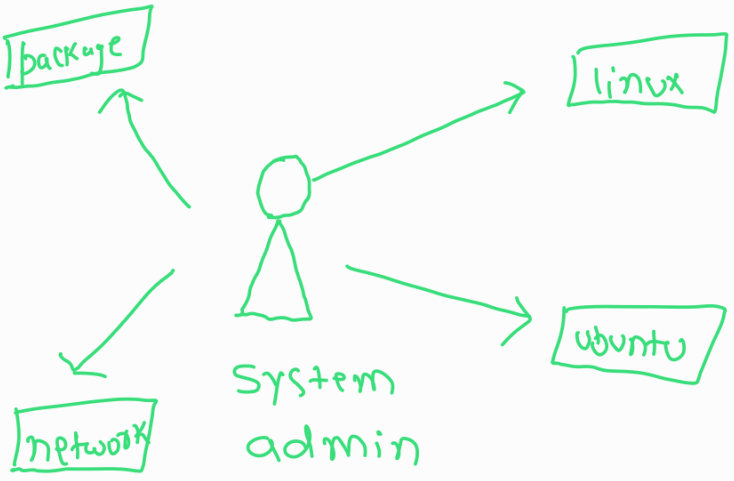
What is Ansible ?

↳ Configuration Management tool

→



System admin manages all the Server in an organization.
Previously they were doing it manually but now automation is introduced using Ansible.

Ansible is easy. It uses YAML file for Scripting.

Ansible Tower is a GUI of Ansible provided by RedHat but Ansible Tower is a

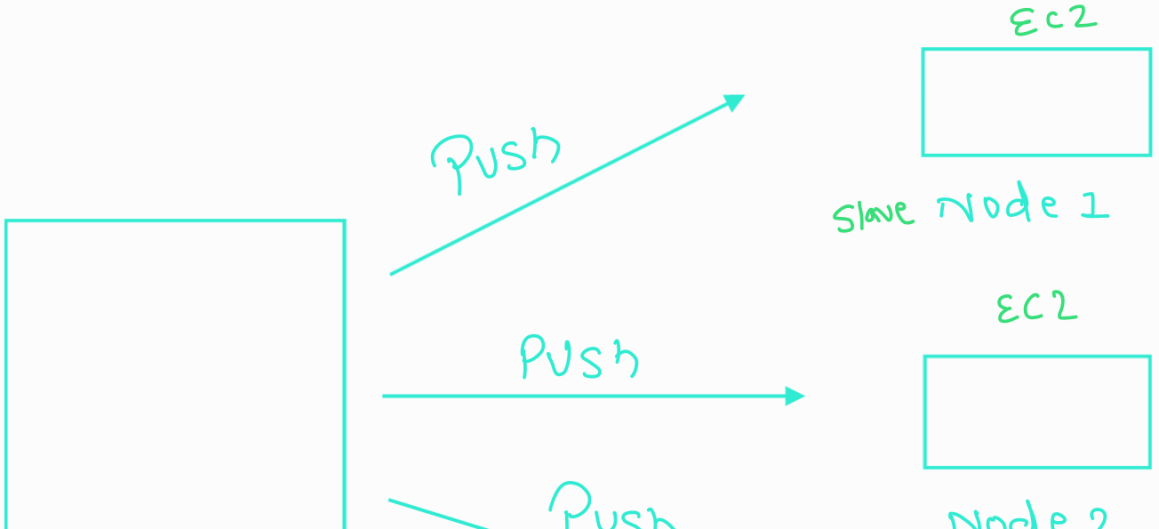
paid service.

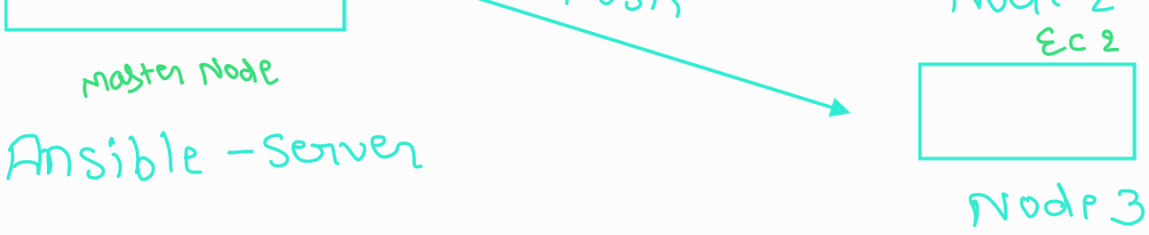
→ Ansible is a configuration tool by which we can install packages or we can configure packages to 100's of server at a time.

→ Ansible is available for RHEL, Debian, centos, oracle linux etc.

→ can use this tool whether your servers are in on-premises or in the cloud.

→ It turns your code into infrastructure.
(IAC) - Infrastructure as a code

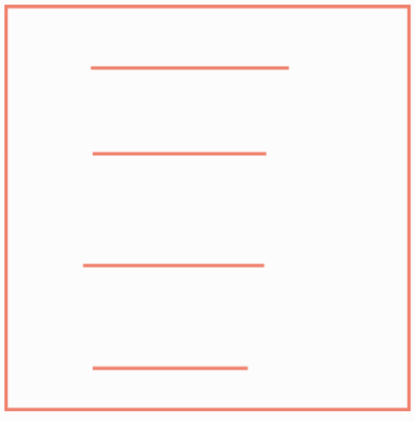




YAML → yet another Markup language
↳ Human Readable

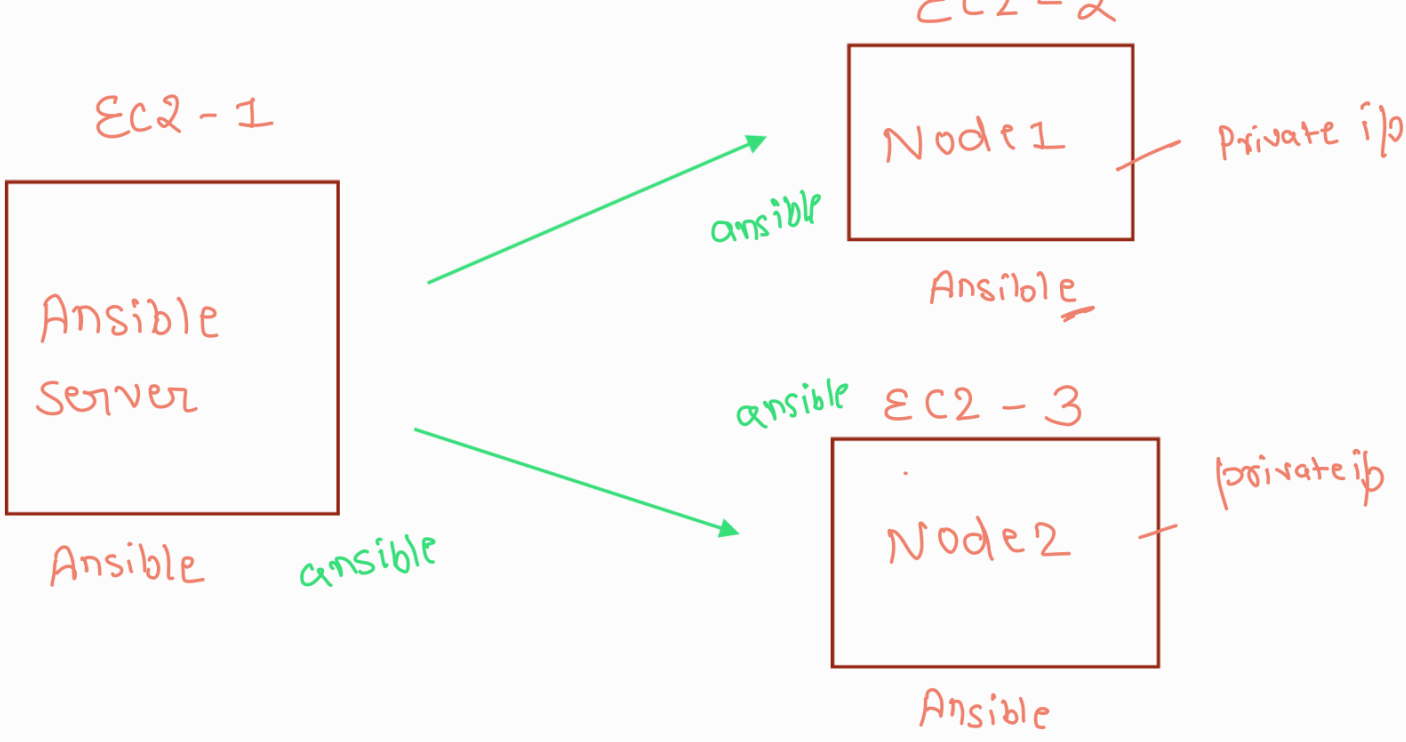
Ansible is agentless. Ansible need not to be installed in node side.

→ works directly through SSH.
→ Ansible use SSH to communicate with Node.



→ Playbook
.yaml

Go to Aws account → Create 3 EC2 instances in same Az.



→ Take access of all machines via putty.

→ Now go inside ansible server and download ansible package.

ec2-user - sudo su root

→ wget (http://) ------ epel-release-latest)

→ ls

→ yum install epel-release-latest ---.rpm

↓ -y

extra packages for enterprise linux.

→ yum update -y

→ yum install git python python-level
python-pip openssl ansible -y

* Now go to hosts file inside ansible server and paste private-ip of node1 & Node2.

* vi /etc/ansible/hosts

Now this hosts file is only working after updating ansible.cfg file:=

* vi /etc/ansible/cfg

inventory = /etc/ansible/hosts

sudo-user = root

↳ Commented command

↳ Remove # to uncomment

Ansible Server will require root password to communicate with Nodes.

So, in place of providing root password in real-world scenario, we create a new user in each node and give that user password to ansible server.

→ Now create one user, in all the three instances.

[ ]# useradd ansible

Now set passwd for this user

[ ]# passwd ansible

— Enter password

— Enter confirm passwd

[root]# su - ansible

[ansible@ip] ↵

→ This ansible user don't have sudo priviledges right now. If you want to give sudo priviledge to ansible user

[root@ip] visudo

Now go inside this file

root ALL=(ALL) ALL

+ ansible ALL=(ALL) NOPASSWD: ALL

privilege power

same thing you have to do in other nodes also.

[ansible@ip] sudo yum install httpd

↓ -y

this sudo command you have to add before because you are not a root user. Just you are having the root priviledges.

[ansible@ip] ssh 172.31.41.240

o/p - permission denied.

- we have to do same changes in

we have to do some changes in sshd-config file go to ansible server.

[root@ip] vi /etc/ssh/sshd-config

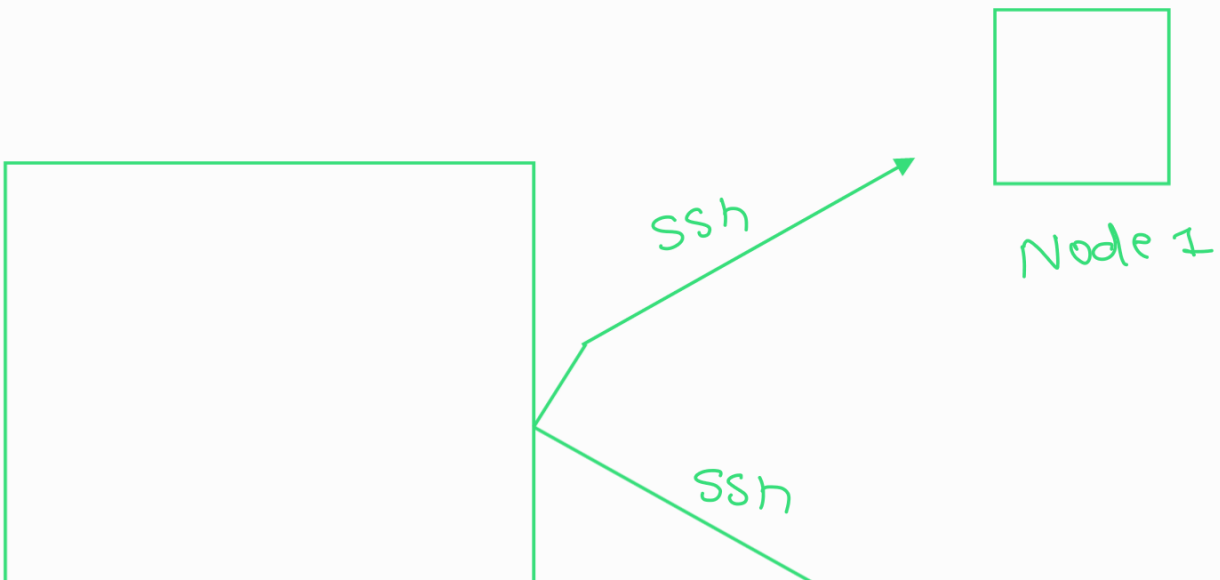
Do some changes and save the file.

Do this work in Node1 and Node2 also.

[root@ip] su -ansible

[ansible@ip] ssh 172.31.41.240

↳ Now it ask for password, enter the password, after that you will be inside Node 1.



(i) Ad-hoc (Temporary) commands (simple linux)
↳ No idempotency

(ii) Modules → Single work → httpd -install

(iii) Playbooks → (More than one Modules)
(combination of modules)
↓
httpd start webserver } in a single playbook

→ Why not Ad-hoc ?

We use Ad-hoc only for temporary process. Ad-hoc do the same work again and again even if it is already done. (No idempotency)

e.g → touch file1
touch file1 } it will overwrite the file1 in second time.

su - ansible

e.g → ansible demo ← group -a "ls"
↓
argument

→ ansible all -a "touch file1"

→ ansible all -a "sudo yum install httpd -y"

→ become (sudo)

→ ansible all -ba "yum install httpd -y"

→ ansible all -ba "yum remove httpd -y"

Ansible Modules (Idempotency is present)

→ Ansible ships with a number of modules (called module library) that can be executed directly on remote hosts or through playbooks.

→ Stored in location for the inventory file /etc/ansible/hosts.

e.g → ansible demo -b -m yum -a
"pkg=httpd State=present"

ansible demo -b -m yum -a
"pkg=httpd State=latest"

YAML
install = present
uninstall = absent
Update = latest

→ Systemctl start httpd
→ service httpd start

group name { list of node ip }

ansible demo -b -m service "name=httpd
State=Started"

ansible demo -b -m yum -a
"pkg=httpd State=absent"

For Creating User

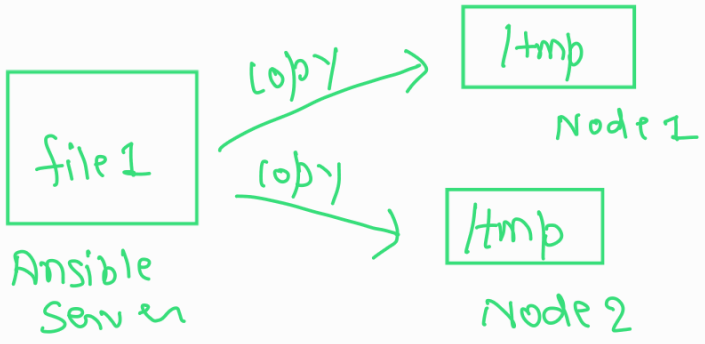
ansible demo -b -m user -a
"name=Kishan State=present"
optional

State = absent

To remove user

ansible demo -b -m copy -a

"src = file1 dest = /tmp"



ansible demo -m setup

↓ all the details/config of Nodes.

↳ Setup module only helps us to achieve idempotency by using ansible modules.

→ In Backend Ansible use this to get configuration of Nodes.

ansible demo -m setup -a "filter=*ipv4*"

Ansible playbook

vi target.yaml

```
- hosts: demo
  user: ansible
  become: yes
  connection: ssh
  gather_facts: true
```

3rd →
hosts: demonode
user: ansible
become: yes
connection: ssh

ansible-playbook target.yaml

demo -groupname

gather — ip
       — ip

ok = 1 ignore = 0

vi task.yaml

```
- hosts : demo
  user : ansible
  become: yes
  connection: ssh
  tasks :
   - name : install httpd on linux
     action : yum name=httpd state=installed
```

yum → module name

installed → Present

```
ansible-playbook task.yaml
```