**Configuration Management**

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

This is process of configuring remote servers from one point of control.

Advantages

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

1) Provisioning of servers

The applications that should be installed on server can be done very quickly from a single centralized location.

2) Idempotent

Configuration management tools are used to bring the server to a particular state, called as desired state. If a server already in the desired state, configuration management tools will not reconfigure that server.

Note: Configuration management tools cannot be used for installing OS from the scratch.

They can be used only for managing the applications on top of the OS.

Configuration management tools - Ansible, chef, puppet, salt etc

++++++++++++++++++++++++++++++++++++++++++

Ansible -- It is a open source configuration management tool, created using Python.

Main machine in which ansible is installed, is called as controller.

Remote severs that Ansible configures, are called as managed nodes.

Ansible uses agent less policy for configures remote servers i.e. Ansible is installed only on 1 machine, and we do not require any client side software to be installed on the remote servers.

Ansible performs configuration management through password less ssh.

++++++++++++++++++++++++++++++++++

Create 4 Servers ( Ubuntu 18 )

1 is controller

3 are managed nodes

Name the instances as

Controller

Server1

Server2

Server3

Ubuntu machines default come with Python3

Ansible supports Python2

We need to install Python2

Connect server1

Check the version

$ python3 --version

To Install Python2

$ sudo apt-get update

$ sudo apt-get dist-upgrade ( It will point to older apt repository where python2 is available)

$ sudo apt-get install -y python2.7 python-pip

$ sudo apt-get install python3-pip

Now check the version of python

$ python --version

+++++++++++++++++++++++++

Establish password less ssh connection

$ sudo passwd ubuntu

( lets give the password as ubuntu only )

$ sudo vim /etc/ssh/sshd\_config

change

PasswordAuthentication yes (line no. 56)

Save and QUIT

$ sudo service ssh restart

$ exit

++++++++++++++++++

Repeat the same steps in server2 and server3

++++++++++++++++

Now, Connect to controller

Even in controller also python2 version should be available

(So, run the same commands)

$ sudo apt-get update

$ sudo apt-get dist-upgrade

$ sudo apt-get install -y python2.7 python-pip

Now check the version of python

$ python --version

+++++++++++++++++++

Now , We need to generate ssh connections

$ ssh-keygen

Now copy the key to managed nodes

$ ssh-copy-id ubuntu@172.31.0.98 ( private Ip of server1 )

$ ssh-copy-id ubuntu@172.31.1.183 ( private Ip of server2 )

$ ssh-copy-id ubuntu@172.31.14.179 ( private Ip of server3 )

+++++++++++++++++++++

Installing ansible now

Connect to controller.

$ sudo apt-get install software-properties-common

( software-properties-common , is a base package which is required to install ansible )

$ sudo apt-add-repository ppa:ansible/ansible

$ sudo apt-get update

$ sudo apt-get install -y ansible

+++++++++++++++++

To check ther version of ansible

$ ansible --version

++++++++++++

Write the ip address of nodes in the inventory file

$ cd /etc/ansible

$ ls

$ sudo vim hosts

insert the private ip addresss of 3 servers

save and quit

$ ls -la ( to see the list in the current machine )

$ ansible all -a 'ls -la' ( you will get the list of the files in all managed nodes )

2 Ways ansible can

1) adhoc commands

2) playbooks

adhoc commands

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

**Important modules in ansible**

1) command - This module is used for executing basic linux commands on managed nodes.

2) shell - This module is used to execute commands which involved redirection and piping and to execute shell scripts on managed nodes.

3) ping -- This module is used to check if the remote server is pingable or not.

4) user -- This module is used for user management like create user, setting password, assign home directory etc

5) copy -- This module is used to copy the files and folders from controller to managed nodes

6) fetch -- This module is used to copy files and folder from managed nodes to controller

7) file -- This module is used for creating or deleting files and folders on managed nodes.

8) stat -- Used to capture detailed information about files and folders present in managed nodes.

9) debug -- Used to display output of any module

10) apt -- Used for performing package management on managed nodes ie installing softwares / upgrading repositories etc . It works on ubuntu, debin flavours of linux.

11) yum -- similar to apt module. It works on Red hat linux, centos etc

12) git -- used to perform git version controlling on managed nodes

13) replace -- This is used to replace specific text in configuration file with some other text.

14) service -- used for starting / stopping / restarting services on managed nodes.

15) include -- Used for calling child play books from parent play book

16) url -- useful in checking if remote url is reachable or not.

17) docker\_container -- used to execute docker commands related to container management on managed nodes

18) docker\_image -- used to execute commands related to docker images on managed nodes.

19) docker\_login -- used to login to docker hub from managed nodes.

20) setup -- used to capturing system information related to the managed nodes.

++++++++++++++++++++++++++++++++++++++++++++++

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

$ ansible all -i /etc/ansible/hosts -m command -a 'touch file1'

To check the file which is created

$ ssh 172.31.2.173 ( this command will go that machine )

$ ls

$ exit ( to come back to controller )

++++++++++++++++

To install docker in all managed nodes

$ ansible all -i /etc/ansible/hosts -m shell -a 'curl -fsSL https://get.docker.com -o get-docker.sh'

$ ansible all -i /etc/ansible/hosts -m shell -a 'sh get-docker.sh'

+++++++++++++

To check docker is installed or not

$ ssh 172.31.2.173

$ docker --version

$ exit ( to come back to controller )

+++++++++++++++++

Notes:

Ansible performs remote configurations in 2 ways

1) using adhoc commands

2) using play books

Syntax of adhoc commands

$ ansible all/group\_name/ipaddress -i path\_of\_inventory\_file -m modulename -a 'arguments'

+++++++++++++++++

Ansible command module to check the memory info on all managed nodes

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

++++++++++++++++++

To open the default inventory file

$ sudo vim /etc/ansible/hosts

( Observation: 3 ip address are available )

+++++++++++++++++

Now, I copy the first two IP address ( in a new notepad file )

quit the inventory file

+++++++++++++

Create my own inventory file

$ vim myinventory

go to insert mode

paste two ip address

save and quit

+++++++++++

To check the inventory file

$ cat myinventory

+++++++++++

$ ansible all -i myinventory -m command -a 'free'

Observation: free command works on only two machines

+++++++++++++

If you do not mention the inventory file, it takes default inventory file.

ex:

$ ansible all -m command -a 'free'

+++++++++++++++++++++++++++++++

command module is the default module in ansible

$ ansible all -a 'free'

++++++++++++++++++++++

Note:

The defualt inventory file is /etc/ansible/hosts and when using this inventory file, we need not use -i option.

ex:

$ ansible all -m command -a 'free'

The default module is module. When using command module we need not use -m option

ex:

$ ansible all -a 'free'

Shell Module

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

ansible command to execute ls -la and store the output into file1 on all the managed nodes.

$ ansible all -m shell -a 'ls -la > file2'

To check the file which is created

$ ssh 172.31.12.239

$ ls

$ exit ( to come back to controller )

+++++++++++++++

command to install docker on all managed nodes

$ ansible all -m shell -a 'curl -fsSL https://get.docker.com -o get-docker.sh'

$ ansible all -m shell -a 'sh get-docker.sh'

++++++++++++++++++++++++

User Module:

( From controller )

To create new user

$ sudo useradd sai

$ vim /etc/passwd ( User will be created in this file )

To set the password

$ sudo passwd sai ( sai is the username)

+++++++++++

Now, i want to create user in all managed nodes

$ ansible all -m user -a 'name=anu password=sunil'

( we ger error : permission denied )

$ ansible all -m user -a 'name=anu password=sunil' -b ( become , for higher privileges on managed nodes )

++++++++++++++

To check if user is create or not

$ ssh 172.31.12.239

$ vim /etc/passwd

$ exit

+++++++++++++++++++++

Command to create user and set home directory, user id, default working shell etc

Another example

$ ansible all -m user -a 'name=Ravi password=freefree uid=1234 comment="A regular user" home=/home/ubuntu/Ravi shell=/bin/bash' -b

To check for the new user

$ ssh 172.31.44.218

$ vim /etc/passwd

++++++++++

Install git in all managed nodes

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

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

Observation:

We get "changed": false

( That means git is already installed on it. The command has no effect in the nodes)

Now , run the below command

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

( absent means - uninstall )

output, we get in yellow color

( scroll up ) we get "changed":true

( The command is effected the instance )

Now if we run the below command ( with present option )

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

we get "changed":true

Notes:

apt module -- This is used for package management.

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

state=present is for installation

state=latest for upgradation

state=absent for uninstallation

++++++++++++++++++++++++++++++++++++++++++

I want to update apt-repository and install tomcat9

ansible all -m apt -a 'name=tomcat9 state=present update\_cache=yes' -b

The above command will update apt repository and install tomcat9

To update apt-repository on managed nodes update\_cache=yes is used

+++++++++++++++++++++++++