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: Cofiguration 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.

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

++

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

Main machine in which anisble 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 ie Ansible is installed only on 1 machine, and we do not require any client side software to be installed on the remote serers.

Ansible performs configuration management through password less ssh.

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

Create 3 Servers (Ubuntu 18)

1 master

2 nodes

Ubuntu machines default come with Python3

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

Establish password less ssh connection

$ sudo su -

$ sudo passwd ubuntu

(lets give the password as ubuntu only)

```
ubuntu@ip-172-31-34-56:~$ sudo su -
root@ip-172-31-34-56:~# sudo passwd ubuntu
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@ip-172-31-34-56:~#
```

$ sudo vim /etc/ssh/sshd_config

change

PasswordAuthentication yes

Save and QUIT

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no
```

$ sudo service ssh restart

$ exit

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

Repeat the same steps in server2 and server3

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

Now, Connect to controller

Now , We need to generate ssh connections

$ ssh-keygen

```
ubuntu@ip-172-31-36-77:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa.
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:G2lYU7qYu9y/wLHj6VPkwvG1v9au607tQZoSpYFXWVk ubuntu@ip-172-31-36-77
The key's randomart image is:
+---[RSA 2048]----+
|          .  .o.E|
|         o. .. . |
|        +. o .   |
|       =.+o =    |
|      +.S= + . . |
|       +o=+ o +. |
|      . *o . +.o.|
```

Now copy the key to managed nodes

$ ssh-copy-id ubuntu@172.31.34.56 (private Ip of node1)

```
ubuntu@ip-172-31-36-77:~$ ssh-copy-id ubuntu@172.31.34.56
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ubuntu/.ssh/id_rsa.pub"
The authenticity of host '172.31.34.56 (172.31.34.56)' can't be established.
ECDSA key fingerprint is SHA256:5QgbZOhcKOY2G4VytGUhHbuf1oqP82POIZgy/OSLFHM.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already
installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install t
he new keys
ubuntu@172.31.34.56's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ubuntu@172.31.34.56'"
and check to make sure that only the key(s) you wanted were added.

ubuntu@ip-172-31-36-77:~$
```

$ ssh-copy-id ubuntu@ 172.31.37.253 (private Ip of node2)

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

Installing ansible now

Connect to controller.

$ sudo apt-get install software-properties-common

```
ubuntu@ip-172-31-36-77:~$ sudo apt-get install software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
software-properties-common is already the newest version (0.96.24.32.18).
software-properties-common set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-36-77:~$
```

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

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

```
ubuntu@ip-172-31-36-77:~$ sudo apt-add-repository ppa:ansible/ansible
 Ansible is a radically simple IT automation platform that makes your applications and systems easier to
deploy. Avoid writing scripts or custom code to deploy and update your applications— automate in a langua
ge that approaches plain English, using SSH, with no agents to install on remote systems.

http://ansible.com/

If you face any issues while installing Ansible PPA, file an issue here:
https://github.com/ansible-community/ppa/issues
 More info: https://launchpad.net/~ansible/+archive/ubuntu/ansible
Press [ENTER] to continue or Ctrl-c to cancel adding it.
```

$ sudo apt-get update

```
ubuntu@ip-172-31-36-77:~$ sudo apt-get update
Hit:1 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
ubuntu@ip-172-31-36-77:~$
```

$ sudo apt-get install -y ansible

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

To check ther version of ansible

$ ansible --version

```
ubuntu@ip-172-31-36-77:~$ ansible --version
ansible 2.9.27
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/home/ubuntu/.ansible/plugins/modules', u'/usr/share/ansible/plugins
/modules']
  ansible python module location = /usr/lib/python2.7/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 2.7.17 (default, Jul  1 2022, 15:56:32) [GCC 7.5.0]
ubuntu@ip-172-31-36-77:~$
```

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

Write the ip address of nodes in the inventory file

$ cd /etc/ansible

```
  python version = 2.7.17 (default, Jul  1 2022, 15:56:32) [GCC 7.5.0]
ubuntu@ip-172-31-36-77:~$ cd /etc/ansible
ubuntu@ip-172-31-36-77:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-36-77:/etc/ansible$ sudo vim hosts
ubuntu@ip-172-31-36-77:/etc/ansible$
```

$ls

```
172.31.34.56
172.31.37.253
# This is the default ansible 'hosts' file.
#
# It should live in /etc/ansible/hosts
#
#   - Comments begin with the '#' character
#   - Blank lines are ignored
#   - Groups of hosts are delimited by [header] elements
#   - You can enter hostnames or ip addresses
#   - A hostname/ip can be a member of multiple groups
```

$ sudo vim hosts

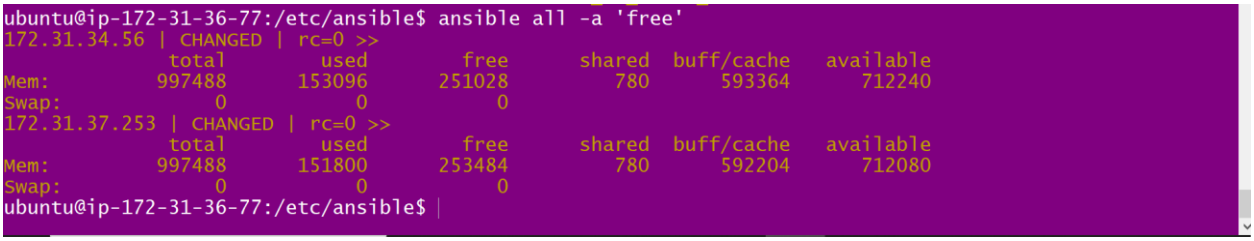
insert the private ip addresss of 3 servers

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)

```
ubuntu@ip-172-31-36-77:/etc/ansible$ sudo vim hosts
ubuntu@ip-172-31-36-77:/etc/ansible$ ansible all  -a  'ls  -la'
172.31.34.56 | CHANGED | rc=0 >>
total 40
drwxr-xr-x 6 ubuntu ubuntu 4096 Jul 30 04:42 .
drwxr-xr-x 3 root   root   4096 Jul 30 03:43 ..
drwx------ 3 ubuntu ubuntu 4096 Jul 30 04:42 .ansible
-rw------- 1 ubuntu ubuntu   23 Jul 30 03:53 .bash_history
-rw-r--r-- 1 ubuntu ubuntu  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Apr  4  2018 .bashrc
drwx------ 2 ubuntu ubuntu 4096 Jul 30 03:45 .cache
drwx------ 3 ubuntu ubuntu 4096 Jul 30 03:45 .gnupg
-rw-r--r-- 1 ubuntu ubuntu  807 Apr  4  2018 .profile
drwx------ 2 ubuntu ubuntu 4096 Jul 30 03:43 .ssh
-rw-r--r-- 1 ubuntu ubuntu    0 Jul 30 03:47 .sudo_as_admin_successful
172.31.37.253 | CHANGED | rc=0 >>
total 40
drwxr-xr-x 6 ubuntu ubuntu 4096 Jul 30 04:42 .
drwxr-xr-x 3 root   root   4096 Jul 30 04:17 ..
drwx------ 3 ubuntu ubuntu 4096 Jul 30 04:42 .ansible
-rw------- 1 ubuntu ubuntu   15 Jul 30 04:21 .bash_history
-rw-r--r-- 1 ubuntu ubuntu  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Apr  4  2018 .bashrc
drwx------ 2 ubuntu ubuntu 4096 Jul 30 04:19 .cache
```



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, debain 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 / stoping / restarting services on managed nodes.

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

16) uri -- 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'

```
ubuntu@ip-172-31-36-77:/etc/ansible$ ansible all -i /etc/ansible/hosts -m command -a 'free'
172.31.34.56 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:         997488      153328      250784         780      593376      712028
Swap:             0           0           0
172.31.37.253 | CHANGED | rc=0 >>
              total        used        free      shared  buff/cache   available
Mem:         997488      151620      253484         780      592384      712276
Swap:             0           0           0
ubuntu@ip-172-31-36-77:/etc/ansible$
```

Yellow means that nodes are in executed in the two nodes and made some changes

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

To check the file which is created

$ ssh 172.31.10.243 (this command will go that machine)

$ ls

```
ubuntu@ip-172-31-36-77:/etc/ansible$ ssh 172.31.37.253
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1078-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Jul 30 04:53:51 UTC 2022

  System load:  0.0               Processes:           93
  Usage of /:   16.4% of 7.58GB   Users logged in:     0
  Memory usage: 20%               IP address for eth0: 172.31.37.253
  Swap usage:   0%


0 updates can be applied immediately.

New release '20.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Last login: Sat Jul 30 04:47:34 2022 from 172.31.36.77
ubuntu@ip-172-31-37-253:~$ ls
file1
ubuntu@ip-172-31-37-253:~$
```

$ exit (to come back to controller)

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

To install docker in all managed nodes

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

```
ubuntu@ip-172-31-36-77:/$ ansible all -i /etc/ansible/hosts -m shell -a 'sh get-docker.sh'

172.31.37.253 | CHANGED | rc=0 >>
# Executing docker install script, commit: b2e29ef7a9a89840d2333637f7d1900a83e7153f
Client: Docker Engine - Community
 Version:           20.10.17
 API version:       1.41
 Go version:        go1.17.11
 Git commit:        100c701
 Built:             Mon Jun  6 23:02:56 2022
 OS/Arch:           linux/amd64
```

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

To check docker is installed or not

$ ssh 172.31.10.243

$ docker --version

```
Last login: Sat Jul 30 05:00:44 2022 from 172.31.36.77
ubuntu@ip-172-31-37-253:~$ docker --version
Docker version 20.10.17, build 100c701
ubuntu@ip-172-31-37-253:~$
```

$ exit (to come back to controller)

```
ubuntu@ip-172-31-36-77:/$ ansible all -m apt  -a 'name=git  state=present' -b
172.31.37.253 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1659163042,
    "cache_updated": false,
    "changed": false
}
172.31.34.56 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "cache_update_time": 1659163042,
    "cache_updated": false,
    "changed": false
}
ubuntu@ip-172-31-36-77:/$
```

nstall 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)

I wan to update apt-repositoty and install tomcat8

ansible all -m apt -a 'name=tomcat8 state=present update_cache=yes' -b

```
}
ubuntu@ip-172-31-36-77:/$ ansible all -m  apt -a 'name=tomcat8 state=present  update_cache=yes' -b
```

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

Notes:

Ansible performs remote configurations in 2 ways

1) using adhoc commands

2) using play books

Syntx 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 172.31.10.243

$ ls

$ exit (to come back to controller)

Play books

Notes:

Adhoc commands are capable of working only on one module and one set of arguments.

When we want to perform complex configuration management activities,

adhoc commands will be difficult to manage.

In such scenarios, we use play books.

Play book is combination of plays.

Each play is designed to do some activity on the managed nodes.

These plays are created to work on single host or a group of hosts or all the hosts.

The main advantage of play books is reusability.

Play books are created using yaml files.

```
$ mkdir playbooks
$ cd playbooks
$ vim playbook1.yml
INSERT mode

---
- name: Install git and clone a remote repository
  hosts: all
  tasks:
    - name: Install git
      apt:
       name: git
```

state: present

update_cache: yes

- name: clone remote git repository

git:

repo: https://github.com/sunilkumark11/git-9am-batch.git

dest: /home/ubuntu/newgit

...

```
ubuntu@ip-172-31-36-77:~$ mkdir  playbooks
ubuntu@ip-172-31-36-77:~$ cd playbooks
ubuntu@ip-172-31-36-77:~/playbooks$ vim playbook1.yml
ubuntu@ip-172-31-36-77:~/playbooks$ ansible-playbook  playbook1.yml  -b

PLAY [Install git and clone a remote repository] ***********************************************

TASK [Gathering Facts] *************************************************************************
ok: [172.31.37.253]
ok: [172.31.34.56]

TASK [Install git] *****************************************************************************
ok: [172.31.37.253]
ok: [172.31.34.56]

TASK [clone remote git repository] *************************************************************
changed: [172.31.37.253]
changed: [172.31.34.56]

PLAY RECAP *************************************************************************************
172.31.34.56               : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
172.31.37.253              : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

```
---
- name: Install git and clone a remote repository
  hosts: all
  tasks:
    - name: Install git
      apt:
       name: git
       state: present
       update_cache: yes
    - name: clone remote git repository
      git:
        repo: https://github.com/archanareddyse/ansible1
        dest: /home/ubuntu/newgit
...
```

To check the syntax:

$ ansible-playbook playbook1.yml --syntax-check

(Do not use tab when creating yml file)

To run the playbook

$ ansible-playbook playbook1.yml -b

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