Class 8 Recording Video – from 47th min

Ansible: is an agent less configuration management tool.

Here code is placed in a central repository called controller and this controller is responsible for pushing the code to different nodes connected to it.

It uses push approach means the controller pushes the code to the nodes connected to it.

Here we used different authentication mechanisms.

1)By using ssh

2)Kerberose

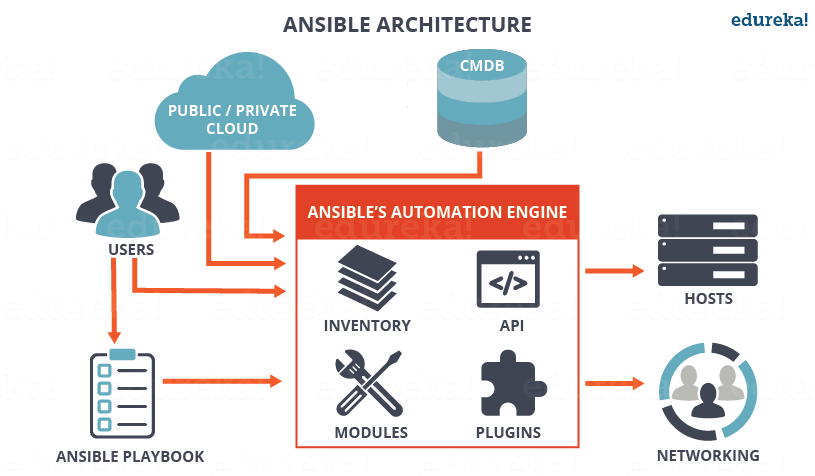
**Ansible Terms:**

* **Controller Machine: The** machine where Ansible is installed, responsible for running the provisioning on the servers you are managing.
* **Inventory:** An initialization file that contains information about the servers you are managing.
* **Playbook:** The entry point for Ansible provisioning, where the automation is defined through tasks using YAML format.
* **Task:** A block that defines a single procedure to be executed, e.g. Install a package.
* **Module:** A module typically abstracts a system task, like dealing with packages or creating and changing files. Ansible has a multitude of built-in modules, but you can also create custom ones.
* **Role:** A pre-defined way for organizing playbooks and other files in order to facilitate sharing and reusing portions of a provisioning.
* **Play:** A provisioning executed from start to finish is called a play. In simple words, execution of a playbook is called a play.
* **Facts:** Global variables containing information about the system, like network interfaces or operating system.
* **Handlers:** Used to trigger service status changes, like restarting or stopping a service.

Ansible is a helpful tool that allows you to create groups of machines, describe how these machines should be configured or what actions should be taken on them. Ansible issues all commands from a central location to perform these tasks.

No other client software is installed on the node machines. It uses SSH to connect to the nodes. Ansible only needs to be installed on the control machine (the machine from which you will be running commands) which can even be your laptop. It is a simple solution to a complicated problem.

Architecture:



As you can see, in the diagram above, the Ansible automation engine has a direct interaction with the users who write playbooks to execute the Ansible Automation engine. It also interacts with cloud services and Configuration Management Database (CMDB).

The Ansible Automation engine consists of:

* **Inventories:** Ansible inventories are lists of hosts (nodes) along with their IP addresses, servers, databases etc. which needs to be managed. Ansible then takes action via a transport – SSH for UNIX, Linux or Networking devices and WinRM for Windows system.
* **APIs:** APIs in Ansible are used as transport for Cloud services, public or private.
* **Modules:** Modules are executed directly on remote hosts through playbooks. The modules can control system resources, like services, packages, or files (anything really), or execute system commands. Modules do it by acting on system files, installing packages or making API calls to the service network. There are over 450 Ansible-provided modules that automate nearly every part of your environment. For e.g.
  + Cloud Modules like cloudformation which creates or deletes an AWS cloud formation stack;
  + Database modules like mssql\_db which removes MYSQL databases from remote hosts.
* **Plugins:** Plugins allows to execute Ansible tasks as a job build step. Plugins are pieces of code that augment Ansible’s core functionality. Ansible ships with a number of handy plugins, and you can easily write your own. For example,
  + Action plugins are front ends to modules and can execute tasks on the controller before calling the modules themselves.
  + Cache plugins are used to keep a cache of ‘facts’ to avoid costly fact-gathering operations.
  + Callback plugins enable you to hook into Ansible events for display or logging purposes.

There are a few more components in Ansible Architecture which are explained below:

**Networking**: Ansible can also be used to automate different networks. Ansible uses the same simple, powerful, and the agentless automation framework IT operations and development are already using. It uses a data model (a playbook or role) that is separate from the Ansible automation engine that easily spans different network hardware.

**Hosts**: The hosts in the Ansible architecture are just node systems which are getting automated by Ansible. It can be any kind of machine – Windows, Linux, RedHat etc.

**Playbooks:** Playbooks are simple files written in YAML format which describes the tasks to be executed by Ansible. Playbooks can declare configurations, but they can also orchestrate the steps of any manual ordered process, even if it contains jump statements. They can launch tasks synchronously or asynchronously.

**CMDB** : It is a repository that acts as a data warehouse for IT installations. It holds data relating to a collection of IT assets (commonly referred to as configuration items (CI)), as well as to describe relationships between such assets.

**Cloud:** It is a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server. You can launch your resources and instances on cloud and connect to your servers

**Modules:** Modiles are like command centre for Ansible. Using modules we are doing installation/configuration on our remote machines. When we download ansible by default 432 modules will be installed automatically. These modules are like checking the uptime, running some commands, starting servers etc.

**Inventory:** List or group of servers on which our configuration will be deployed.

Lets say if we want to deploy software on 50 servers and out of these 50, 10 are dev , 10 are qa and 30 are prod and if we want to deploy only in prod then we group all servers under inventory and we push the configuration.

Default location of inventory file: /etc/ansible/hosts

Eg:

[QASevers] --Groupname

Server1 --servername1

Server2 --servername2

**Play book:** in ansible we write our code in YAML language. We need to follow identitaion .

It is a glue which connects inventory and module.

Inside playbook we define plays. Inside plays we define tasks.

**Play:** A provisioning executed from start to finish is called a play. In simple words, execution of a playbook is called a play.

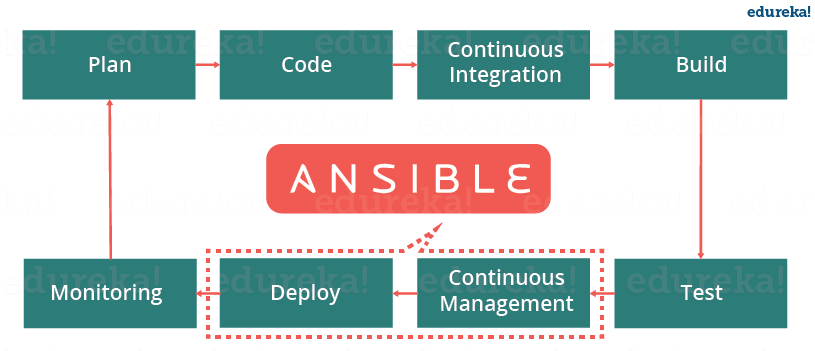
**Task:** A block that defines a single procedure to be executed, e.g. Install a package.

**Ansible Config:** It is a configuration file where global settings are placed like log locations , location of an inventory file, modules location etc.

Note: Python should be installed before installing ansilbe. And the controller should be installed in linux/Centos and node can be in any machine like windows, mac etc.

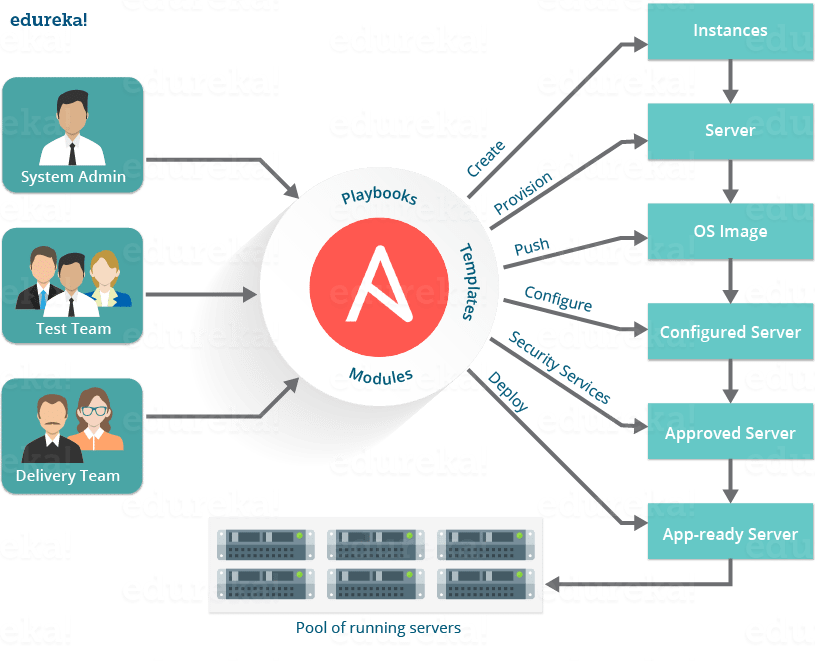
**Ansible in DevOps**

In DevOps, as we know development and operations work is integrated. This integration is very important for modern test-driven application design. Hence, Ansible integrates this by providing a stable environment to both development and operations resulting in smooth orchestration. Refer to the image below to see how Ansible fits into DevOps:



Let us discuss now how Ansible manages the entire DevOps infrastructure. When developers begin to think of infrastructure as part of their application i.e as Infrastructure as code (**IaC**), stability and performance become normative. Infrastructure as Code is the process of managing and provisioning computing infrastructure (processes, bare-metal servers, virtual servers, etc.) and their configuration through machine-processable definition files, rather than physical hardware configuration or the use of interactive configuration tools. This is where Ansible automation plays a major role and stands out among its peers.

In DevOps, Sysadmins work tightly with developers, development velocity is improved, and more time is spent doing activities like performance tuning, experimenting, and getting things done, and less time is spent fixing problems. Refer to the diagram below to understand how the tasks of sysadmins and other users are simplified by Ansible.



**Installation of ansible:**

**In controller/master machine:**

Go to root directory: sudo -i

Command to install: yum install -y ansible

Instead of using root user we are creating a new user and we are using that for configuration because in all the time we don’t get root user to create the tasks i..e we don’t get admin rights for all the nodes in general.

Ansible uses public private key sharing i.e. whatever the user we are using here in controller, we use the same user in node machine also.

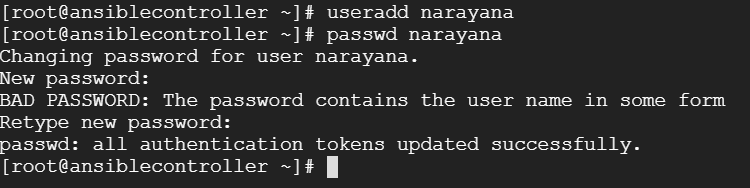
To see the users in linux:

Vi /etc/passwd

Now create user in controller machine:

Useradd narayana -- To create user [narayana-username]

Passwd narayana -- TO create password for user narayana



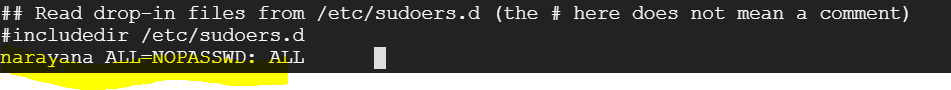
Go to Root directory:

Sudo -i

Then add the below line at the end of the file in the sudoers file and save

Vi /etc/sudoers

narayana ALL=NOPASSWD: ALL



Then restart the service: systemctl restart sshd

Now switch to user created above:

To switch user : su - narayana [narayana - username

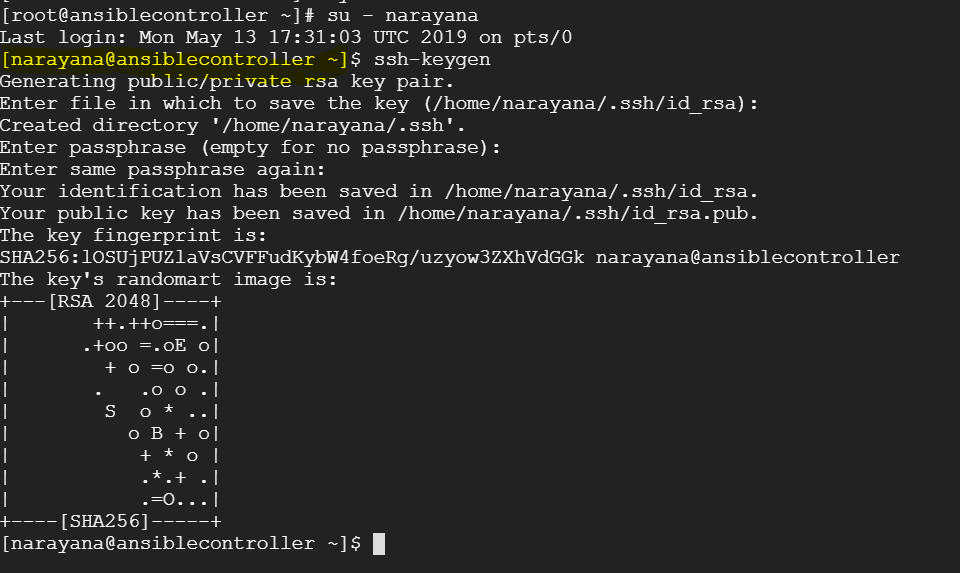


Follow the above steps for creating the same user in node machine also.

Then swith user : su – narayana

Go to Controller machine and run the below command to generate the ssh key:

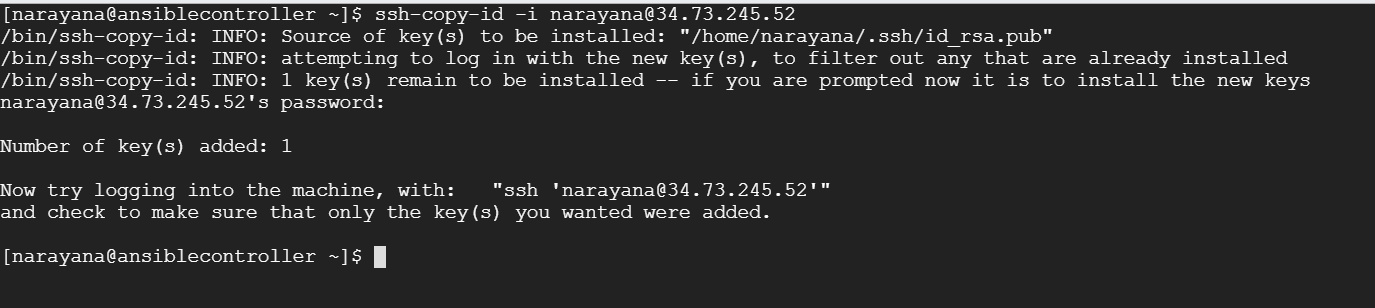
Generate the ssh key: ssh-keygen



Once the ssh key generated , to copy the files

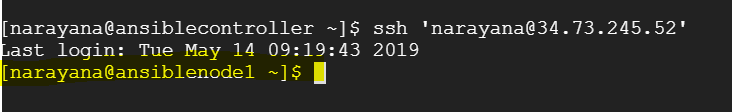
Ssh-copy-id -I narayana@ipaddressofslavenode

Ssh-copy-id -I narayana@34.73.245.52



To check the node added run the below from the above message

ssh 'narayana@34.73.245.52'



Now we see the machine connected to slave node

**Now create a group in inventory file:**

Go to location in controller: cd /etc/ansible/

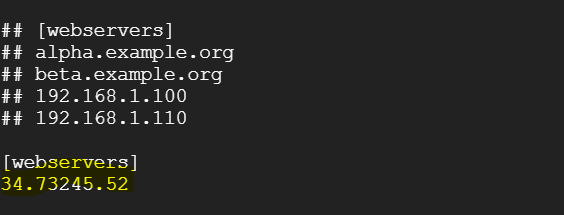
Ls

Sudo vi hosts

Add the below in the hosts file

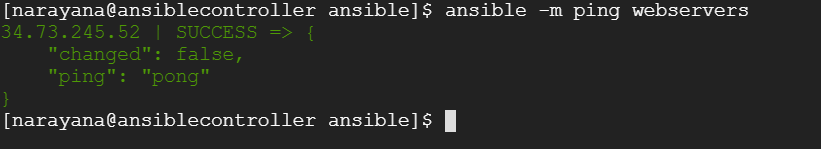
[webservers]

Ipaddress of slave node



**To check which servers the controller connects**

Ansible -m ping webservers



**To see the documentation of a module:**

Ansible-doc modulename

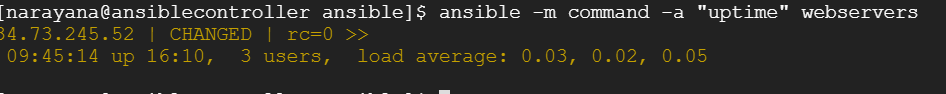
Example: ansible-doc ping

**To see all the modules:**

Ansible-doc l

**To find how long the slave machine is up/uptime of slave :**

Ansible -m command -a “uptime” webservers



**To install httpd package on the slave node from controller:**

Ansible -m yum -a “name=httpd state=present” webservers -s

**To start the service of httpd package:**

Ansible -m service -a “name=httpd state=started” webservers -s

**To create a user in slave from master/controller:**

Ansible -m user -a “name=nari password=nari” webservers -s

**To check the user created in slave or not:**

Go to slave: vi /etc/passwd

**Class 9 Recording**

**To copy a file from remote server(controller) to slave nodes:**

Ansible -m copy -a “src=/tmp/file.txt dest=/tmp/file.txt” webservers

**To install GIT:**

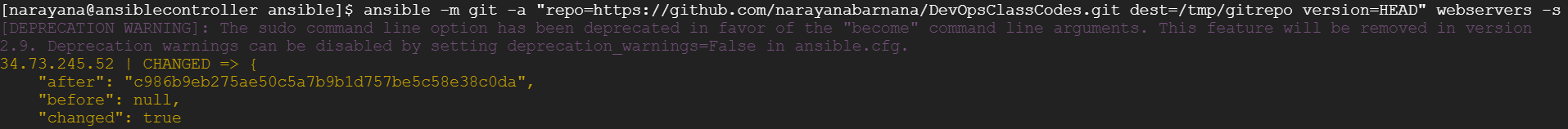
Ansible -m yum -a “name=git state=present” webservers -s

**To clone the git repository**

Ansible -m git -a “repo=gitrepositorypath dest=/tmp/gitrepo version=HEAD webservers -s

Here version means branch and HEAD Means master branch. If you want to clone other than master instead of HEAD give the name of the branch name.

Example: ansible -m git -a “repo=https://github.com/narayanabarnana/DevOpsClassCodes.git dest=/tmp/gitrepo version=HEAD” webservers -s



If the ouptput is in yellow color tehn the build got success. If it is in red color then the build failed. If it is in green color nothing has changed.

To validate whether the git repo master branch is installed in slave or not

Go to cd /tmp

Ls

You will see git repo

**To see the file system usage on slave node:**

Ansible -m command -a “df -kh” webservers

To see how the communication is happening betweem controller and slave:

Ansible -m setup -a webservers

**To filter the results of the above command:**

Ansible -m setup -a “filter=ansible\_user” webservers.

Note: All the above command are adhoc commands i.e. individual commands

If we want to execute multiple tasks like installing the software, starting the service we create playbooks instead of executing the adhoc commands(individual language) which will be written in YAML.

**Creation of Playbook:**

**Scenario is:** Installing the httpd package and start the service.

---

- hosts: webservers

become: true

become\_user: root

tasks:

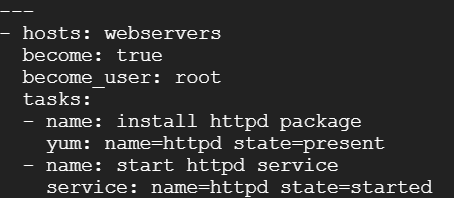
- name: install httpd package

yum: name=httpd state=present

- name: start httpd service

service: name=httpd state=started

Save the plabook and quit



If you want to run the playbook with some other user i.e the user which you are not logged in the use the below two lines

become: true

become\_user: root

here we are running the playbook with root user so we have given that. If we want to run the playbook with other user then give that username.

**To verify the playbook syntax is correct or not:**

Ansible-playbook –syntax -check plabookname

Ansible-playbook –syntax -check test1.yml

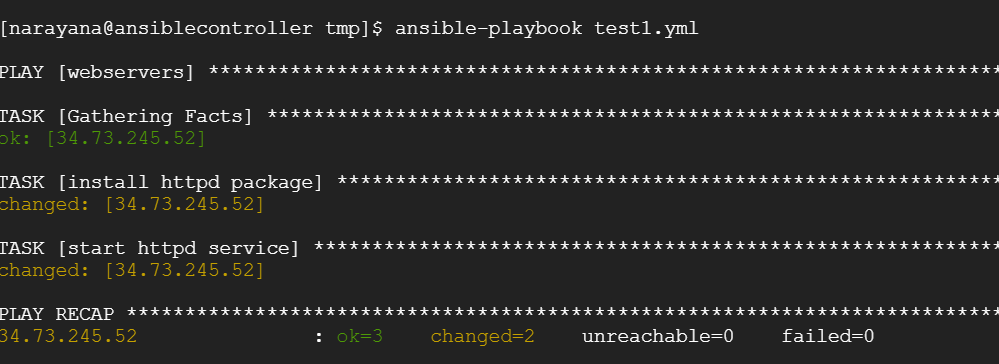


**To run the playbook:**

Ansible-playbook test1.yml



Once you run the playbook it will push the configuration from controller to the node machines which are connected to it.



**Scenario: How to run the shell file in ansible**

Create a playbook which contains the below

1)copy the shell file from controller to slave node

2)Then execute the shell file

Solution:

1)create a shell file in the /tmp directory

Sudo vi test.sh

Inside the test.sh write the below code

Sudo yum install -y git

2)create a playbook with name test2.yml

Sudo vi test2.yml

Enter the below code in the file

---

- hosts: webservers

become: true

become\_user: root

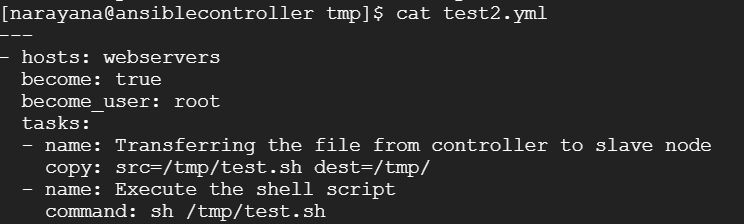
tasks:

- name: Transferring the file from controller to slave node

copy: src=/tmp/test.sh dest=/tmp/

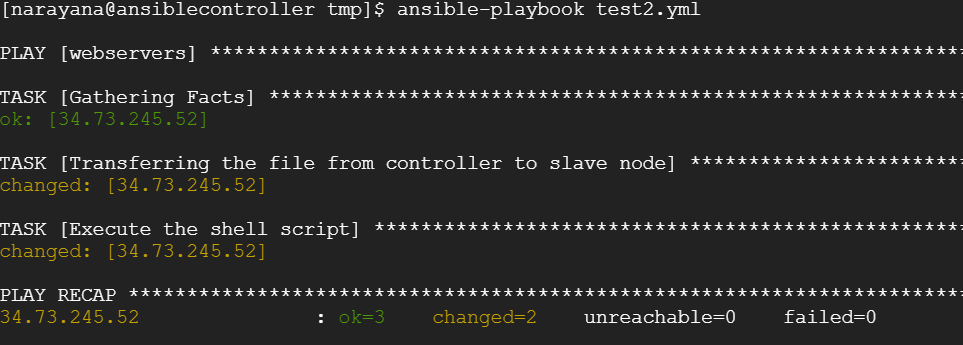
- name: Execute the shell script

command: sh /tmp/test.sh



Validate the syntax of the playbook: ansible-playbook –syntax -check test2.yml

Run the playbookk ansibel-playbook test2.yml



How to setup dependency parameter in ansible?

To download the file from web in linux

Command: wget linkaddress



**Scenario: whenever the tomcat package gets deployed , the service will restart automatically.**

i.e. Whenever the application(war file) gets deployed the tomcat service will restart automatically.

By using notify and handlers we can achieve this scenario.

**Notify and handlers:**

1)Notify and handlers are a sort of dependency parameters

2)Calling particular task under handler is called Notify

3)Execution of the task based on the notify.

Code:

---

- hosts: webservers

become: true

become\_user: root

tasks:

- name: Install Tomcat package

yum: name=tomcat state=latest

- name: Start tomcat service

service: name=tomcat state=started

- name: deploy sample.war file

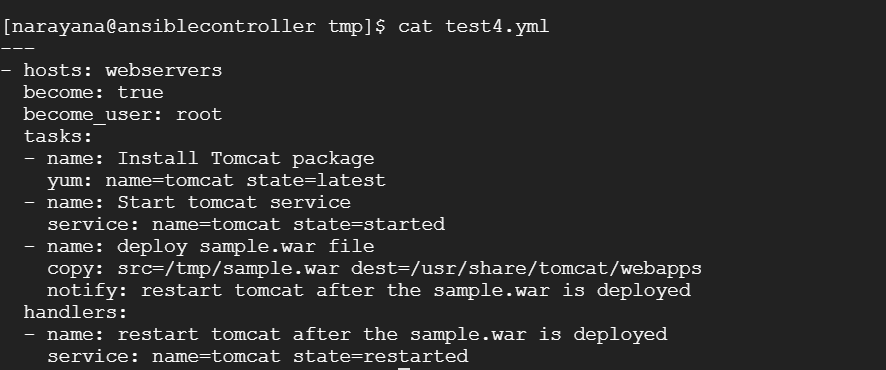
copy: src=/tmp/sample.war dest=/usr/share/tomcat/webapps

notify: restart tomcat after the sample.war is deployed

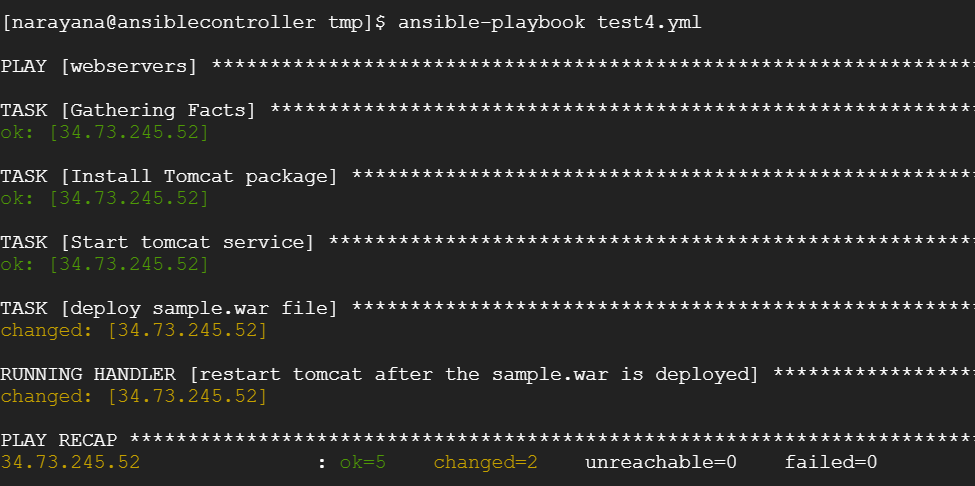
handlers:

- name: restart tomcat after the sample.war is deployed

service: name=tomcat state=restarted



Validate the syntax and run the playbook



Tomcat has a specific location. If we are using tomcat as a webservice then the tomcat files are stored in specified below location.

/usr/share/tomcat/webapps

When the above sample.war file gets downloaded then only the tomcat service gets restarted otherwise it should not.

Notify and handler name should have same name.

**How to define variables in ansible:**

---

- hosts: webservers

become: true

become\_user: root

vars:

source\_path: /etc/ansible

dest\_path: /var/www/html

tasks:

- name: Install httpd Package

yum: name=httpd state=present

- name: start httpd service

service: name=httpd state=started

- name: using variables

copy: src={{source\_path}}/index.html dest={{dest\_path}}

notify: restart httpd

handlers:

- name: restart httpd

service: name=httpd state=restarted

Note:

using vars we can declare variables as global

and by using {{variable\_name}} we can use the variable

**Roles in Ansible:**

With ansible roles we can group our variables, tasks, handlers etc.

Ansible roles are similar to modules in puppet and cookbook in chef.

In order to create roles, we use the ansible-galaxy command which has all the templates to create it.

We create roles in the specific directory

Default directory: /etc/ansible/roles

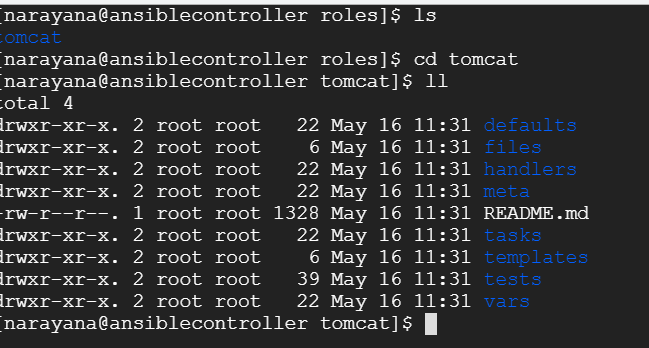
To create a custom role use the below command

Command: sudo ansible-galaxy init rolename

Eg: sudo ansible-galaxy init tomcat

When you run the command it will create an ansible role with tomcat.

Go to tomcat folder which created above and you can see different files inside it.



The main components we will use in this section include:

* tasks/main.yml – This is the starting point for tasks created for the role. You can use the main.yml file to point to the other task files.
* Tasks: This is to define the tasks
* vars – This is to define any variables used.
* meta – This is to define information about yourself or the author.

Class 10th Recording:

Go to tasks and enter the below in main.yml

- name: Install Tomcat package

yum: name=tomcat state=latest

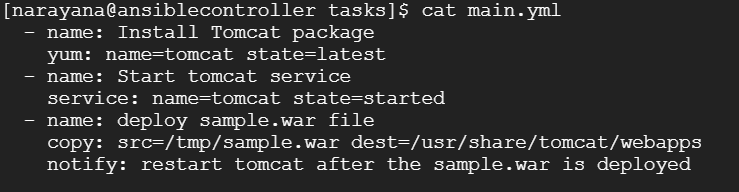
- name: Start tomcat service

service: name=tomcat state=started

- name: deploy sample.war file

copy: src=/tmp/sample.war dest=/usr/share/tomcat/webapps

notify: restart tomcat after the sample.war is deployed



Go to Files folder and the enter the below code in yml file

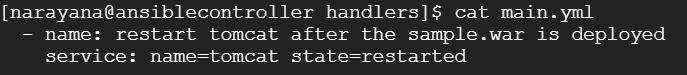
Sudo copy /sample.war .



Go to handlers file and edit the yml file

- name: restart tomcat after the sample.war is deployed

service: name=tomcat state=restarted



Run the below command to copy the sample.war file from /tmp folder to present directory i.e. roles

sudo cp /tmp/sample.war .

Go to Ansible folder

And create an site.yml file and write the below code

---

- hosts: webservers

become: true

become\_user: root

vars:

jenkins\_http\_port: 9000

roles:

- tomcat

- geerlingguy.java

- geerlingguy.jenkins



Run the playbook

**To download the roles from galaxy.com**

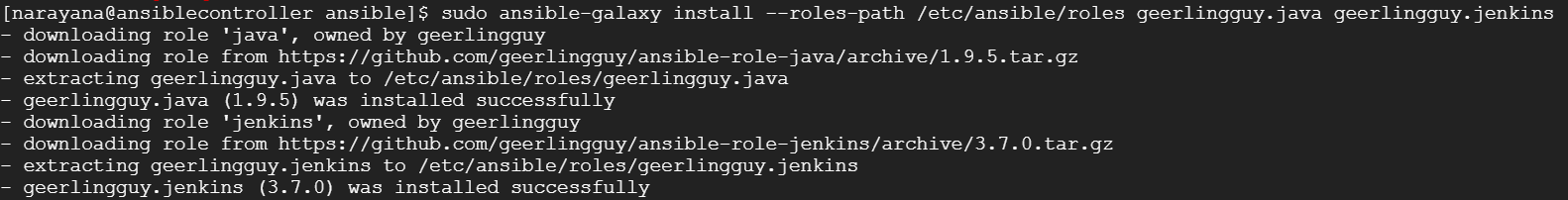
Go to website: <https://galaxy.ansible.com/>

Search for the role like java , jenkins

Run the command in controller

Command: sudo ansible-galaxy install –roles-path /etc/ansible/roles geerlingguy.java geerlingguy.jenkins

--roles-path:means we are defining the path of the role and the path is /etc/ansible/roles



Ansible Tower:

Important links:

<https://www.edureka.co/blog/what-is-ansible/>

<https://www.redhat.com/en/blog/integrating-ansible-jenkins-cicd-process>

<https://www.softwaretestinghelp.com/ansible-playbooks-ansible-vaults/>

<https://www.softwaretestinghelp.com/ansible-roles-jenkins-integration-ec2-modules/>

<https://github.com/ansible/ansible-examples/tree/master/tomcat-standalone>

<https://www.middlewareinventory.com/blog/category/ansible/>

Interview Questions:

What is the difference between command module and shell module in ansible?

Command module will execute only one command i.e. pwd. Whereas shell module will execute multiple commands at a time.

Cat sample.txt | grep -i sample