Ansible

Configuration Management (C.M):

A configuration management is refers to the process of systematically handling the changes to a system.

→ A C.M system is used to keep track of an organization hardware, software and related information.

→ with respect to IT, It covers set of things like below

@ hardware
@ S/w
@ Network
@ people
@ process

→ C.M Tools helps us to implement :

* procedures.
* policies
* Technicics
* Tools

→ Benefits of CM :-

1. Reduced Risk
2. cost reduction
3. strict control
4. Greater agility & faster problem resolution
5. Quicker restoration of service
6. Increase uptime
7. prevent Errors
8. code re-usability

*. Ansible :-

Ansible is an automation platform that configures and manages your infrastructure, whether it is "$\boxed{\text{on premises}}$" or in the cloud.

→ configuration management tool for IT professions

→ Ansible is an independent

→ you only need to tell to ansible what the desired configuration should be, not how to achieve it.

→ Tell what to do not how to do. (Ansible code)

E.g: 1. check whether the pkg is installed or not

2. Installgit 1.2

3. we need to compare whether git 1.2 is compatable to our system.

4. If not, then install git 1.3 version

5. return values

→ If you use ansible code (Eg: install git 1.2) (that too re

→ it turns your infrastructure as a code.

*. **why ansible :-**

→ No need to go with huge configuration and setup c to chef and puppet.

→ pull mechanism.(

→ lot of learning is required for chef and puppet.

*. **pros of ansible :**

→ Agentlen
→ Relies on ssh
→ it uses python internally
→ push mechanism

→*. we have to install ansible in one machine and push the configuration and all to other machines through ssh connections.

*. **Archetecture :-**

Ansible server - - - - - - - YAML - - - - - - - - - → Node

*. **Configurations :**

Ansible . config (Environment variable)
ansible . cfg (current directory),
/etc / ansible / ansible . cfg (default)

*. **Ansible Inventory :-**

Inventory is a place where all the systems present to which ansible is to configure and run.

*. **Host patterns :**

→ How to identify which machine i need to run

→ a pattern can refer to a particular machine or group name.

`ansible < host pattern> -m < module name> -a < argume`

*. **Adhoc commands :-**

→ If you want to run any simple and one tim commands we will used ad-hoc commands.

Syntax : ansible [group / host] - m [module] - a [cmd

→ Install / remove a packages :

```
ansible demo -m yum -a " pkg = httpd  state = present " install
ansible all -m yum -a " pkg = httpd  state = latest " update
ansible demo -m yum -a "pkg = httpd  state = absent " remove
```

→ start / stop a services :

```
ansible -m service -a " name = httpd  state = started"
ansible -m service -a "name = httpd  State = restarted"
ansible -m service -a "name = httpd  state = stopped"
```

→ Create / delete a user account :

```
ansible demo -m user -a " name = uma"
ansible demo -m user -a " name = uma  state = absent"
```

→ Add / Remove a cron job :-

```
ansible demo -m cron -a " name = 'name_of_cronjob' minute = 
hour = '18'  day = '4'  month = '12'  weekday = '5' ~~job~~ =
job = 'ls -alh > /dev / null "
```

*· Gathering facts / convergence / Idempotence :-

→ As soon as ansible connects to machine it gathers the information about the machine and it start comparing the state that what we have defined and what it has gathered. in case it has diff then based on that it is going to update / maintaining the state accordingly, if already in same state it is not going to do anything.

→ If we want to see what and all modules it is going to gather from machine, we can use

```
ansible all -m setup
ansible all -m setup -a " filter = 'ansible-node name' "
```

★. playbooks :-

→ playbooks are ansible configurateons, playbooks defin a policy you want your remote machine to enforce.
→ play books are retured in YAML format.

ansible - playbook < playbook - name . yml >

★. We have 3 Sections in playbook :-

1. Target section
2. Variable section
3. Task Section

★. YAML :-

→ Every YAML starts with a list
→ Each item in the list is a list of key/value pairs, commonly called a "hash" or a " directory".
→ all YAML files optionally begin with "---" end with '...'
→ All members of a list are lines begining at the same indentation levelstarting with "-".

a list of courses

```
Eg:  --- # Sample playbook
      - hosts: all
        tasks :
          - name : install ftp package
        action : yum name = ftp   state = present
```

★. Handlers :

having ability to call another task, only when the task run succesfully

```
       -name : restart vsftpd
    action service name = vstpd   state = restarted
```

★. Outline the playbook :-

→ outline to my playbook

★. Dry run ;

→ without ~~extra~~ Executing the Steps, just ~~go~~ checki what are possibilities (format, syntax ... etc)

* Asynchronous Actions and polling:-

→ while using ansible against multiple machines, the operations may run longer then usual. so, we will not have a control on it.

→ to have control whether the task is running are not and running the tasks parallel, we use async mode.

Eg: async : 300 (if it running beyond 300 sec then it will automatically get terminated).

poll : 6

→ to get the status of a parallelly running tasks we use poll option.

→ it will check and get some data on specified time given.

→ run Once :-

It will run the task only one time in every 1st machine of group, if you specify hosts : all also.

Eg: run_ once : true

→ delegate_to :

Specifing a individual host to run the task

Eg: delegate_to : localhost

→ loop :

If you want to run single task multiple times, we use loops

Eg :-
```
        # loop
          -hosts : all
      tasks :
          -name : adding list of users
      user : name = {{ item }} state = present
    -with_items
            - user 1
            - user 2
            - user 3
```

* Conditional :

→ Suppose in 1st group there are 2 users and both are using different OS do, that time redhat commands will not work on ubuntu os so we hv to ~~[illegible]~~ mention ubuntu commads specificall