Using Ansible for Orchestration

What is Ansible ?
Ansible as an orchestration/configuration management tool.
push based architecture for sending configurations
easy,fast to setup ,minimal requirements.IT automation,configuration,management
developed in python

pip - python package manager

What is Orchestration ?
The planning or coordination of the elements of a situation to produce a desired effect, especially surreptitiously

Types of Orchestration tools
- Push based vs pull based Conf mgmt
Puppet,chef are pulled based

Inventory location -> /etc/ansible/hosts

Ansible commands
ansible all -s -n shell -a 'uptime' -> uptime of all machines
‘date' -> data of all machines
'cat /etc/redhat-release' - redhat release of all machines
'mount' -> kind of mount on all the machines
'service sshd status' -> check the service status on all the machines
shell module is used here

Module
Set of commands and configurations written in sequence and executed to be on server is called module.

Playbooks
Playbook is set of modules that work together.
all the yaml files starts with 3 dashes on top
Sample YAML file

```
 ##########################################
  ---
 - host: webserver   // playbooks will be running on all webservers
 tasks :     // task you want to run
   -name:Installs nginx web server
   apt:pkg=ngnix  state=installed
  update_cache=true  // specific the package to be installed
```

```
          notify :
           -start nginx
 handlers: //tasks that are get run after certain triggers 'notify' is the trigger here,handler will be
                //run only one no matter how many times playbook runs
 -name : start nginx
  service:name=nginx state=started
```

##

How to run playebook

```
ansible-playbook filename.yml
```

Ansible Installation and setup

Control machine   // 10.10.10.6

Step 1

update all packages // yum update -y

Step 2

Ansible package is not available in default yum repository so we will enable epel repo

Install epel   // extrapackages for enterprise linux

```
wget http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
rpm -ivh epel-release-6-8.noarch.rpm
```

Step 3

Install ansible

Command : ***yum install ansible***

Check version : ***ansible --version***

Step 4

Post Installation

Add the target node in a file

Command ***echo "10.10.10.8" > ~/ansible_hosts***

Export this file as Ansible inventory file

Command ***export ANSIBLE_INVENTORY=~/ansible_hosts***

Ping the node using ansible ping module.

```
ansible all -m ping --ask-pass
```

If it gives Error as follows :

```
SSH password:
10.10.10.6 | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).\r\n",
    "unreachable": true
}
```

These means SSH is not enabled for server machine.

Enabling SSH

Generating public SSH key in controller machine

command : ***ssh-keygen***

Your identification(private key) has been saved in : ***/root/.ssh/id_rsa.***

Your public key has been saved in ***/root/.ssh/id_rsa.pub.***

Now place public key on your server so that you can use SSH key authentication to login

use ssh-copy-id tool

Enabling password authentication

In order to copy public key to the server, password based authentication

step 1:

open sshd_config file -> vim /etc/ssh/sshd_config

step 2 :

ssh-copy-id username@remote_host -> ssh-copy-id root@10-10-10-8

login to remote host using following command if you are able to login without prompting for password then your ssh connection is configured now.

ssh username@remote_host -> ssh 'root@10.10.10.8'

How to login into remote server ->

```
[root@host-10-10-10-6 ~]# ssh 'root@10.10.10.8'
Last login: Tue Aug  8 00:35:10 2017 from 10.10.10.6
[root@host-10-10-10-8 ~]#
```

Step 3 :

Now disable Password Authentication

sudo nano /etc/ssh/sshd_config

set PasswordAuthentication to no

How to enable PasswordAuthentication ?

##Need of SSH-COPY-ID - COPY SSH KEY TO SERVER

-> ssh-copy-id install an SSH key on a server as an authorized key. Its purpose is to provision access without requiring a password for each login. This facilitates automated, passwordless logins and single sign-on using the SSH protocol.

Steps to Run Ansible modules:

```
##To list the tasks:
ansible-playbook -i <inventory-file> <playbook-file> --list-tasks
##To actually execute the tasks:
ansible-playbook -i <inventory-file> <playbook-file>
e.g. ansible-playbook -i prod-hosts ams.yaml
```

##Configuring SSH##
To do this, we can use a special utility called ssh-keygen, which is included with the standard OpenSSH suite of tools. By default, this will create a 2048 bit RSA key pair, which is fine for most uses.

SSH or secure shell, is an encrypted protocol

```
ssh-agent bash
ssh-add ~/.ssh/id_rsa
```

```
1. 10.10.10.6          hostame: host-10-10-10-6
2. 10.10.10.8          host-10-10-10-8
```

updated /etc/hosts file on both linux system

Firewall should be off on server

```
$ ansible atlanta -a "/sbin/reboot" -f 10 -> /usr/bin/ansible
 ansible atlanta -a "/usr/bin/foo" -u username -> run command with different username
 ansible raleigh -m shell -a 'echo $TERM ->
```

Playbooks :

Playbooks are a completely different way to use ansible than in adhoc task execution mode, and are particularly powerful.

Add hosts inventory -> open /etc/ansible/hosts file
-> create group of servers as providing list of ip addresses
-> [test-servers]
-> 10.10.10.6
-> check ansible connection with host -> ansible -m ping ' name-of-group-of-servers'

Making SSH connection with the node machine ->
write playbook to install nginx on node machine ->