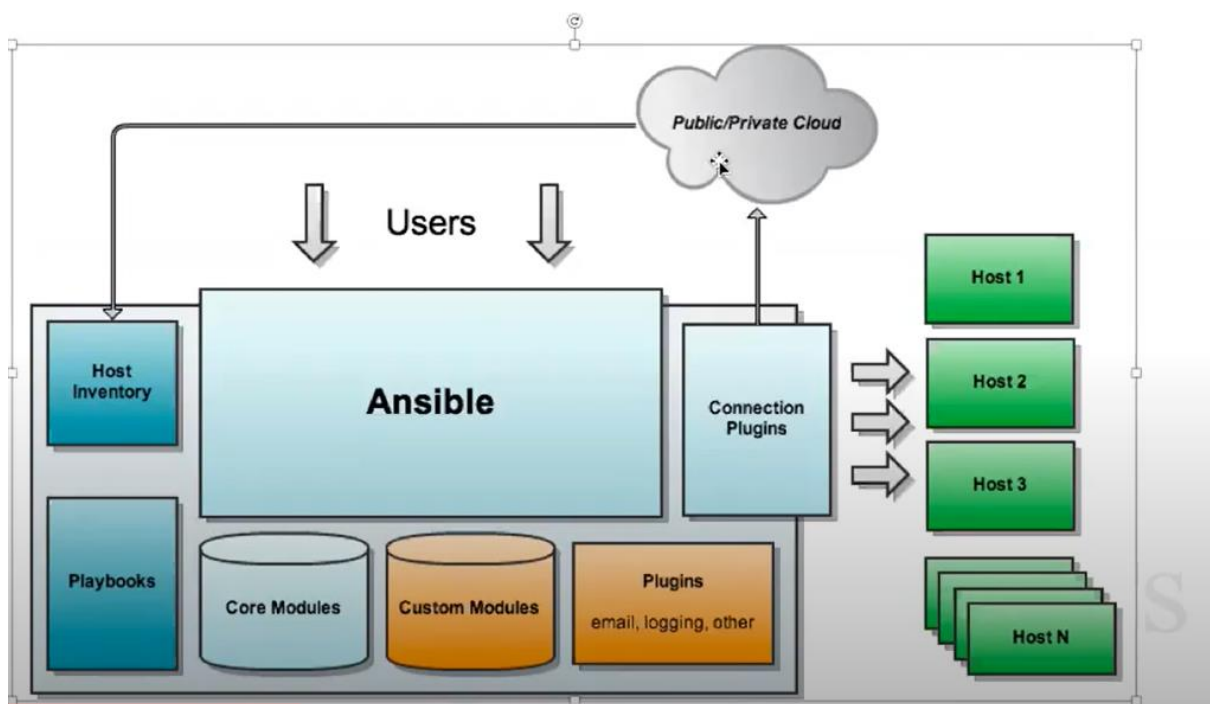
(Configuration Management System)

Ansible--> Configuration Management Tool And Also We can use as a Deployment Tool.

Configuration is some task which we want to execute on the Server. It can be

- 1) Creating Users/Groups..etc
- 2) Installing a software/package.
- 3) Creating/Updating/Copying files.
- 4) Start/Stop Services ..etc

Ansible Architecture



Ansible all -m ping

It will check the ansible machine whether it is connecting to the host machine (ip address given in the /etc/ansible/hosts in this path)

vi /etc/ansible/hosts

```
172.31.8.97 ansible_user=ec2-user ansible_ssh_private_key_file=~/.DevOpsTwo.pem
```

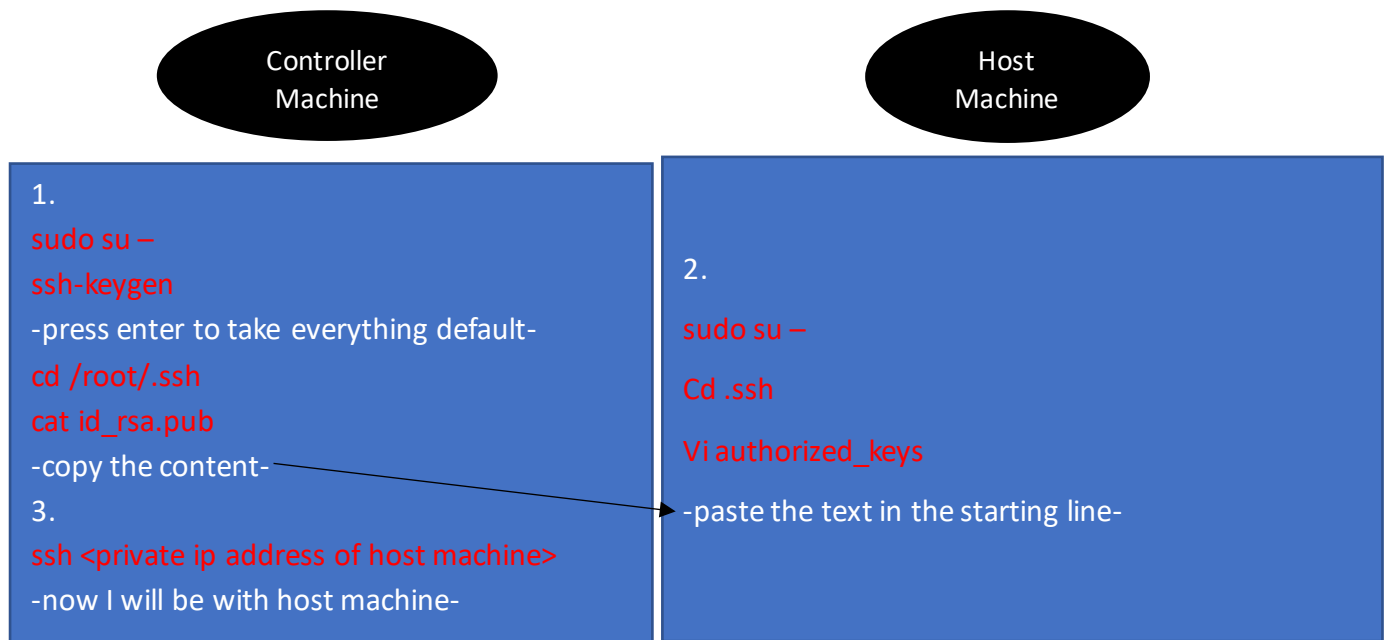
```
172.31.7.45 ansible_user=ec2-user ansible_ssh_private_key_file=~/.DevOpsTwo.pem
```

We can add ip's and which user to configure and mention the specific pem file path to configure

Chmod 400 /xxx.pem

We need to provide the permission to the pem file

First of all lets see how to connect host machine and control it from controller machine



Installing ansible to the controller machine:

```
Apt-get update
apt-add-repository ppa:ansible/ansible
apt-get update
apt-get install ansible
ansible --version
```

There are two ways we can add inventory

1.global 2.local

1.global

```
cd /etc/ansible/hosts
```

-by default in hosts all the inventory will be stored in global-

2.local

Refer ppt

```
cd /home/ubuntu
```

```
mkdir ansible
```

```
cd ansible/
```

```
vi inventory.txt
```

```
testweb ansible_host=<host private ip address>
```

-here testweb will be our host name-

```
ansible -m ping testweb -i inventory.txt
```

-this is to ping weather the local host is reachable or not-

https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html#rebooting-servers

-here we will get ad hoc commands by these commands we can configure host system like managfiles, managing packages, managing user or groups.....etc

```
ansible testweb -m ansible.builtin.copy -a "src=/home/ubuntu text.txt dest= home/ubuntu " -i inventory.txt
```

-this is to copy text file from controller machine to host machine

PLAYBOOK

-we use yml script to do playbooks

Here we will do write all task to yml which we need to configer in host machine like installing apache and run it and create directory and copying the file etc-

In controller machine:

```
sudo su
```

```
cd ansible
```

```
vi apache.yml
```

-copy the content we need in it..... for reference check the file he shared-

```
ansible-playbook apache.yml -i inventory.txt
```

-here it is to execute the yml script... here apache.yml is the script written by us-

ROLES

role is a module i.e if playbook is lengthy so we can split into multiple tasks

```
cd /etc/ansible/
```

```
vi hosts
```

```
[webservers]
```

<host private ip address>

-here webserver and ip address should add in the bottom in the hosts file-

cd role

-here we need to create roles over here-

ansible-galaxy init apache --offline

-this is the command it will create the role structure over here, init-initialization, apache-name of the role, it will create in offline-

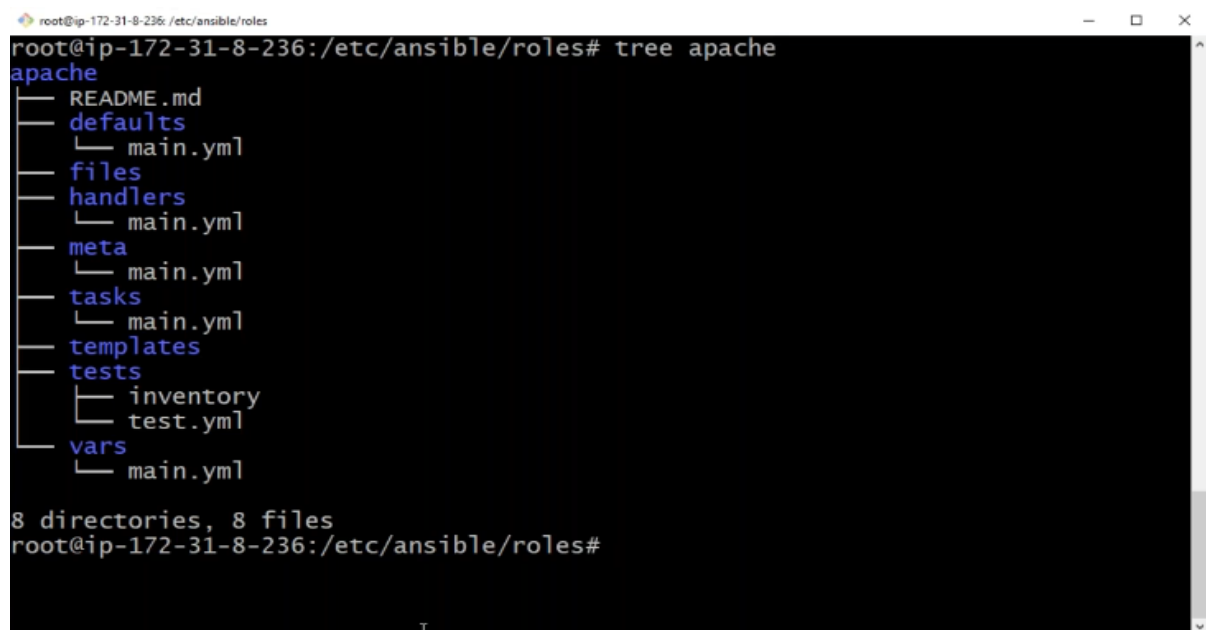
apt-get update

apt-get install tree

-tree is a linux command which it will show the structure-

tree apache

-it will show the tasks created for the Apache



```
root@ip-172-31-8-236: /etc/ansible/roles
root@ip-172-31-8-236:/etc/ansible/roles# tree apache
apache
├── README.md
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
├── vars
│   └── main.yml
8 directories, 8 files
root@ip-172-31-8-236:/etc/ansible/roles#
```

-we should create yml according like tasks, vars, test separately and place in the consequent structure separately-

-here need to check the 4th class video for adding roles and tasks-

cd..

-now we are with ansible folder-

vi site.yml

-hosts: webservers

become: true

roles:

-apache

-here webservers is which we have mentioned private ip address of the host and we have written roles as apache, and we can add more roles as we need.. that is the main advantage of the role, we can do multiple roles at a time.-

ansible-playbook site.yml --syntax-check

-here it will do only syntax check for site.yml

ansible-playbook site.yml --check

it will just dry Run (trail) that playbook is working or not

ansible-playbook site.yml -v

-it will execute the roles in with site.yml, **here -v shows detail logs of the task in detail and -vvv more details**

ansible-vault encrypt inventory.txt

-it will encrypt the file by asking g to create password for doing it-

ansible-playbook apache.yml -I inventory.txt --ask-vault-pass

-it will execute the ansible by asking vault password for the inventory.txt-

ansible-doc-l

it will list all the core modules

ansible-doc <moduleName>

it will show detail about that module

ansible <all/groupname/hostname/IP> -m <module> -a <args>

format of modules in single line

ansible all -m shell -a "cat /etc/*release"

here shell will run the all the linux commands in the servers given in the host inventory and show the output

ansible all -m shell -a "name=git<version> state=present"

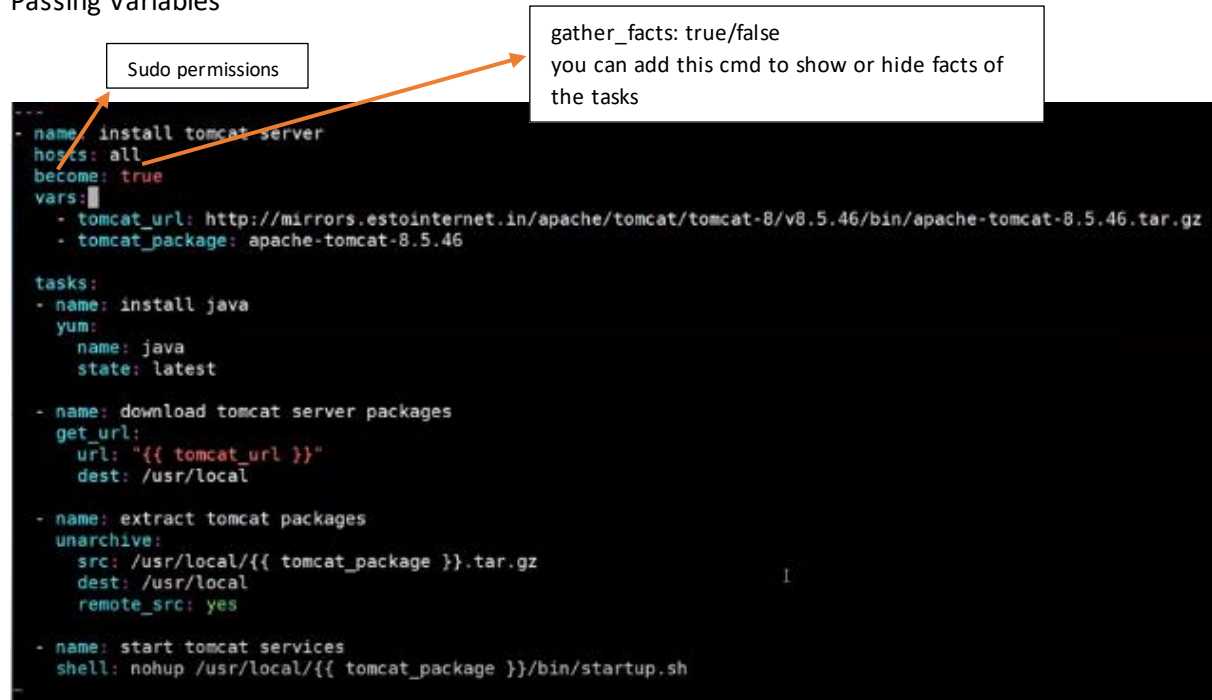
here it will install git with the specific version

(present = install, absent = uninstall, latest=it will check package is latest or it will update it)

ansible all -m service -b -a "name=httpd state=<started/restarted/stopped>"

to start any service(here -b is for root privileges

Passing Variables



- yaml script which install tomcat in the node server-

here we can observe we can pass the variable through ansible-playbook

and we can use variable outside of our playbook by giving specific variable in separate file

vi tomcat_vars

```
tomcat_url: http://mirrors.estointernet.in/apache/tomcat/tomcat-8/v8.5.46/bin/apache-tomcat-8.5.46.tar.gz
tomcat_package: apache-tomcat-8.5.46
```

here we can add all the variable which we need in yaml

```
---
- name: install tomcat server
  hosts: all
  become: true
  vars_files:
    - tomcat_vars

  tasks:
    - name: install java
      yum:
        name: java
        state: latest

    - name: download tomcat server packages
      get_url:
        url: "{{ tomcat_url }}"
        dest: /usr/local

    - name: extract tomcat packages
      unarchive:
        src: /usr/local/{{ tomcat_package }}.tar.gz
        dest: /usr/local
        remote_src: yes

    - name: start tomcat services
      shell: nohup /usr/local/{{ tomcat_package }}/bin/startup.sh
```

now here in Yaml file we need to specify the variable file. in our case It is tomcat_vars

now we can execute the Yaml script

```

- hosts: webservers
  become: True
  tasks:
    - name: Install packages
      yum:
        name: "httpd"
        state: "present"
      tags:
        - install
    - name: Start Apache server
      service:
        name: httpd
        state: started
        enabled: True
      tags:
        - start
    - name: Deploy static website
      copy:
        src: index.html
        dest: /var/www/html/
      tags:
        - deploy

```

here we can add tags so we can run only the specific task according to the tag

```

[ec2-user@ip-172-31-1-94 ~]$ ansible-playbook -i dev.inv --private-key=appkey.pem apache.yml --tags=install,start
PLAY [webservers] *****
TASK [Gathering Facts] *****
ok: [172.31.13.194]
TASK [Install packages] *****
ok: [172.31.13.194]
TASK [Start Apache server] *****
ok: [172.31.13.194]
PLAY RECAP *****
172.31.13.194 : ok=3  changed=0  unreachable=0  failed=0

```

while running ansible playbook command here we can give --tag=<tag1,tag2>

so it will run the only the task with the specific tag

ansible-playbook <ymlFile> --list-host

It will list the IPs on which the yml file executing on it