

Ansible By Rajesh Singamsetti



ANSIBLE

Content:

Intro to ansible

Setting up ansible on virtualbox or linux or cloud

Introduction yaml

Inventory files

Playbooks

Variables

Conditionals

Loops

Roles

What is ansible:

Ansible maintain by redhat and developed by using python language.

Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling infrastructure as code. It runs on many Unix-like systems, and can configure both Unix-like systems as well as Microsoft Windows.

Ansible is an push based mechanism.

Ansible we can write yaml files.

Reference pdf: <http://people.redhat.com/mskinner/rhug/q2.2017/Ansible-Hands-on-Introduction.pdf>
<https://docs.ansible.com/ansible/latest/index.html>

Setup Ansible (installation Ubuntu)

Take 2 servers one is ansible master other one is ansible slave (i am taking 2 ubuntu servers)

Launch servers.

ref:https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html#installing-ansible-on-ubuntu

```
sudo apt-get update -y
sudo apt install software-properties-common
sudo apt-add-repository --yes --update ppa:ansible/ansible
sudo apt install ansible
```

To check python version --- python3 --version

ansible --version (for ansible shows the version)

from above steps the ansible configuration is done.

Now we have to make ssh connection between the node and server.

```
vi /etc/hosts
```

```
private ip address server1.com server1
```

```
Private ip address node1.com node1
```

Do the same operations in both server and node.

After going to aws console check server and node actions click on reboot.

```
cd .ssh
```

ssh-keygen (it will generate keys)

before generate ssh-keygen go to

```
vi /etc/ssh/sshd_config
```

-----continued-----

In this file
Permit root login will be yes &
PasswordAuthentication yes
systemctl restart sshd (its used for the restart a ssh)
Now in the position of .ssh type ssh-keygen
Is now u wil see the public key and private key
vi public key (copy that one)

Login into node server;

cd .ssh
vi authorized_keys
To go for end of ssh file type shift +a
Paste the public key of ansible server in node..
Save the file.
In node go
vi /etc/ssh/sshd_config
In this file
Permit root login will be yes &
PasswordAuthentication yes
systemctl restart sshd (its used for the restart a ssh)
After ssh restart on node go to main server console.
and sudo su
ssh give here node server ip address.

type yes

Now it will login on node server.

-----for this succesfully we made ssh connection between server to
node-----

Master server inventory containn all the ip address of nodes.

cd /etc/ansible/

vi hosts

[webserver]

172.31.38.220 (its is a private ip address of node like how many nodes present give step by step)

ansible -m ping all

Sample will be like this:

```
172.31.38.220 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/libexec/platform-python"
  },
  "changed": false,
  "ping": "pong"
}
```

Ansible ON RHEL & CENTOS

`sudo su && cd`

`pwd`(now u r in root directory to install softwares)

`yum install wget` (its for download web packages in linux)

`yum info wget` (info command used to know complete information about that package)

`cat /etc/redhat-release` (to find version of redhat)

`rpm -Uvh https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm` (its for Upgrade package we downloaded that epel repo)

`yum install epel-release -y` --- its for downloading the epel package in our linux machine

`yum install python` (ansible develop by python so we need python if python not available means we will go with python 3)

If python not install means it will give suggestions like this below

There are following alternatives for "python": python2, python36, python38

Now we go with python3.

`yum install git python3 python3-pip ansible openssl -y` --- using this command we at a time installing the python and ansible)

Now it will download all the packages.

To check python version --- `python3 --version`

`ansible --version` (for ansible shows the version)

vi /etc/hosts
private ip address server1.com server1
Private ip address node1.com node1
Do the same operations in both server and node.
After going to aws console check server and node actions click on reboot.
cd .ssh
ssh-keygen (it will generate keys)
before generate ssh-keygen go to
vi /etc/ssh/sshd_config
In this file
Permit root login will be yes &
PasswordAuthentication yes
systemctl restart sshd (its used for the restart a ssh)
Now in the position of .ssh type ssh-keygen
Is now u wil see the public key and private key
vi public key (copy that one)

Login into node server;

cd .ssh
vi authorized_keys
To go for end of ssh file type shift +a
Paste the public key of ansible server in node..
Save the file.
In node go
vi /etc/ssh/sshd_config
In this file
Permit root login will be yes &
PasswordAuthentication yes
systemctl restart sshd (its used for the restart a ssh)
After ssh restart on node go to main server console.
and sudo su
ssh give here node server ip address.

type yes

Now it will login on node server.

-----for this succesfully we made ssh connection between server to node-----

Master server inventory contains all the ip address of nodes.

cd /etc/ansible/

vi hosts

[web server]

172.31.38.220 (its is a private ip address of node like how many nodes present give step by step)

ansible -m ping all

Sample will be like this:

```
172.31.38.220 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/libexec/platform-python"
  },
  "changed": false,
  "ping": "pong"
}
```

-----from above we are successfully learned ansible installation on ubuntu && rhel-----

Ansible Inventory

ssh nodepublicipaddress (it will ask yes make it yes)

mkdir test

cd test

cat > inventory.txt

target1 ansible_host=172.31.5.130 (if password is there) ansible_ssh_pass=mypassword

Ctrl +c to save the file

ansible target1 -m ping -i inventory.txt

We will see message like this:

```
target1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
```

**note if we have multiple nodes we can put in inventory file and we run

ansible target2 -m ping -i inventory.txt (now it will through error bcoz

ssh target1 ipaddress already we enable ssh using yes or no option but target2 ip we did not made any ssh connect for that solving issues we have 2 solutions 1. Make ssh 2ndtargetipaddress (allow yes) or -->

2 nd solution go to default ansible inventory path change settings.

path: vi /etc/ansible/ansible.cfg

/host_key (search for it) press enter

U will find #host_key_checking = False

Just remove # and save the file.

Now u try with target u will saw that success message.

ansible target1 -m ping -i inventory.txt

Below fill with theory of ansible inventory.

YAML

In ansible playbooks we can build by using yaml

Ansible playbooks are textfiles or configuration files are written in a particular format called yaml

WHAT IS YAML?

XML

```
<Servers>
  <Server>
    <name>Server1</name>
    <owner>John</owner>
    <created>12232012</created>
    <status>active</status>
  </Server>
</Servers>
```

JSON

```
{
  Servers: [
    {
      name: Server1,
      owner: John,
      created: 12232012,
      status: active,
    }
  ]
}
```

YAML

```
Servers:
- name: Server1
  owner: John
  created: 12232012
  status: active
```

YAML

Key Value Pair

```
Fruit: Apple
Vegetable: Carrot
Liquid: Water
Meat: Chicken
```

Array/Lists

```
Fruits:
- Orange
- Apple
- Banana

Vegetables:
- Carrot
- Cauliflower
- Tomato
```

Dictionary/Map

```
Banana:
  Calories: 105
  Fat: 0.4 g
  Carbs: 27 g

Grapes:
  Calories: 62
  Fat: 0.3 g
  Carbs: 16 g
```

Sample Yaml Files:

1)employee:

name: john

gender: male

age: 24

address:

city: edison

state: 'new jersey'

country: 'united states'

2)

employees:

-

name: john

gender: male

age: 24

-

name: sarah

gender: female

age: 28

List of Dictionary in Dictionary

Now try adding the pay information. Remember while `address` is a dictionary, `payslips` is an array of `month` and `amount`

Key/Property	Value												
name	john												
gender	male												
age	24												
address	...												
payslips	<table><tr><th>#</th><th>month</th><th>amount</th></tr><tr><td>1</td><td>june</td><td>1400</td></tr><tr><td>2</td><td>july</td><td>2400</td></tr><tr><td>3</td><td>august</td><td>3400</td></tr></table>	#	month	amount	1	june	1400	2	july	2400	3	august	3400
	#	month	amount										
	1	june	1400										
	2	july	2400										
3	august	3400											

Above Image Answer:

employee:

name: john

gender: male

age: 24

address:

city: edison

state: 'new jersey'

country: 'united states'

payslips:

-

month: june

amount: 1400

-

month: july

amount: 2400

-

month: august

amount: 3400

In playbook we can give what we want to do

Ansible playbooks

Simple Ansible Playbook

- Run command1 on server1
- Run command2 on server2
- Run command3 on server3
- Run command4 on server4
- Run command5 on server5
- Run command6 on server6
- Run command7 on server7
- Run command8 on server8
- Run command9 on server9
- Restarting Server1
- Restarting Server2
- Restarting Server3
- Restarting Server4
- Restarting Server5
- Restarting Server6
- Restarting Server7

Complex Ansible Playbook

- Deploy 50 VMs on Public Cloud
- Deploy 50 VMs on Private Cloud
- Provision Storage to all VMs
- Setup Network Configuration on Private VMs
- Setup Cluster Configuration
- Configure Web server on 20 Public VMs
- Configure DB server on 20 Private VMs
- Setup Loadbalancing between web server VMs
- Setup Monitoring components
- Install and Configure backup clients on VMs
- Update CMDB database with new VM Information

Playbook is a single yaml file (playbook.yaml)

Play Defines set of activities.

Task to be run on hosts

Task - An action to be performed on the host

- execute a command
- Run a script
- Install a package
- shutdown/restart

Sample playbook:

```
---
- hosts: webserver
  remote_user: root
  become: yes
  tasks:
    - name: install httpd
      yum: name=httpd state=installed
    - name: copy index.html file
      copy: src=index.html dest=/var/www/html
    - name: restart service
      service: name=httpd state=started
```

Running Ansible Playbook Commands

RUN

ansible

```
ansible <hosts> -a <command>
```

```
ansible all -a "/sbin/reboot"
```

```
ansible <hosts> -m <module>
```

```
ansible target1 -m ping
```

ansible-playbook

```
ansible-playbook <playbook name>
```



```
ansible-playbook playbook-webserver.yaml
```

cd test (inventory project exist on in this folder)
ansible -m ping all - i inventory.txt

It will ping all host which are present in our server
Or else u want add direct default all hosts follow below procedure

```
cd /etc/ansible/  
vi hosts  
[webserver]  
172.31.38.220 (its is a private ip address of node like how many nodes present give step by step)  
ansible -m ping all  
172.31.38.220 | SUCCESS => {  
  "ansible_facts": {  
    "discovered_interpreter_python": "/usr/libexec/platform-python"  
  },  
  "changed": false,  
  "ping": "pong"  
}
```

Like this it will display all records..

Now in this same folder create one sample file
Create
sample.yml

vi sample.yml

-

name: Test connectivity to target servers

hosts: all

tasks:

- name: ping test

ping:

Save and exit

To run ansible playbooks follow below command:

ansible-playbook sample.yml(playbookname) -i inventory.txt

Tips and Tricks of while Developing ansible playbooks

Develop playbooks using free tool ATOM IDE

Need to research on work from windows when we push that folder will be there in server need to focus if we required or comfortable with notepad++ or any other tool.

Practise: ansible playbooks.

Reference url: https://kodekloud.com/p/ansible-practice-test/?scenario=questions_ansible_playbook

Important ansible playbooks yml commands

```
- name: 'Stop the web services on web server nodes'  
hosts: web_nodes  
tasks:
```

```
- name: 'Stop the web services on web server nodes'  
  command: 'service httpd stop'
```

```
- name: 'Shutdown the database services on db server nodes'  
hosts: db_nodes  
tasks:
```

```
- name: 'Shutdown the database services on db server nodes'  
  command: 'service mysql stop'
```

```
- name: 'Restart all servers (web and db) at once'  
hosts: all_nodes  
tasks:
```

```
- name: 'Restart all servers (web and db) at once'  
  command: '/sbin/shutdown -r'
```

```
- name: 'Start the database services on db server nodes'  
hosts: db_nodes  
tasks:
```

```
- name: 'Start the database services on db server nodes'  
  command: 'service mysql start'
```

```
- name: 'Start the web services on web server nodes'  
hosts: web_nodes  
tasks:
```

```
- name: 'Start the web services on web server nodes'  
  command: 'service httpd start'
```

Ansible Modules

Command module:

It used to execute a command on remote node

Ex: referenceurl: https://kodekloud.com/p/ansible-practice-test/?scenario=questions_ansible_modules

-

name: plat1

hosts: any

tasks:

- name: executing date
command: date

- name: display content
command: cat /etc/sample.php

Script command used to execute the script in remote server

Service Command: start,stop,Restart

-

name: plat1

hosts: any

tasks:

- name: executing date
service: name=postgresql state=started (similar name place httpd,nginx we use to start web services state=started)

Variables:

Its used to store the information.

- hosts: webserver

remote_user: root

become: yes

vars:

pkg: httpd

tasks:

- name: install httpd

yum: name={{pkg}} state=installed

- name: creates index.html file

copy: src=index.html dest=/var/www/html

- name: restart httpd

service: name={{pkg}} state=started

Save

Ansible files:

Previously we store variables inside a playbook but now we are storing variables in a file.

Create fileabc.yml file paste below code on this.

pkg: httpd

After create files.yml (paste below code)

```
---
- hosts: webserver
  remote_user: root
  become: yes
  vars_files:
    - fileabc.yml
  tasks:
    - name: install httpd
      yum: name={{pkg}} state=installed
    - name: creates index.html file
      copy: src=index.html dest=/var/www/html
    - name: restart httpd
      service: name={{pkg}} state=started
```

Checking syntax:

```
ansible-playbook files.yml --syntax
```

```
ansible-playbook files.yml
```


Ansible Handler

It will notify when something is happen

Ex:if any changes is there means we will restart the server otherwise no

vi handler.yml

- hosts: webserver

remote_user: root

become: yes

tasks:

- name: install httpd

 - yum: name=httpd state=installed

- name: copy index.html file

 - copy: src=index.html dest=/var/www/html

 - notify: restart httpd

- name: restart service

 - service: name=httpd state=started

handlers:

- name : restart the server

 - service: name=httpd state=restarted

Note :

Must u have to check the syntax:

ansible-playbook handler.yml --syntax

ansible-playbook handler.yml

Loops playbook ansible

For suppose i want install multi packages in nodes or at a time we install 100 packages at a time.

vi loop.yml

```
- hosts: webserver
  remote_user: root
  become: yes
  tasks:
    - name: installing multiple packages
      yum: name={{item}} state=installed
      with_items:
        - httpd
        - curl
        - wget
```

ansible-playbook loop.yml --syntax

Above if we observe withitems place we need to give whatever the packages we need to install.

Ansible Include:

By using this we are including the yaml file or playbooks

```
vi include.yml
---
- hosts: webserver
  remote_user: root
  become: yes
  tasks:
    - include: pavani.yml
    - include: config.yml
    - include: service.yml
```

Register

Register module we used to capture the output

Debug is used to display the output

```
Vi register.yml
---
- hosts: webserver
  remote_user: root
  become: yes
  tasks:
    - name: install httpd
      yum: name=httpd state=installed
    - name: creates index.html file
      copy: src=index.html dest=/var/www/html
    - name: restart httpd
      service: name=httpd state=started
      register: output
    - debug:
        msg: "{{output}}"
```

```
ansible-playbook register.yml --syntax
```

Tags:

In play book if u want perform a particular task for ex i have 3 task i want to run 2 task that time i just give 2 tagname is enough to run that task.

vi tags.yml

- hosts: webserver

remote_user: root

become: yes

tasks:

- name: install httpd

yum: name=httpd state=installed

tags:

- install

- name: creates index.html file

copy: src=index.html dest=/var/www/html

tags:

- configure

- name: restart httpd

service: name=httpd state=started

tags:

- service

save:

Execute below commands:

Method1:

ansible-playbook tags.yml --tag "install,configure"

ansible-playbook tags.yml --skip-tags "install,configure"

Method2:

ansible-playbook tags.yml --start-at-task="creates index.html file"

Method3:

ansible-playbook tags.yml --step (this step ask permission 2 execute)

Ansible Security or vault

```
---  
- hosts: webserver  
  remote_user: root  
  become: yes  
  vars:  
    password: raj  
  tasks:  
    - name: install httpd  
      yum: name=httpd state=installed  
    - name: creates index.html file  
      copy: src=index.html dest=/var/www/html  
    - name: restart httpd  
      service: name=httpd state=started
```

To run this
ansible-vault encrypt vault.yml

Above command will convert the plain text to other different hidden format

It will ask u r playbook password give it 2 times now that file will be encrypted
Now if u type cat password.yml its not in human readable language.

```
ansible-playbook vault.yml --ask-vault-pass
```

Above command is asked before ur playbook will execute

```
ansible-vault edit vault.yml (command for edit the encrypt file)  
ansible-vault rekey vault.yml (we can modify the password)  
ansible-vault decrypt vault.yml (we can decrypt the ansible file)
```

Ignore errors

Create a vi ignore.yml file

Below code i am giving wrong name 4 index.html

```
- hosts: webserver
  remote_user: root
  become: yes
  tasks:
    - name: install httpd
      yum: name=httpd state=installed
    - name: copy index.html file
      copy: src=indgix.html dest=/var/www/html
      ignore_errors: yes
    - name: restart service
      service: name=httpd state=started
```

Check syntax and run

Simple Task implementation with Any Configuration management tools

- ✓ If we go with any automation tools like ansible, chef, puppet, salt or any other then we can complete this simple task in 3 min (which is required per server) for all 100 server.
- ✓ Ansible will execute a task on all servers parallelly. So it will take only 3 min of time for any number of server for our requirement.
- ✓ Generally writing shell script is complex compare to playbooks and playbooks are very short in code length.

Ansible

Introduction to Ansible

- ✓ Ansible is an open source Automation tool.
- ✓ It is very, very simple to setup and yet powerful.
- ✓ Ansible will be helpful to perform:
 - ✓ Configuration Management
 - ✓ Application Deployment
 - ✓ Task Automation
 - ✓ and also IT orchestration

Install vim and wget pkgs on all servers using Ansible

Inventory

```
172.31.93.110  
172.31.93.120  
172.31.93.132
```

Ansible.cfg

Playbook/Ad-hoc
commands using
Modules

Server-X
172.2.31.81

Ansible Engine/Controller/Master

Ansible uses SSH for operations

Sever-1
172.31.93.110

Sever-2
172.31.93.120

Sever-3
172.31.93.132

Ansible Nodes/Clients

- It is a free open source Automation tool and simple
- Using Existing OpenSSH
- Agent-less – No need to install any agent on Ansible Clients/Nodes
- Python/YAML based
- Highly flexible and configuration management of systems.
- Large number of ready to use modules for system management
- Custom modules can be added if needed



- Create same user(**ansadmin**) across all servers and provide password for all users.
- Provide root privileges to all **ansadmin** users on all servers.
- Make sure that PasswordAuthentication yes in all servers under `/etc/ssh/sshd_config` file.
- Generate ssh-keys using **ssh-keygen** command from ansadmin.
- Copy ssh public key using **ssh-copy-id <hostname>** from `/home/ansadmin/.ssh/` location.
- Now login to remote server without providing password with the following command:
 - **ssh user_name@hostname**
- Now we can test connection from Ansible Engine to Manage Node using:
 - **ansible all -m ping**

Real Time password less authentication ansible configuration.

Take 2 aws rhel instances. (#!/bin/bash)

Need ansible (if want ansible u need to have python installed)

```
yum install wget
```

```
rpm -Uvh https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

```
yum install epel-release -y
```

```
yum install python
```

```
yum install git python3 python3-pip ansible openssl -y
```

```
python --version
```

```
python3 --version
```

```
ansible --version
```

** Now we r going 2 creating a user

```
useradd ansadmin
```

```
[root@ip-172-31-9-112 ~]# passwd ansadmin
```

Changing password for user ansadmin.

New password:

BAD PASSWORD: The password is shorter than 8 characters

Retype new password:

passwd: all authentication tokens updated successfully.

Provide same procedure to all ansible nodes

**

****providing root privileges to ansadmin.**

Go to root path and type:

visudo

Go to lastline under ec2-user add your ansadmin user give privileges

ansadmin ALL=(ALL) NOPASSWD: ALL

**** go to this path and make password authentication yes**

/etc/ssh/sshd_config (in this file under above comment below PasswordAuthentication change no to yes)

To disable tunneled clear text passwords, change to no here!

PasswordAuthentication yes

Save

Restart sshd : systemctl restart sshd

**** Generate ssh key gen (this step is only for master node)**

Before login to ansadmin user

su - ansadmin

Check pwd (u were in /home/ansadmin path)

sudo ls -l (u have to use must sudo here)

sudo ssh-keygen

Enter 3 times enter

Before we copy public ssh key loginto all ur nodes and follow above procedure (install ansible, configure ansible)

Login into node.

Ssh copyid to nodes.
cd .ssh (here we can see public ip and private ip)
Go to master node. (paste below command)
ssh-copy-id 172.31.7.66 (nodes private ip address)
Click on yes.
Give your node user password.
Now u r successfully logged in node server.

If you want to check
ssh give urnode publicip or privateip it will directly logged into your server.
Or
ssh ansadmin@ipaddress
Now passwordless authentication done.
Now copy all ur nodes ip address.
**

Now update inventory files with node ips but default inventory file path is
vi /etc/ansible/hosts
Open file top of that paste all node ipaddress.
Save

To check all nodes connectivity.
ansible all -m ping (**important note when run this command you must be in ansadmin user and run coomand)
It will ping all comands (nodes must be python with ansible).

Ansible Directory Structure

Default ansible path:
/etc/ansible.

Here we have 3 files 1-> ansible.cfg 2.roles 3.hosts

In hosts

Paste all node private ips if master also same in aws or master in vmware or etc just use publicip of nodes

Host file is called as inventory file.

**if want change default path of the inventory just go to ansible.cfg under inventory give exact path

If multiple person are used to work same host file that time conflict will come thats why.

Create our own directory inside of that

```
mkdir myinventory
```

```
cd myinventory
```

```
sudo cp -rpP /etc/ansible/* .
```

 (it will copy all destination files into our folder)

Here create our host files also.

now ansible.cfg file in inventory line give inventory = ./inventory (new host file name).

Now check ansible all -m ping

Never Don't modify name of ansible.cfg entire ur ansible carrer)

**if we want to run our custom ansible file without configuring ansible.cfg follow below command

```
ansible all -m ping -i inventory
```

 (now it will run directly)

Disable hostkey checking

What is it “ if any time u want to connect to nodes it will ask yes or no in your remote server) it's asking for confirmation for that we need to disable hostkey checking.

In real time

```
mkdir non-production
```

```
mkdir production
```

```
cd non-production
```

```
sudo cp -r /etc/ansible/* .
```

```
sudo chown -R ansadmin *
```

Open ansible.cfg inventory give .(current folder)

****2 ways to disable hostkey check**

1.export ANSIBLE_HOST_KEY_CHECKING=False

2.ansible.cfg file

host_key_checking = False (make this one true)

Inventory file with groups

If u want to ping all hosts at a time:

```
ansible all -m ping
```

If you want ping only oneip that time

```
ansible iipaddress -m ping
```

If u want ping two ipaddress.

```
ansible ip1address:ip2address -m ping
```

For suppose u have 100 servers but u want to work on 50 servers only i.e we can use group concept

```
cd non-production
```

In inventory file or hosts

[webservers] (group name must be written without spaces)

localhost

lp1

lp2

[dbservers]

localhost

lp3

lp4

To ping or run groups

```
ansible webservers -m ping (now it will run only group1 ip's)
```

```
ansible webservers:dbservers -m ping (now it will run both groups ip's)
```

Host file group of groups

cd non-production

In inventory file or hosts

[webservers] (group name must be written without spaces)

localhost

lp1

lp2

[dbservers]

localhost

lp3

lp4

[merginggroup]

Webservers

dbservers

To ping

ansible merginggroup -m ping

ansible.cfg (when u run default first it will call that to priority basis we can see this one below picture)

Ansible

Ansible ansible.cfg

- The default location is: `/etc/ansible/ansible.cfg`, in which we can make various settings like
 - Location of inventory file
 - `host_key_checking` as False
- But we can define ansible configuration file in different location and for this there is a priority for this files.
- Locations with priority(starting from top to bottom):
 - `ANSIBLE_CONFIG` environment variable
 - `./ansible.cfg` from the current directory
 - `~/.ansible.cfg` file present in home directory
 - `/etc/ansible/ansible.cfg` default ansible.cfg file.
- Ansible will only use the configuration settings from the file which is found in this sequence first, it will not look for the settings in the higher sequence files if the setting is not present in the file which is chosen for deployment.

Working With Nodes

Two ways :

- 1.Ad-hoc commands
- 2.Playbooks

1.Ad-hoc commands

Its useful to execute single task on ur manage nodes

ansible all -m shell -a "uptime" (uptime is a adhoc command)

ansible all -m shell -a "uptime (free-h is a adhoc command)

Push base mechanism practical

ansible all -m shell -a "sleep 5 : echo hai " (each node print echo after 5 seconds)

ansible-doc -l (will giive the list of all adhoc commands)

Ansible when running it will push a folder to nodes

If u want to check login to node and run above any commands on master .

In node go to user ansadmin here we can see one .ansible

/home/ansadmin/.ansible/tmp/here the folder will come in the format of folder.py

**Default ansible will run 5 servers at a time if u want change . change in ansible.cfg fort=how many u want

Default ansible run parallely (at a time it will execute all nodes)

If u want run sequential go to ansible.cfg and change or made fork = 1 (run comand it will execute one by one)

Adhoc transfer a file from one server to other

```
ansible db -m copy -a "src=./hosts dest=/tmp/inventory"
```

ssh nodeip (check in tmp folder it will create)

** ansible is an idempotency means any changes made before transfer then only it will transfer it exists previous one it won't transfer

```
ansible db -m copy -a "content='hello transfer a file' dest=/tmp/hello.txt"
```

```
ansible db -m copy -a "content='db conf' dest=/home/ansadmin/db.conf"
```

**Download a file from nodes using fetch module ansible adhoc commands

```
ansible db -m fetch -a "src=/home/ansadmin/demo.txt dest=./demo"
```

Changed true means your task executes successfully.

```
ansible db -m fetch -a "src=/home/ansadmin/demo.txt dest=./newdemo/{{inventory_hostname}}_demo.txt flat=yes"
```

Above command will download all servers demo.txt files with the name format of their IP names

Ex: 56.886_demo.txt

78.452.54_demo.txt

Create or delete a file or directory using adhoc

Create a file:

Here db is an group name insted we can use all also
ansible db -m file -a "path=/tmp/hello.txt state=touch"

ansible db -m file -a "path=/etc/hello.txt state=touch"

It will throgh error bcoz etc belongs to root user if u want to execute u have to add -b or -i

ansible db -m file -a "path=/etc/hello.txt state=touch" -b

Different modules to work with files ansible ad-hoc command

Reference url : https://docs.ansible.com/ansible/latest/user_guide/intro_adhoc.html

Install packages like git,mysql,httpd,nginx using adhoc commands

su - ansadmin

cd non-production

ansible -m ping all

ansible db -m yum -a "name=git state=latest" -b (here -b used for become root user without this we will get eror)

ansible db -m yum -a "name=https state=latest" -b

Check once about command module.

ansible all -m command -a "uptime"
ansible all -m command -a "date"
ansible all -m command -a "who"

Stat (file exist or not)

ansible all -m stat -a "path=/etc/hosts"

Yum install a backage

ansible all -m yum -a "name=git state=latest" -b

•

user (for creating a user) to check users vi /etc/passwd

ansible all -m user -a "name=john" -b

•

Setup is a module which will give the entire information of ansible.

ansible all -m setup

In ansible.cfg

We can set default modules like command & yum .

In privilege escalation

become true means we don't need mention -b

Ansible Facts & Variables

Ansible facts nothing about information about ur nodes like os,distribution,release,python etc..

The task of collected this remote system information called as gathering facts and collected facts or variable

```
cd non-production
ansible db -m setup
```

Create and work with ansible custom facts .

Default facts
Custom facts

```
cd non-production
ansible all -m shell -a "git --version"
ansible all -m shell -a "/usr/sbin/httpd --version"
```

This topic not that mush important

Ansible Inventory static & dynamic inventory

Collection of hosts known as ansible inventory

Dynamic inventory are scripted like shell/python for dynamic environment
Cloud is dynamic environment.

By default ansible supports some cloud environments.

Refurl: <https://github.com/ansible/ansible/tree/stable-2.9/contrib/inventory>

In above file we see that ec2.ini and ec2.py

If we want work on dynamic inventory on aws we need to download ec2.ini & ec2.py

cd non-prod

vi ec2.ini (paste the ec2.ini code)

vi ec2.py (from github copy the code and paste in this file)

chmod 755 ec2.py

Now we have 2 files ec2.py is a dynamic file ,

And ec2.ini is an configuration file.

To run script ./ec2.py

We will get error bcoz we have python3 not python our ec2script run python thats why

** sudo ln -s /usr/bin/python3 /usr/bin/python

This command will move our /usr/bin python 3 to /usr/bin/python now we will run python --version it works.

Now type ./ec2.py
Now it will ask No module named 'boto'

For this we instal boto
For install boto we install python3 thatswhy we run pip3
sudo pip3 install boto

Installing collected packages: boto
Successfully installed boto-2.49.0

Create a role aws and assign to ec2 ansible machine.
Attach or detach a role create a new role
Create a role → ec2 → next → ec2fullpermission → next review → role name (ec2 full access) → create role

Refresh → Attach or detach a role → choose ec2fullaccess → click on apply.

NOW RUN ./ec2.py (u will get below error to solve this)
ERROR: "Forbidden", while: getting ElastiCache clusters[ansadmin@ip-172-31-9-112 non-prod]\$ ^C
sudo vi ec2.ini
/ElastiCache (search for this uncomment it)
./ec2.py

After running under
"ec2": [
 "13.126.158.22"
],

We saw our ec2 instances.

Here by default group is ec2
If u want to run dynamically inventory
`ansible -i ec2.py ec2 -m ping`

After setup again new requirement came u have to do means simply u took previous server ami image and simply we can run on that server it will run.

Real time above ami procedure we can follow.

** if managing nodes don't have python we can communicate through raw module.

Working with managing nodes with password

There 2 ways to connect

1. Using sshkeys
2. Using ssh password

Launch one new instances and create a ansible user and provide root privileges.

```
vi /etc/sshd_conf
```

```
PasswordAuthentication : yes
```

```
service sshd restart
```

Now from ur ansible engine trt connect new ssh node

```
ssh newnodeipaddress
```

It will ask password enter now u get access of node

If visudo

```
ansible all
```

Same like that if u want ping node

```
ansible all -m ping -k
```

It will ask password

Here -k is for asking password

If u want to install git or httpd

```
ansible ipaddress -m yum -a "name=git state=latest" -k -b
```

-k is for asking password to login

-b is for become root privileges

At a time 4 ansible engine doing something

```
ansible ipaddress -m yum -a "name=git
```

```
state=latest" -k -b -K
```

-K capital is for sudo password

-k small k is for ssh password

Execute ansible tasks with default and different users

When u login ansible and u will execute any commands it will run on default user ansible

If u want run on different user

Go to ur nodes and create user and make it password

When u r executing tasks from ansible engine at that time u just add -u username

Ex:

```
ansible nodeipaddress -m file -a "path=sample.txt state=touch" -k -u xyz
```

To login from server to node with username

```
ssh username@password
```

-u remote user to connect

But when u login ansible and running on tasks that time default user is ansible

Ansible Variables

Variables used to store the values

Ansible variables names should be letters,numbers,underscores and they should always start with letter.

Types:

Default variables,

Inventory variables.

Facts and local facts

Registered vars etc.

Default variables are :

inventory_hostname

Inventory_hostname_short

groups/groups.keys()

```
ansible all -m debug -a "msg='this is debug module' "
```

```
ansible all -m debug -a "var=inventory_hostname"
```

```
ansible all -m debug -a "msg={{inventory_hostname}}"
```

```
ansible all -m debug -a "var=groups"
```

Debug works only on server master not nodes

Ansible Play Books.

Two ways to execute tasks in mange nodes

- 1.ad hoc commands
- 2.Play books

If u want to run multiple tasks at all servers at that time we can user playbooks

Ex : i want to install httpd,nginx , mysql at a time

But in ad hoc commands

ansible all -m yum -a "name=httpd state=latest" like that we need to write single commands but playbook at a time we can configure all servers

Playbook:

It build on yaml language

Structure of playbook

Task

Play

Playbook

Sample ansible playbook:

- - (represents its a playbook file)
- hosts: all
- become: yes
- tasks:
- yum: name=httpd state=latest

ansible-playbook sample.yml (to run ansible playbok)

Simple playbooks:

cd non-prod

vi hosts (we have 2 groups assume)

[web]

1452.24.23434

45,545.545

[db]

44.44.4454

856.252.45

Now i want shift a file only to the web group servers only

ansible web -m copy -a "src=sample.txt dest=tmp/ "

In playbook

- - -(represents its a playbook file)
- hosts: web
- gather_facts: false
- become: yes
- tasks:
 - copy: src=web.xml dest=web.xml
 - file: path:demonew.txt state=touch

For naming format u have to give means follow this procedure

- - -(represents its a playbook file)
- name: this is a sample test of web servers
- hosts: web
- gather_facts: false
- become: yes
- tasks:
 - name: this is first task
 - copy: src=web.xml dest=web.xml
 - name: this is first task
 - file: path:demonew.txt state=touch

3

```
-- (represents its a playbook file)
- name: this is a sample test of web servers
  hosts: web
  gather_facts: false
  become: yes (its for root privileges)
  tasks:
    - name: this is first task
      yum: name=httpd state=present (absent for uninstalling any service)
    - name: this is first task
      yum: name=wget state=present
```

Basic Key points to run ansible

To run ansible playbook in shortcut

cd non-prod

which ansible-playbook

/usr/bin/ansible-playbook

Copy above path go to ur yaml or yml file at top

-- (remove this hyphen and paste in this place)

#!/usr/bin/ansible-playbook (save it)

chmod +x ymlfile.yml

./ymlfile.yml (thats it it will execute)

Above is the way for without running ansible-playbook filename

Now if u want check ur syntax error of yml file

`./filename.yml --syntax-check`

If it's no errors it will display playbook name otherwise it will display errors

Dry Run.

It won't affect any task on your nodes it will just tell you what will happen if u run this ansible playbook

Command:

`./filename.yml --check`

--check (its a dry run it will what will happen if u run)

When u run

`./file.yml -v` (it will give extra information while running on server)

-v (verbose mode)

-vvvvv(how its going to connect it will give detail information)

Basic concepts about playbook.

print any message using playbook

We r using debug module

This module prints statements during the executions and can be useful for debugging variables or expressions

It accept 3 parameters

msg ,var , verbosity

```
---  
- hosts: web  
  tasks:  
    - name: debug module  
      debug:  
        msg:  
          - "under debug array message1 "  
          - "under debug array message2"  
          - "variable printing: {{inventory_hostname}}"
```

If want print message along with variable follow above procedure msg

Below debug remove total msg array and write

```
var: inventory_name
```

Ansible variables

A variables defined by us called custom variable

```
- hosts: web
vars:
  x: 23
gather_facts: false
tasks:
  - name: debug module
    debug: var=x
```

if u want to display multiple variables

```
- hosts: web
vars:
  x: 23
  b: 24
gather_facts: false
tasks:
  - name: debug module
    debug:
      msg:
        - "first variable is {{x| type_debug}}"
        - "second variable is {{b}}"
```

Ansible Data Structures

Its used to stores multiple data or roles

Collection of multiple values into single variable

- - -

- hosts: web

 - vars:

 - x: 23

 - pkg: ['httpd', 'vim', 'nano']

 - pkg2: {'linux': 'httpd', 'ubuntu': 'apache2'}

 - gather_facts: false

 - tasks:

 - name: debug module

 - debug: var=pkg

Usage of register find bash version of all nodes.

When ever playbook run it will generate output that output we store in ansible register variable

We can use that variable in different scenario like condition statements and logging

vi bashversion.yml

```
- hosts: web
  gather_facts: false
  tasks:
    - shell: "bash --version"
      register: bashvariable
    - debug var=bashvariable.stdout.split("\n")[0].split()[3]
```

(stdout is to print output of register variable)

Split (to split the output)

How to Read and print a variable value.

It used to run an playbook while that time u can pass some values and you can read that values in

Console

The command is

vars_prompt

- hosts: web

vars:

x: 23

b: 24

vars_prompt:

- name: username

prompt: enter username

private: false (display data while entering on console)

- name: password

prompt: enter password

private: false

gather_facts: false

tasks:

- debug:

msg: "username is {{username}} and password {{password}}"

Reading variables from file

vi sample1.yml

x: 23

b:

- 4

- 42

- 43

pkg: {"linux":"httpd" , "ubuntu" : "tomcat"}

vi mainfile

- hosts: web

vars_files: sample1.yml

tasks:

debug: var=pkg

here pkg is the variable of sample1 file

Command line arguments

- hosts: web
- gather_facts: false
- tasks:
 - debug: var=x

ansible-playbook myfile.yml -e x=24

When we run on ansible that time we can pass that values

- hosts: web
- gather_facts: false
- tasks:
 - debug:
 - msg:
 - {{x}}
 - {{y}}

ansible-playbook myfile.yml -e "x=24 y=89"

```
- hosts: web
  gather_facts: false
  become yes
  tasks:
  - name: debug module
    yum:
      name: "{{pkg}}"
      state: "{{state}}"
```

`ansible-playbook myfile.yml -e "pkg=httpd state=latest"`

Usage of gather facts:

Collecting information of nodes using gather facts

Working with inventory hostname with hostvars

- hosts: web
- gather_facts: false
- tasks:
 - debug: var=inventory_hostname
 - debug: var=hostvars[inventory_hostname]

in hostvars

by default we fetch information all inventory along with this host name

For playbooks if u want gui u can download microsoft visual studio code-----

Ansible Arithmetic Operators

```
- hosts: web
vars:
  x: 23
  y: 24
gather_facts: false
tasks:
- name: debug module
  debug:
    msg:
      - "The value of x is {{x}}"
      - "The value of y is {{y}}"
      - "{{x}} + {{y}} = {{x+y}}"
      - "{{x}} - {{y}} = {{x-y}}"
      - "{{x}} * {{y}} = {{x*y}}"
      - "{{x}} / {{y}} = {{x/y}}"
```

Run ur playbook.

Ansible Filter Methods

```
- hosts: web
vars:
  x: "this is sample text"
  y: "12"
  z: {4,5,6}
gather_facts: false
tasks:
- name: debug module
  debug:
    msg:
      - "{{x|lower}}"
      - "{{x|upper}}"
      - "{{x|title}}"
      - "{{x.lower()}}"
      - "{{x.upper()}}"
      - "{{y|int}}"
      - "{{z|min}}"
      - "{{x|max}}"
      - "{{x.split()}}"
```

To know more search on [ansible filter documentation](#)
Run ur playbook.

Membership Operator

Applications Places Atom Sat 22:42

test_oper.yml — ~/my_ansible_nprod — Atom

File Edit View Selection Find Packages Help

Project

- my_ansible_nprod
 - ansible.cfg
 - arithmetic_oper.yml
 - bkp_inventory
 - cmd_line_args.yml
 - comparision_operators.yml
 - data_structures.yml
 - demo_.yml
 - demo.yml
 - empty.yml
 - example.yml
 - filter_and_methods.yml
 - first_playbook_vsc.yml
 - gather_facts.yml
 - hosts
 - hosts_bkp
 - hostvars_inventory_hostname.yml
 - imp.sh
 - input_outputs.yml
 - install_uninstall_httpd.yml
 - install_yum_pkgs.yml
 - inventory
 - member_ship_op.yml
 - mv_start_run

```
2 - name: This is about test operators
3   hosts: localhost
4   gather_facts: false
5   vars:
6     x: 40
7     my_name: 'ansible'
8     my_path: '/home/ansadmin/my_ansible_nprod/member_ship_op.yml'
9     my_link_path: '/home/ansadmin/my_ansible_nprod/operators.yml'
10  tasks:
11    - debug:
12      msg:
13        - "x is defined:  {{ x is defined }}"
14        - "y is defined:  {{ y is defined }}"
15        - "z is undefined: {{ z is undefined }}"
16        - "my_name is lower: {{my_name is lower}}"
17        - "my_name is upper: {{my_name is upper}}"
18        - "my_name is string: {{my_name is string}}"
19        - "x is divisibleby 2: {{x is divisibleby 2}}"
20        - "x is even: {{ x is even }}"
21        - "x is odd: {{ x is odd }}"
22        - "x is number: {{ x is number }}"
23        - "The given path is: {{my_path }}"
24        - "my_path is file:  {{my_path is file }}"
25        - "my_path is directory: {{my_path is directory }}"
26        - "my_path is exists:  {{my_path is exists }}"
27        - "my_link_path is link  {{my_link_path is link }}"
28
```

my_link_path

test_oper.yml* 27:60 LF UTF-8 YAML GitHub Git (0)

ansadmin@localhost:~/my_ansible_n... test_oper.yml — ~/my_ansible_npr...

Conditional Operator

- simple play to install httpd
 - hosts: web
 - gather_facts: false
 - become: yes
 - tasks:
 - name: installing httpd using yum
 - yum:
 - name: httpd
 - state: absent
 - when: ansible_distribution != "ubuntu"
 - name: installing httpd using apt
 - yum:
 - name: apache2
 - state: latest
 - when: ansible_distribution == "ubuntu"

Run ur playbook.

Handlers In Ansible

```
- - -
- hosts: web
  gather_facts: false
  become: yes
  tasks:
    - name: installing httpd using yum
      yum:
        name: httpd
        state: absent
      register: out (registering output of the first yum task after this task done)
    - name: installing httpd using apt
      services:
        name: httpd
        state: started
      when: out.changed == True
```

By using handlers

```
- - -
- hosts: web
  gather_facts: false
  become: yes
  tasks:
    - name: installing httpd using yum
      yum:
        name: httpd
        state: present
      notify:
        - start httpd
  handlers: (it means that when first task is changed then only second task will execute)
    - name: installing httpd using apt
      services:
        name: httpd
        state: started
      when: out.changed == True
```

Loops.

- - -

```
- hosts: web
gather_facts: false
become: yes
tasks:
- name: installing httpd using yum
  yum:
    name: httpd
    state: absent
  loop:
    - git
    - tomcat
    - nginx
    - httpd
```

Run this ansible playbook

Tags Conditional Operator

- simple play to install httpd
 - hosts: web
 - gather_facts: false
 - become: yes
 - tasks:
 - debug:
 - msg: "this is first task"
 - tags:
 - first
 - common(default first task will execute)
 - debug:
 - msg: "this is 2 task"
 - tags: sec
 - debug:
 - msg: "this is 3 task"
 - tags: third
 - debug:
 - msg: "this is 4 task"
 - tags: fourth

ansible-playbook tags.yml -t first,sec (now it will execute only first and second tasks).

Real Time java apache tomcat playbook

```
---  
-  
  become: true  
  gather_facts: false  
  hosts: server1  
  name: "tomcat installation"  
  vars:  
    req_java: java-1.7.0-openjdk  
    set_java: java-1.8.0-openjdk  
  tasks:  
    - name: "nginx update"  
      yum:  
        name: "{{req_java}}"  
        state: latest  
    - name: "updating java"  
      alternatives:  
        name: java  
        link: /usr/bin/java  
        path: /usr/lib/jvm/set_java/bin/java
```

demo.yml

-

become: true

gather_facts: false

hosts: server1

name: "tomcat installation"

tasks:

- name: "wget apache tomcat install"

get_url:

url: <https://downloads.apache.org/tomcat/tomcat-8/v8.5.57/bin/apache-tomcat-8.5.57.tar.gz>

dest: /usr/local

- name: extracting tomcat url

unarchive:

src: "/usr/local/apache-tomcat-8.5.57.tar.gz"

dest: /usr/local

remote_src: yes

- name: renaming tomcat

command: mv /usr/local/apache-tomcat-8.5.57.tar.gz /usr/local/latest

- name: running or starting tomcat

shell: nohup /usr/local/latest/bin/startup.sh

Playbooks reference git:

https://github.com/yankils/ansible_for_beginners

yamlnit website name to practise yml playbooks.

Ansible playbook

- A playbook is a text file written in YAML (YAML Ain't Markup Language) format, and is normally saved as **.yaml**.
- The playbook begins with a line consisting of three dashes (**---**) as a start of document marker.
- An item in a YAML list starts with a **single dash** followed by a **space**.
- **hosts** and **tasks** are **mandatory** items in a playbook
- The playbook primarily uses **indentation** with **space characters** to indicate the structure of its data
- **Modules** are used to perform **tasks** **ansible all -m user -a "name=john" -b**
- **Comment** start with **#**

Create and delete nodes using ansible

```
---  
- name: installing packages  
  hosts: all  
  become: yes  
  tasks:  
    - name: installing git packages  
      file:  
        path: /home/ansadmin/demofile  
        state: touch  
    - name: creating a directory  
      file:  
        path: /home/ansadmin/directory  
        state: directory
```

Copy module

```
---  
- name: installing packages  
  hosts: all  
  become: yes  
  tasks:  
    - name: installing git packages  
      copy:  
        src: /home/ansadmin/host  
        dest: /home/ansadmin/  
        mode: 0600
```

Httpd install on ansible playbook.

- name: installing packages
 - hosts: all
 - become: yes
 - tasks:
 - name: installing http packages
 - yum:
 - name: httpd
 - state: latest
 - notify: starting httpd package
 - handlers:
 - name: starting httpd package
 - service:
 - name: httpd
 - state: started

Note: notify message and

Handler:

Name: name of handler

****notify name and handler name should be same otherwise it will through an error.**

Gathering facts:

It will going to get the nodes system information and then it will execute the ansible playbook tasks.

```
ansible all -m setup
```

Above command also will give the information of all nodes

If u want disable gathering facts:

```
become: true
gather_facts: false
hosts: server1
name: "tomcat installation"
```

** to search packes installed or not in linux:

```
rpm -qa | grep wget
```

```
rpm -qa | grep httpd
```

Etc

When condition:

- name: installing packages
hosts: all
become: yes
tasks:
 - name: installing http packages
yum:
 - name: httpd
 - state: latest
 - when: ansible_os_family == "RedHat"
 - ignore_errors: yes
- name: starting httpd package
service:
 - name: httpd
 - state: started
 - when: ansible_os_family == "RedHat"
- name: installing apt packages
yum:
 - name: apache2
 - state: latest
 - when: ansible_os_family == "Debian"
- name: starting httpd package
service:
 - name: httpd
 - state: started
 - when: ansible_os_family == "Debian"

Uninstalling httpd:

- name: installing packages
 - hosts: all
 - become: yes
 - tasks:
 - name: starting httpd package
 - service:
 - name: httpd
 - state: stopped
 - when: ansible_os_family == "RedHat"
 - name: installing http packages
 - yum:
 - name: httpd
 - state: absent
 - when: ansible_os_family == "RedHat"

Adding a copy task to ansible playbook

```
--  
- name: installing packages  
  hosts: all  
  become: yes  
  tasks:  
    - name: installing http packages  
      yum:  
        name: httpd  
        state: latest  
        when: ansible_os_family == "RedHat"  
    - name: starting httpd package  
      service:  
        name: httpd  
        state: started  
        when: ansible_os_family == "RedHat"  
    - name: installing apt packages  
      yum:  
        name: apache2  
        state: latest  
        when: ansible_os_family == "Debian"  
    - name: starting httpd package  
      service:  
        name: httpd  
        state: started  
        when: ansible_os_family == "Debian"  
    - name: coping a file from server to node  
      copy:  
        src: /home/ansadmin/index.html  
        dest: /var/www/html/  
        mode: 0666
```

For loop with in items

```
- name: installing packages
hosts: all
become: yes
tasks:
  - name: installing http packages
    yum:
      name: ['git' , 'httpd']
      state: installed
```

Convert shell commands Ansible playbook

```
---
- name: setup tomcat
  hosts: all
  become: true
  tasks:
    - name: install java
      yum:
        name: java
        state: installed
      when: ansible_os_family == "RedHat"

    - name: install java on ubuntu
      apt:
        name: default-jdk
        state: present
      when: ansible_os_family == "Debian"

    - name: download tomcat packages
      get_url:
        url: http://mirrors.estointernet.in/apache/tomcat/tomcat-8/v8.5.50/bin/apache-tomcat-8.5.50.tar.gz
        dest: /opt

    - name: untar apache packages
      unarchive:
        src: /opt/apache-tomcat-8.5.50.tar.gz
        dest: /opt
        remote_src: yes

    - name: add execution permissions on startup.sh file
      file:
        path: /opt/apache-tomcat-8.5.50/bin/startup.sh
        mode: 0777

    - name: start tomcat services
      shell: nohup ./startup.sh
      args:
        chdir: /opt/apache-tomcat-8.5.50/bin
```

Tags: (when u want execute particular task only)

- name: this playbook install httpd
 - hosts: all
 - become: true
 - tasks:
 - name: install package
 - yum:
 - name: httpd
 - state: installed
 - when: ansible_os_family == "RedHat"
 - tags: install_apache
 - name: start apache
 - service:
 - name: httpd
 - state: started
 - when: ansible_os_family == "RedHat"
 - tags: start_apache

ansible-playbook.yml file.yml --tags "install_apache"

It's a feature of ansible that allows to keep sensitive data such as passwords.

Ansible Vault

Ansible Vault is a feature of ansible that allows you to keep sensitive data such as passwords or keys in encrypted files, rather than as plaintext in playbooks or roles.

- **create** : to create ansible vault file in the encrypted format
- **view**: to view data of encrypted file
- **edit**: to edit encrypted file
- **encrypt** : to encrypt an unencrypted file
- **decrypt**: to decrypt an encrypted file

- **--ask-vault-pass** : to provide password while running playbook
- **--vault-password-file** : to pass a vault password through a file.

To create a encrypted file:
ansible-vault create valutpass.yml
It will ask password giveit and write ur file
This is encrypted file

To view the valult file
ansible-vault view valutpass.yml
It will ask password enter and now u can see the data

To decrypt file
ansible-vault decrypt valutpass.yml

ansible-vault encrypt valutpass.yml

Ansible vault with git

in valut-pass.yml
password: rajesh123 (save it here ur username of git)

vi ansible-valut.yml

```
- name: ansible palybook to test ansible vault
hosts: all
become: true
vars_files:
  - vault-pass.yml
tasks:
- name: clone a repo
  git:
    repo: https://yankils:{{ password }}@github.com/yankils/vault.git
    dest: /opt/ansadmin/test-vault
```

**here password is a variable of vault-pass.yml file

Ansible Roles:

When our playbooks data are going to be bigger its difficult to manage all the tasks i.e we are using ansible playbook.

** Roles are re usability

Ansible galaxy contain the roles

Dynamic

Static

Refurl: https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html

Creating a role:

`ansible-galaxy init setup-apache`

Ansible galaxy will initialize a role and it will give the directory structure.

```
|— defaults
|   |— main.yml
|— files
|— handlers
|   |— main.yml
|— meta
|   |— main.yml
|— README.md
|— tasks
|   |— main.yml
|— templates
|— tests
|   |— inventory
|   |— test.yml
|— vars
|   |— main.yml
```

```
-- (its a part1)
- name: this playbook install httpd
  hosts: all
  become: true
  vars:
    port: 8082
  tasks:
    - name: install package
      yum:
        name: httpd
        state: installed
      when: ansible_os_family == "RedHat"
      notify: start apache

    - name: install apache2
      apt:
        name: apache2
        state: present
      when: ansible_os_family == "Debian"
      notify: start apache2

    - name: copy index.html
      copy:
        src: /opt/ansible/index.html
        dest: /var/www/html
        mode: 0666
```

- name: Ensure the default Apache port is {{ port }}

lineinfile:

path: /etc/httpd/conf/httpd.conf

regexp: '^Listen '

insertafter: '^#Listen '

line: Listen {{ port }}

when: ansible_os_family == "RedHat"

notify: restart apache

- name: Ensure the default Apache port is {{ port }} on ubuntu

lineinfile:

path: /etc/apache2/ports.conf

regexp: '^Listen '

insertafter: "# /etc/apache2/sites-enabled/000-default.conf"

line: Listen {{ port }}

when: ansible_os_family == "Debian"

notify: restart apache2

handlers:

- name: start apache

service:

name: httpd

state: started (part 2 completed)

All parts in single file: ref url: https://github.com/yankils/ansible_for_beginners/blob/master/setup-apache_backup.yml

```
---
- name: this playbook install httpd
  hosts: all
  become: true
  vars:
    port: 8082
  tasks:
- name: install package
  yum:
    name: httpd
    state: installed
  when: ansible_os_family == "RedHat"
  notify: start apache

- name: install apache2
  apt:
    name: apache2
    state: present
  when: ansible_os_family == "Debian"
  notify: start apache2

- name: copy index.html
  copy:
    src: /opt/ansible/index.html
    dest: /var/www/html
    mode: 0666

- name: Ensure the default Apache port is {{ port }}
  lineinfile:
    path: /etc/httpd/conf/httpd.conf
    regexp: '^Listen '
    insertafter: '^#Listen '
    line: Listen {{ port }}
  when: ansible_os_family == "RedHat"
  notify: restart apache
```

```
- name: Ensure the default Apache port is {{ port }} on ubuntu
  lineinfile:
    path: /etc/apache2/ports.conf
    regexp: '^Listen '
    insertafter: "# /etc/apache2/sites-enabled/000-default.conf"
    line: Listen {{ port }}
  when: ansible_os_family == "Debian"
  notify: restart apache2
```

```
handlers:
- name: start apache
  service:
    name: httpd
    state: started
```

```
- name: start apache2
  service:
    name: apache2
    state: started
```

```
- name: restart apache
  service:
    name: httpd
    state: restarted
```

```
- name: restart apache2
  service:
    name: apache2
    state: restarted
```

Now we are going to convert ansible playbook into role: (reusability):

- name: this playbook install httpd
- hosts: all
- become: true
- roles:
 - setup-apache

```
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml
```

In our previous playbook we configure in role

->var should be place in vi vara/main.yml

->tasks means below all task all data copy & place from starting without spaces

->files: cp /opt/ansible/index.html files/

->handlers: vi handlers/main.yml (copy below handler code paste it)

-> if u miss anything it will take from default:

vi defaults/main.yml

Port: 8080

** note in a role task is mandory remaining all are optional

To check port no:

cat /etc/httpd/conf/httpd.conf | grep Listen

Thank You & All The Best From

Rajesh Singamsetti





