Adhoc commands

====================

Syntax of Adhoc commands

--------------------------------------

ansible all/ipaddress/group\_name -i path\_of\_inventory -m module\_name -a ' ' .

Command Module for adhoc command

=============================

Ansible command to see the memory information of all managed nodes

ansible all -i /etc/ansible/hosts -m command -a 'free m'

apt modules

=======================

Ansible command to install git on all managed nodes

ansible all -m apt -a 'name=git state=present' -b

state=present is for installation

state=absent is for uninstallation

state=latest is for upgrading to a later version

Important Modules in Ansible

=================================================

1. command: This is used to execute linux commands on the managed nodes

This is the default module of Ansible

2. shell: This is used to execute shell scripts or python scripts on

the managed nodes. It is also used for running commands related

to redirection and piping

3. ping: Used to check if the remote servers are pingable or not

4. user: This is used to perform user administration on the remote

servers like creating/deleting users, setting passwords, setting

home directories etc

5. copy: This is used to copy files and folders from the controller to

the managed nodes

6. fetch: This is used to copy files from the managed nodes to the controller

7. file: Used to create/delete files or directories on the managed nodes

8. apt: Used for s/w package management on the managed nodes like installing

s/w's, deleting, upgrading etc. This works on ubuntu based machines

9. yum: Similar to apt but it works on Centos, Redhat linux etc

10. service: Used to start stop and restart services on the managed nodes

11. uri: Used to check if a url is reachable from the managed nodes

12. git: Used to perform git version controlling on the remote managed nodes

13. get\_url: Used to download files from remote servers works like

linux command wget

14. stat: Captures info about files and folders present on the managed nodes

15. debug: This is the print statement of ansible

16. include: Used to call child playbooks from the level of a parent playbook

17. replace: Used to change specific sections of the file

18. pause: Used to pause the playbook execution for a specific period

19. docker\_container: Used to handle docker containers on the managed nodes

20. docker\_image: Used to handle docker images on the managed nodes

Ansible perform remote configuration on servers in 3 ways

1 Adhoc commands

2 Playbooks

3 Roles

Formation of the yaml file like be below

---

Info-it solutions

administration:

hr: dhana

advisor: krishna

Devops enginers:

ceo: harsha

admin: charan

services: sumanth

Devops trainers:

name: mukesh

name: chandra

…

**Ansible playbooks**

Playbooks are created using yaml and each playbook is a combination of multiple plays. A play contains info about what module has to be be executed.These plays are designed to work on a single host or a group of hosts

(or) all the hosts.

Install of the java by using playbook

vi installjava.yml

---

- hosts: all

become: true

tasks:

- name: Update APT package manager repositories cache

become: true

apt:

update\_cache: yes

- name: Install Java using Ansible

become: yes

apt:

name: openjdk-11-jre-headless

state: present

...

To check if the playbook is syntactically correct or not

ansible-playbook installjava.yml --syntax-check

To run the playbook

Ansible-playbook installjava.yml -b

Create a sample user on all managed nodes

vi create-user.yml

---

- name: Create user

hosts: all

tasks:

- name: User creation

user:

name: Hari

password: 123456

uid: 1590

home: /home/ubuntu/Hari

comment: "A normal user"

shell: /bin/bash

...

To run the playbook

Ansible-playbook create-user.yml -b

Creation of the jenkins installtion

vi jenkins-install.yml

---

- hosts: all

tasks:

- name: ensure the jenkins apt repository key is installed

apt\_key: url=https://pkg.jenkins.io/debian-stable/jenkins.io.key state=present

become: yes

- name: ensure the repository is configured

apt\_repository: repo='deb https://pkg.jenkins.io/debian-stable binary/' state=present

become: yes

- name: ensure jenkins is installed

apt: name=jenkins update\_cache=yes

become: yes

- name: ensure jenkins is running

service: name=jenkins state=started

…

To run playbook

Ansible-playbook jenkins-install.yml -b

Take public\_ip of the node:8080

Variables in Ansible

=========================

Variables are categorised into 3 type

1 Global scope varaibles

2 Host Scope variables

3 Play scope variables

Global scope variables

===========================

These variables are defined from the command prompt using "--extra-vars"

and they have the highest level of priority

Ansible playbook to install or uninstall various s/w applications

vim playbook1.yml

---

- name: Install s/w applications

hosts: all

tasks:

- name: Install/uninstall s/w

apt:

name: "{{a}}"

state: "{{b}}"

update\_cache: "{{c}}"

...

To run the above playbook to uninstall git

ansible-playbook playbook1.yml --extra-vars "a=git b=absent c=no" -b

We can use the same playbook to work on some other set of s/w's like install java

ansible-playbook playbook1.yml --extra-vars "a=openjdk-11-jdk b=present c=no" -b

Playscope varibles

=======================

These varibales are defined within a playbook and they have the

least priority

vim playbook2.yml

---

- name: Install/unistall sw applications

hosts: all

vars:

- a: git

- b: present

- c: yes

tasks:

- name: Install/unisntall

apt:

name: "{{a}}"

state: "{{b}}"

update\_cache: "{{c}}"

...

To run the above playbook

ansible-playbook playbook2.yml

Host scope variables

=============================

These are classified into 2 types

1 Variables to work on a group of hosts

2 Variables to work on a single host

Grouping in inventory file try to give the private\_ip can be presented

sudo vim /etc/ansible/hosts

[webserver]

172.31.30.86

172.31.18.115

Variables to work on a group of hosts

===========================================

1 Go to the folder where all playbook are present

cd path\_of\_playbooks\_folder

2 Create a directory "group\_vars" and move into it

mkdir group\_vars

cd group\_vars

3 Create a file whose name is same as group name from inventroy

vim webserver

---

a: maven

b: present

c: yes

...

4 Go back to the folder where the playbooks are present

cd ..

Create a palybook to use the above variables

vim playbok7.yml

---

- name: Create user using host scope varibales

hosts: webserver

tasks:

- name: Install maven

apt:

name: "{{a}}"

state: "{{b}}"

update\_cache: "{{c}}"

...

To run the playbook

ansible-playbook playbook7.yml -b

=================================================================

Variables to work on a single hosts

==========================================

1 Go to the folder where all playbook are present

cd path\_of\_playbooks\_folder

2 Create a directory "host\_vars" and move into it

mkdir host\_vars

cd host\_vars

3 Create a file whose name is same as ipaddress of one machine from inventory file

vim 172.31.86.213 ( private\_ip of one host)

---

a: Radha

b: intelliqit

c: 111111

d: /home/Radha

e: /bin/bash

...

4 Go back to the folder where the playbooks are present

cd ..

5 Create a playbook to use the above variables

vim playbook8.yml

---

- name: Install firewall using host variables

hosts: 172.31.86.213 ( private\_ip of one host)

tasks:

- name: Create user

user:

name: "{{a}}"

password: "{{b}}"

uid: "{{c}}"

home: "{{d}}"

shell: "{{e}}"

...

To run tha above playbook

ansible-playbook playbook8.yml -b

=========================================================

Loops in Ansible

=========================

Loops can be implemented in Ansible using

with\_items and with\_sequence

vim playbook4.yml

---

- name: Install s.w applications

hosts: all

tasks:

- name: Install s/w

apt:

name: ["tree","maven"]

state: present

update\_cache: yes

…

To run tha above playbook

ansible-playbook playbook4.yml -b

example for user creation with loops

vi userloop.yml

---

- name: create users using a loop

hosts: all

tasks:

- name: create users

user:

name: "{{item.name}}"

state: present

groups: "{{item.groups }}"

loop:

- name: anna

groups: wheel

- name: linda

groups: users

- name: bob

groups: users

…

To run playbook>>> ansible -playbook userloop.yml -b

Handlers

===================

1 Handlers are modules that are executed if some other module is executed

succesfully and it has made some changes.

2 Handlers are only executed after all the modules in the tasks section are executed

3 Handlers are executed in the order that they are mentioned in the handlers section and not in the order that they are called in the tasks section

4 Even if a handler is called multiple times in the tasks section it will

be executed only once

vi apache.yml ( without using handlers)

---

- hosts: all

name: Install

become: yes

tasks:

- name: Install Apache2 on Ubuntu server

apt:

name: apache2

state: present

update\_cache: yes

...

vi apache1.yml ( with using handlers)

---

- hosts: all

name: Install

become: yes

tasks:

- name: Install Apache2 on Ubuntu server

apt:

name: apache2

state: present

update\_cache: yes

notify:

- Restart apache2

handlers:

- name: Restart apache2

service:

name: apache2

state: restarted

...

ansible-playbook apache1.yml -b

===========================

Ansible Vault

===================

This is a feature of ansible which allows us to protect the playbooks

via a password.Playbooks created using vault can be viewed,edited or

executed only if we know the password

1 To create a vault playbook

ansible-vault create playbook\_name.yml

2 To view the content of a vault playbook

ansible-vault view playbook\_name.yml

3 To edit the content of a vault playbook

ansible-vault edit playbook\_name.yml

4 To convert an ordinary playbook into a vault playbook

ansible-vault encrypt playbook\_name.yml

5 To convert a vault playbook into an ordinary playbook

ansible-vault decrypt playbook\_name.yml

6 To reset the password of a vault playbook

ansible-vault rekey playbook\_name.yml

=========================================================

include module

=================

This is used to call child playbooks from the level of a parnet playbook

Child playbook

--------------------

vim playbook6.yml

---

- name: Copy /etc/passwd file

copy:

src: /etc/passwd

dest: /tmp

...

Parent playbook

---------------------

vim playbook7.yml

---

- name: Call child playbooks

hosts: all

tasks:

- name: Call child playbook

include: playbook6.yml

...

To execute

ansible-playbook playbook7.yml -b

===============================

<https://www.drpraize.com/2020/07/install-jenkins-using-ansible-playbook_29.html>

<https://www.coachdevops.com/2020/11/ansible-playbook-for-java-8.html>

<https://www.linuxtechi.com/use-handlers-in-ansible-playbook/>

https://youtu.be/3\_VEiMPTvDw

a