# ANSIBLE

**(Push Based Methodology)**

Ansible works on push based methodology. Ansible is agentless such that it requires nothing to be installed on the target host machine except ssh connection and python.

# Difference between chef and ansible

**Chef** →

* It works with client server architecture.
* It works with ruby.
* We need to register each and every host individually to the host machine.
* Chef is more secure than ansible.
* It is difficult to set up.

# Ansible →

* It works with push based methodology. It works with python.
* It supports dynamic inventory.
* Ansible is not secure by default. We need to add an external module called vault.
* It is easy to set up.

# RAW Module

It is also called a dirty module. If there is no python in the target machine we use raw module to install python or run it by some shell commands.

# Ansible Inventory (/etc/ansible/hosts)

We have 2 types of inventory

# → Static Inventory

**→ Dynamic Inventory**

**Static Inventory:**

Static inventory is a file which contains the list of IP addresses of target host machines and groups them together based on user requirements in json or ini format. Default location of inventory is **/etc/ansible/hosts**

# Dynamic Inventory :

Ansible team will provide two file, python script and ini file. When we execute script it will automatically fetch the address of target host machines and stores it in the inventory file. Only thing we have to update is the path of inventory file in python script.

# Modules:

* **Ping Module :** It is used to check connection of target host machines. It will get reply with pong.

Syntax → *ansible ip\_address(target m/c) -m ping ansible group\_name -m ping*

*ansible -i inventory\_filename -m ping*

* **Packaging Module :** It is used to install packages. It is same as yum in centos/rhel and apt in ubuntu/debian.
* **copy module** ---> it is used to copy a file from host machine to the multiple target machines.

**Syntax**:

copy:

src: source path

dest: destination path

* how to change the permission of a file?

**sol.** Using file module

syntax:

file:

path: file path

mode: 0744

owner:

group:

# S tate Module (hydo-potential behaviour):

*state: present* → It will check for the package is already installed or not. If not

installed then only it will install the package.

*state: latest* → It will check whether the package is installed or not, if it is installed it will ensure up to date or install with the latest version.

*state: removed →* It will uninstall the package.

*state: absent →* It will uninstall the package.

# Become Module :

**Privilege Access :** If i want to run with root permissions or other user privilege permission. For this we will use the *become* module

**Syntax →** *become: yes*

*become\_user: user\_name*

* **update\_cache:** used to update repository

***update\_cache: yes***

* **template:** It isused to copy dynamic files from host machine to target hostmachine. Make sure that file has to be created in jinja format (.j2).

template:

src: source path

dest: destination path

**Ansible Ad hoc Commands:**

An Ansible ad hoc command uses the /usr/bin/ansible command-line tool to automate a single task on one or more managed nodes. ad hoc commands are great for tasks you repeat rarely. Ad hoc tasks can be used to reboot servers, copy files, manage packages and users, create directory, to check the connection of target host machines etc.

**Syntax:**

**Ansible hostname -m module\_name -a module\_options**

The -a option accepts options either through the key=value syntax or a JSON string starting with { and ending with }

**Ex:**

1. To Check the connection of Target Host machine

**ansible all -m ping**

**ansible group\_name -m ping**

**ansible host\_ipaddress -m ping**

1. To Reboot server

**ansible all -m “/sbin/reboot”**

1. To execute shell commands

**ansible all -m shell -a “wc -c /home/ubuntu/1.txt”** 🡪 to count no of characters in 1.txt file which is in target host machine

**ansible all -m shell -a “touch /home/ubuntu/1.txt” 🡪** create a file in target host machine

1. To copy a file from host machine to target host machine

**ansible all -m copy -a “src=/home/ubuntu/1.txt dest=/home/ubuntu/”**

1. To create a directory

**ansible all -m file -a “dest=/home/ubuntu/new mode=777 owner=root group=root state=directory”**

1. To delete whole directory and files

**ansible all -m file -a “dest=/home/ubuntu/new state=absent”**

1. To manage packages

**ansible all -m yum -a “name=git state=present”** 🡪 to install

**ansible all -m yum -a “name=git state=absent”**

**Ansible Playbook:**

Playbooks are the files where Ansible code is written. Playbooks are written in YAML format. YAML stands for Yet Another Markup Language

**Ex: Refer class notes for scripts**

To execute playbooks use ansible-playbook filename and to execute in debug mode ansible-playbook filename -vvv

**Ansible-vault: (**how you are going to secure ansible)

Sol: using ansible-vault

Ansible vault is a feature of ansible that allows you to keep sensitive data such as password or keys in encrypted files, rather than plaintext in playbooks or roles.

**Commands**:

* **Create**: it is used to create ansible vault file in the encrypted format.

**ansible-vault create 1.yml**

* **View:** To view the data of encrypted files.

**ansible-vault view 1.yml**

* Edit: It is used to edit the encrypted file

**ansible-vault edit 1.yml**

* Encrypt: used to encrypt the unencrypted files

**ansible-vault encrypt 1.yml**

* Decrypt: used to decrypt an encrypted files

**ansible-vault decrypt 1.yml**

* --ask-vault-pass: used to provide passwords while running playbooks

**Ansible-playbooks 1.yml –ask-vault-pass**

* --vault-password-file: used to pass a vault password through a file

Refer:

1. <https://www.youtube.com/watch?v=L8GzW4sopug>
2. <https://www.youtube.com/watch?v=XNr3BbCtSmA>

**Ansible-galaxy:** it is a command used to create and managing the roles. ansible galaxy is large public repository of ansible roles.

**ansible-galaxy list** : it dispaly a list of installed roles with version number

**ansible-galaxy remove role\_name** : it will remove an installed role

**ansible-galaxy info** : it will provide a information about ansible-galaxy

**ansible-galaxy init Role\_name** : to create a role

**Roles:** Roles splits the single playbook into multiple files. Ansible roles is defined with 8 directory and with 8 files, roles must include at least contain one of above directories.

**Directory structure of roles:**

sample

├── defaults

│ └── main.yml

├── files

├── handlers

│ └── main.yml

├── meta

│ └── main.yml

├── README.md

├── tasks

│ └── main.yml

├── templates

├── tests

│ ├── inventory

│ └── test.yml

└── vars

└── main.yml

**default:** default variable for the roles

**files:** conain files which can be deployed through roles

**handlers:** which contains handlers which may be used by this role or even anywhere outside this role

**meta:** Define some meta data for this role

**tasks:** contain the main list of tasks to be executed by the roles

**templates:** it contains templates(files) which can be deployed through this roles

**tests:** it may contain simple inventory and a test,it may be usefull if u on automated testing process which is build through roles

**vars:** variables used in the role

**Difference B/w default and vars**

Default have lowest priority value than any other variables in ansible. Vars will have the highest priority than default

**Difference B/w files and templates:**

Ansible files are used to copy static files without changing any contents from host machine to target machine.

Templates take the files from hosts machine and changes the contents of files (value of variable defined in templates) before copying. It can be done using jinja filter (.j2 format) and then it will copy a file to the target machine.

**Handlers**: it are special type of tasks which is executed only when there is change in task to which handler is defined

How will you call handlers?

Using notify

**Syntax:** **- notify: handlers name**

How will you include yml (playbooks) file within another yml (playbooks) file or how will you call yml files from another yml file.

Using include

**Syntax:** **- include: yml\_file\_name**

**Error handling:**

By default Ansible stops executing tasks on a host when a task fails on that host.

**ignore\_errors:**

You can use ignore\_errors to continue on in spite of the failure.

**ignore\_errors: true**

**Ignoring unreachable host errors:**

You can ignore a task failure due to the host instance being ‘UNREACHABLE’ with the ignore\_unreachable keyword. Ansible ignores the task errors, but continues to execute future tasks against the unreachable host. For example, at the task level:

**ignore\_unreachable: true**

**Handlers and Failures:**

Ansible runs [handlers](https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_handlers.html#handlers) at the end of each playbook. If a task notifies a handler but another task fails later in the playbook, by default the handler does not run on that host, which may leave the host in an unexpected state. For example, a task could update a configuration file and notify a handler to restart some service. If a task later in the same playbook fails, the configuration file might be changed but the service will not be restarted.

You can change this behavior with the force\_handlers: true in a playbook. When handlers are forced, Ansible will run all notified handlers on all hosts, even hosts with failed tasks.

**Defining faiures:**

Ansible lets you define what “failure” means in each task using the failed\_when conditional. If you want to trigger a failure when any of the conditions is met, you must define the conditions in a string with an explicit or operator.

**Defining changed:**

Ansible lets you define when a particular task has “changed” a remote node using the changed\_when conditional. This lets you determine, based on return codes or output, whether a change should be reported in Ansible statistics and whether a handler should be triggered or not.

**Ansible facts:**

Ansible collects pretty much all the information about the remote hosts as it runs a playbook. The task of collecting this remote system information is called as gathering facts by ansible and the details collected are generally known as facts or variables. This information can be obtained manually using ansible adhoc command and a specialized module named setup.

**ansible all -m setup**

These ansible facts would be collected for each hosts in your hostgroup before playbook execution until you disable it explicitly by defining **gather\_facts: no**