What is ansible?

Ansible is an open source automation tool for configuration management.
It uses push based configuration.
Agent less: No need of an extra software in Target machine. It just needs SSH connection.

Ansible Architecture

Ansible has 2 parts - Ansible node and target node. We need to setup passwordless SSH connection from master to node.
Ansible works on python. So python is required in both ansible nodes and target nodes.
We execute our work using
1. Ad-hoc commands - we run these commands as and when we we require

2. Ansible playbooks - it is written in YAML. We have to plan the task and execute.

Steps to install ansible

1. Install python and ansible in ansible node
2. Install python in Target node
3. Setup SSH passwordless connection
4. Provide Target IP in /etc/ansible/hosts

Ad-hoc commands: it is used to execute simple commands without writing playbook

Ex: ansible all -m ping
ansible test -m ping
ansible 172.92.32.168 -m ping

ansible all -m shell -a 'uptime'
ansible prod -m shell -a 'date'
ansible all -m shell -a 'df -h'
ansible all -m shell -a 'ps -eaf | grep tomcat'
ansible 172.92.32.168 -m ping
ansible all -m shell -a 'sudo service sshd restart'
ansible all -m shell -a 'sudo reboot'

Inventory file/ hosts file

Path: **/etc/ansible/hosts**
*It contains the list of Target IP's
*Ansible will check in this file when we execute any adhoc command or playbook

Example for inventory file

```
[dev]
172.92.31.30
172.92.31.31

[test]
172.92.31.32
172.92.31.33

[prod]
172.92.31.34
172.92.31.35
```

Ansible playbooks: collection of tasks. It is written in YAML language and contains hosts, tasks to be executed.

Example for playbook

```
---
- hosts: all
  name: checking date in target servers
  tasks:
    - name: executing date command
      shell: date
```

Command to execute playbook:

ansible-playbook play.yml :- No verbose
ansible-playbook -v play.yml :- verbosity level 1
ansible-playbook -vv play.yml :- verbosity level 2
ansible-playbook -vvv play.yml :- verbosity level 3 / more logs

Ansible module : Ansible modules are the units of code can control system resources or execute system commands. Ansible provides a module library that you can execute directly on remote hosts or through playbooks.

Few important modules:

command, shell, copy, yum, apt, service, git, script, fetch, user

1. File module: used to create/modify/delete file or folder

state : touch , directory , absent

Ex 1: create a directory using file module

```
file:
  path: /home/ec2-user/dev-dir1
  state: directory
  mode: 0755
```

ex 2: if you want change user or group in file module ?

```
file:
  path: /home/ec2-user/dev-dir1
  state: directory
  mode: 0750
  owner: root
  group: root
become: yes
```

ex 3: To create multiple directories

```
file:
   path: /home/ec2-user/{{ item }}
   state: directory
   mode: 0755
   owner: root
   group: root
loop:
  - folder1
  - folder22/folder3
  - f3
become: yes
```

To create file--> state: touch
To create folder--> state: directory
To delete --> absent

2. Command module and shell module
Both are used to execute Linux commands. Shell module supports | , > , >> and combining 2 commands using ;

```
-name: execute date command
 command: date


-name: execute log rotate
 shell: find -type f -mtime +10 | xargs rm -rf
```

3. Script module: if we have the shell script in ansible node and we want to execute this in Target nodes.

```
 - name: execute update.sh
   script: /home/ec2-user/update.sh
```

how to execute multiple scripts ?

```
- name: execute script1.sh and script2.sh
  script: /home/ec2-user/{{ item }}
  loop:
    - script1.sh
    - f2/script2.sh
```

how to execute a script and pass the name at the run time ? how to use extra-vars ? what is extra-vars ?

```
---
- hosts: all
  name: execute
  tasks:
    - name: execute a shell script out multiple scripts
      script: /home/ec2-user/{{ scr }}


ansible-playbook -v script.yml --extra-vars "scr=myscript.sh"
```

What are extra vars ?
Ansible extra vars is a feature that allows you to specify dynamic values when executing the playbook.

4. Copy module: used to copy files from ansible machine to target machine

example1: to copy new.txt from master to target nodes

```
- name: copy new.txt to target nodes
  copy:
    src: "/home/ec2-user/ansible/playbooks/new.txt"
    dest: "/home/ec2-user"
```

example2: to copy new.txt from master to target nodes with req permissions and change onwership

```
- name: copy new.txt to target nodes
  copy:
    src: "/home/ec2-user/ansible/playbooks/new.txt"
    dest: "/home/ec2-user"
    mode: 0755
    owner: root
    group: root
  become: yes
```

example3: to copy new.txt from **one path in target to differnt in target node**

```
- name: copy new.txt to target nodes
  copy:
    src: "/home/ec2-user/ansible/playbooks/new.txt"
    dest: "/home/ec2-user"
  remote_src: yes
```

copy multiple files from master to target node ?

```
---
 - hosts: all
   tasks:
     - name: copy multiple files
       copy:
         src: /home/ec2-user/{{ item }}
         dest: /home/ec2-user
       loop:
         - f1
         - f2
         - f3
         - f4
```

copy diff files to different destinations ?

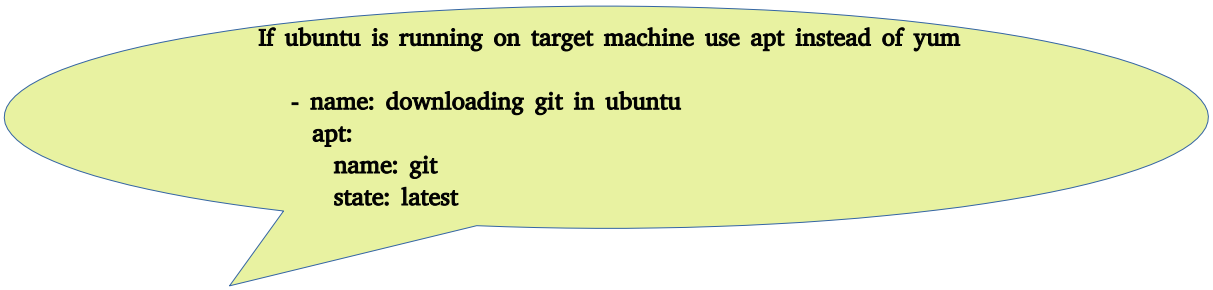
```
 - name: copy files
    copy:
       src: "{{ item.src }}"
       dest: "{{ item.dest }}"
       mode: 0755
       owner: root
       group: root
     loop:
        - { src: /home/ec2-user , dest: /home/amruth }
        - { src: /home/ec2-user/ansible , dest: /etc/new }
```

This will copy the file from ansible node to target node.
If you want to copy file in Target node from one path to another then use shell module or remote_src: yes

5. yum module: install packages from yum repository to target machines

```
- name: downloading maven
  yum:
    name: maven
    state: latest
```

state:--> latest , absent/removed - to uninstall the package



6. service module : used to start, stop , restart a service in target machines

```
- name: restart sshd
  service:
    name: sshd
    state: restarted
```

state:--> restarted , started , stoppped

7. fetch module: to copy files/folders from target to ansible master. It is opposite to copy module.

```
- name: copy from target machines to ansible
  fetch:
    src: /path in target
    dest: /path in master
```

Note: the dest path is not exactly same as we give. it will create folder with the name of target IP and it will create the folder structure similar to that of target machine.

8. git module: to download git repo

```
git:
   repo: https://sgithub.com/.../..git
   dest: /home/ec2-user/newfolder
```

1. Write a playbook to install and start a service

```
---
- hosts: all
  name: installing https and starting
  tasks:
     - name: install
       yum:
            name: https
            state: latest

     - name: start
       service:
            name: httpd
            state: started
```

Assignment:

1. Write a playbook to install tomcat
2. Write a playbook to install Jenkins
3. Write a playbook to install maven
4. Write a playbook to take code from git repository, build and deploy in tomcat servers

5. Write a single playbook to install git in dev servers and nginx in prod servers ? Or How to execute one task in first 5 servers and execute another task in another set of serevers ? Or How to combine 2 playbooks ?

Ans: using **multiple - play**

```
---
- hosts: dev
  name: action on dev
  tasks:
     - name: install git
       yum:
          name: install git
          state: latest

- hosts: prod
  name: action on prod
  tasks:
     - name: install nginx
       yum:
          name: install nginx
          state: latest
```

What is gather_facts ? What is the use of it ? **

It is a default module executed by ansible. It will check whether target servers are reachable or not.
If the servers are not reachable we will get to know in the beggining before executing any tasks, this is save our time. If we dont use gather_facts, there is possibility that the playbooks fails at later stage.

7. How to disable gather facts?
use below line after hosts

gather_facts: no or **gather_facts: false**

Ansible roles: roles allow us to create reusable tasks by grouping them

1. If we want to collaborate with other teams and share the existing snippets of playbook then we can use roles
2. If our playbook is large then we can split it into multiple roles and call these roles in our playbook.

Q1) How to create roles?

We can create manually or **use ansible galaxy to create**

Q2) what is ansible galaxy?
It is a command to create folder structure which is used in roles

Syntax: ansible-galaxy init <role name>
ex: ansible-galaxy init jenkins-role

Path: /etc/ansible/roles

This command will create 8 folders and 8 files

tasks:
--> main.yml : we have to write our tasks here. **********

vars:
--> main.yml : used to assign values for the variable in the form of yaml file ********

defaults:
--> main.yml : default files or default variables for the role

files: we can place the files which are being used by our playbook

handlers
--> main.yml : these special tasks. these are executed only when notified. *********

Templates: folder to place JINJA templates

meta:
--> main.yml : contains meta information

Readme.Md --> file contains readable info

Tests:
--> inventory: inventory for testing
--> test.yml: contains tasks/tests

Q4) write a ad-hoc command to check OS version in Target machines

ansible all -m shell -a 'cat /etc/os-release'

8. Create ansible roles and call them in playbook

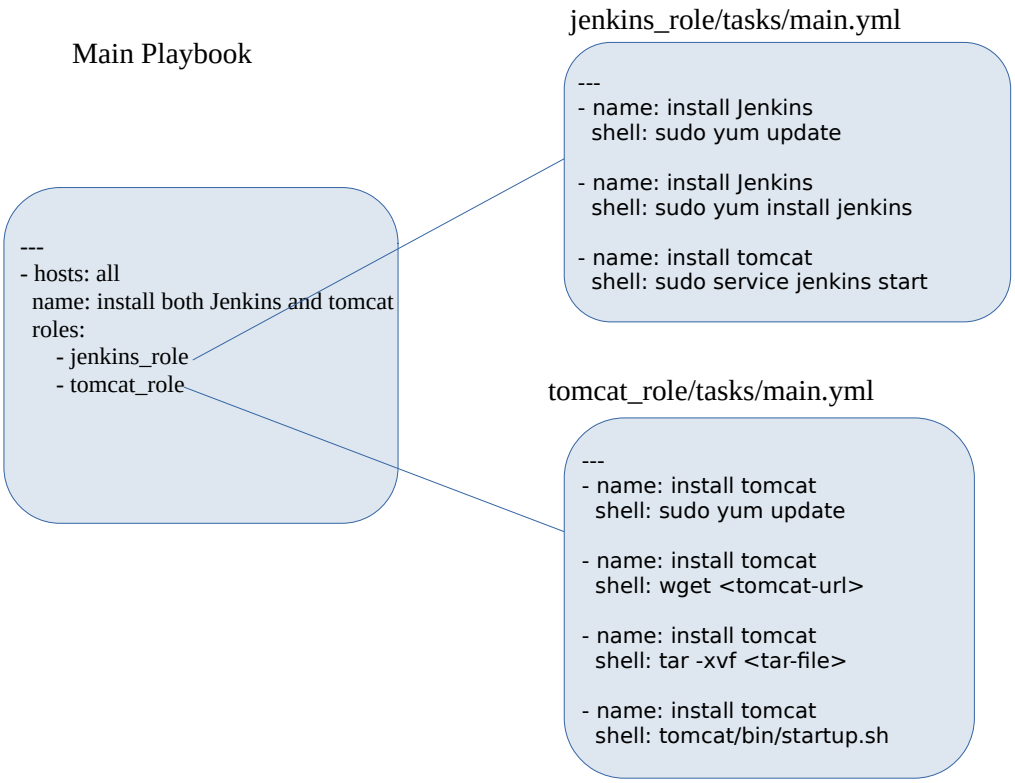
Step1: go to **/etc/ansible/roles**

Step2: ansible-galaxy init jenkins_role
ansible-galaxy init tomcat_role

Step3: go to main.yml of jenkins_role and write task to install Jenkins

Step4: go to main.yml of tomcat_role and write task to install tomcat

Step5: Call these roles in the playbook



9. What is verbose?

It gives description of tasks while running

10. How to activate verbose?

-V --> less details
-VV --> medium level
-VVV --> more details

11. Write a playbook that you have created in your company

I have written this playbook to download code from git , build and deploy this to target servers in dev environment

I have written playbooks for production environment for deployment of non-docker apps

```
---
- hosts: 172.92.32.46
  name: build
  tasks:
       - name: checking pwd
         shell: PWD
       - name: create a folder
         file:
            path: /home/cdh/reporter
            state: directory
            mode: 0755
       - name: clone git repo
          git:
            repo: https://
            dest: /home/ec2-user/
       - name: build using maven
          shell:   |
               cd /home/ec2-user/reporter
                mvn clean install
       - name: copy build from maven Machine to ansible node
         fetch:
            src: /home/ec2-user/reporter/target/reporter.war
            dest: /builds
       - name: delete these folders as they are not required
         file:
             path: /home/cdh/reporter
             state: absent


- hosts: prod
  name: deployment
  tasks:
       - name: deploying the build to tomcat
         copy:
              src: /home/ec2-user/172.92.32.46/builds
               dest: /home/ec2-user/tomcat/webapps
       - name: restarting tomcat if needed
          shell:   |
               sh /home/ec2-user/tomcat/bin/shutdown.sh
               sh /home/ec2-user/tomcat/bin/startup.sh
          become: yes
```

12) How to run tasks serially or IP wise? ****

normal flow : Ansible will execute tasks parallely in all servers?

How to execute taks serially ? or server-wise ? or I want to run all the taks in server1, then server2 and so on....

serial: 1 --> It will execute the tasks server by server. ex: it will execute all the tasks in server1 first and then all the taks in server2 and so on...

serial: 2 --> It will execute all the tasks in first 2 servers, then it will execute all the tasks in another 2 servers and so on ..

serial: 5 --> 5 servers at a time

If you want to specify in percentage ?

serial: "50%" -- it will run all the tasks in 50% of the servers and then again all the tasks in another 50% of the servers

Additional scenario:

serial: "100%" :--> same as parallel. no difference in giving this command.
Example :

```
---
- hosts: dev
  Serial: 1
  name: install git
  tasks:
     - name: install git
       yum:
            name: install git
            state: latest

     - name: install nginx
       yum:
            name: install nginx
            state: latest
```

```
---
- hosts: dev
  Serial: “20%”
  name: install git
  tasks:
     - name: install git
       yum:
            name: install git
            state: latest

     - name: install nginx
       yum:
            name: install nginx
            state: latest
```

13) write a playbook to create users in Target machines

```
---
- name: create new users
  hosts: all
  tasks:
     - name: create user
       user:
            name: user1
            state: present
            password: " {{ my pass | password_hash ('sha512', 'A512') }}
```

14) what is the use of block and rescue?

If any task fails in "server1" and if you want to execute in "server2" then we use block and rescue.
It will execute only one. Either it will execute block or rescue

Playbook:

```
---
- hosts: dev
  name: building a Java project
  tasks:
     - name: maven build
       block:
            - name: build in server1
              shell: mvn clean install
              when: " ' IP1' " in inventory_hostname
       rescue:
            - name: build in server2
              shell: mvn clean install
              when: " 'IP2' " in inventory_hostname
```

15) how to use variables in playbook?

By using **extra-vars**

Playbook:

```
- hosts: all
  name: my playbook
  tasks:
     - name: printing something
       shell: echo " This is {{ fruit }}
```

Example 1:
Command: ansible-playbook test.yml --extra-vars " fruit = apple "
Output: This is apple

Example 2:
Command: ansible-playbook test.yml --extra-vars " fruit = banana "
Output: This is banana

16) How to use modules in ad-hoc commands without playbooks?

1. We can use yum module as below
ansible all -s -m yum -a "name=httpd state=installed"

2. We can use service module as below
ansible all -s -m service -a "name=jenkins state=started"

3. To check the status of a service
ansible all -m service -a "name=httpd" -i ansible_hosts -u vagrant

17) How to write multiple commands in a single shell module

```
- hosts: all
  name: my playbook
  tasks:
     - name: printing something
       shell: pwd ; ls ; sh script.sh
```

```
- hosts: all
  name: my playbook
  tasks:
     - name: printing something
       shell: |
            pwd
            ls
            sh script.sh
```

<u>Handlers</u> ****

Sometimes you want a task to run only when a change is made on a machine. For example, you may want to restart a service if a task updates the configuration of that service, but not if the configuration is unchanged. Ansible uses handlers to address this use case. **Handlers are tasks that only run when notified.**

Here in the below example we are calling the tasks using notify. And we are changing the normal flow of ansible and controlling when the special tasks are executed

we can have handlers in Ansible roles

```yaml
tasks:
- name: copyconfiguration file
  copy:
     src: template.j2
     dest: /etc/foo.conf
  notify:
    - Restart apache
    - Restart sshd

handlers:
  - name: Restart sshd
    service:
       name: sshd
       state: restarted

  - name: Restart apache
    service:
       name: apache
       state: restarted
```

what is dynamic inventory in ansible ?

In Ansible, Dynamic inventory is generated either by scripts which are written in a programming language like python, php etc. or using available inventory plugins. When using script, they gets all real time data from the target source environments, like Cloud platforms AWS, OpenStack, GCP etc.

The inventory list will be updated by latest server Ips. We dont have to add manually.

What is ansible-tower or AWX ?

Both provides user interface (UI) to trigger ansible playbooks and helps in better management of playbooks, inventory, credentials. We can seperate the playbooks in to projects.

AWX – open source UI
Ansibe-Tower – enterprise UI