

TABLE OF CONTENT



- Adhoc Commands
- Ansible Modules
- Colour Pattern
- Ansible Playbook
- Ansible Tags
- Ansible Variables
- Static variables
- Dynamic variables
- Ansible Handlers
- Ansible Vault
- Ansible Roles
- Ansible Galaxy
- Debug Module
- Ansible LookUps
- Ansible Conditions
- Ansible Loops

What Is Ansible



- Ansible is an open-source automation tool used for configuration management, application deployment, and task automation.
- It is also called as configuration management tool.
- Invented by Micheal Dehaaan in 2012.
- Later it was taken by Redhat.
- It has both free and paid versions.
- It is platform Independent.
- Ansible works with YAML Language.

PRE-REQUISITES



- **Ansible server:** It is used to communicate with worker nodes and install packages.
- **Worker Nodes:** They will take commands from Ansible server and will work accordingly to it.
- **Playbook:** Playbook will contains the code which is used to perform actions.
- **Inventory File:** It will contains the information about the worker nodes.

ANSIBLE SETUP



ANSIBLE

- Create three servers , one is Ansible server and remaining are worker nodes.
- create ssh and share with worker nodes.
- `yum -y install ansible-core` (to install package on ansible server only)
- `vim /etc/ansible/hosts` (to make inventory list)
- `<192.168.8.101>`
- `<192.168.8.102>` (mention ip of worker nodes in hosts file)
- `:wq` (save and quit hosts file)

ADHOC COMMANDS

- `ansible all -a "yum -y install git"`
- `ansible all -a "systemctl status git"`
- `ansible all -a "systemctl start httpd"`
- `ansible all -a "systemctl enable httpd"`
- `ansible all -a "touch file1"`
- `ansible all -a "mkdir dir1"`
- `ansible all -a "useradd ali"`
- `ansible all -a "userdel sunny"`
- `ansible all -a "cat /etc/group"`
-



ANSIBLE

ANSIBLE MODULES



ANSIBLE

what is Module?

An Ansible module is a reusable, standalone script used to perform specific tasks in Ansible, such as managing files, installing packages, or configuring systems.

ANSIBLE MODULES



ANSIBLE

- `ansible all -m file -a "path=/root/file1 state=touch"`
- `ansible all -m file -a "path=/tmp/dir state=directory"`
- `ansible all -m user -a "name=ahmad state=present"`
- `ansible all -m user -a "name=ahmad state=absent"`
- `ansible all -m yum -a "name=httpd state=present"`
- `ansible all -m service -a "name=httpd state=started"`
- `ansible all -m service -a "name=httpd state=stopped"`
- `ansible all -m service -a "name=httpd enabled=yes"`
- `ansible all -m copy -a "src=/root/file1 dest=/tmp"`
- `ansible all -m lineinfile -a "path=/etc/test1 line=anyline create=yes"`
- `ansible all -m lineinfile -a "path=/etc/test1 regexp=anyline state=absent"`

ANSIBLE PLAYBOOK



ANSIBLE

- Playbook is a collection of Modules.
- In Playbook we can execute multiple commands at a same time.
- Playbook is written in YAML Language.
- YAML: Yet Another Markup Language.
- YAML is a human readable language.
- YAML is syntax based.

ANSIBLE PLAYBOOK



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: installing httpd
```

```
      yum: name=httpd state=present
```

```
    - name: starting httpd
```

```
      service: name=httpd state=started
```

```
:wq
```

```
> ansible-playbook my-playbook.yml (to run the playbook)
```

ANSIBLE PLAYBOOK



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: adding user
```

```
      user: name=sunny state=present
```

```
    - name: removing user
```

```
      user: name=sunny state=absent
```

```
:wq
```

```
> ansible-playbook my-playbook.yml
```

ANSIBLE PLAYBOOK



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: adding user
```

```
      user: name=sunny state=present
```

```
    - name: removing user
```

```
      user: name=sunny state=absent
```

```
:wq
```

```
ansible-playbook my-playbook.yml
```

ANSIBLE PLAYBOOK



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: removing httpd
```

```
      yum: name=httpd state=present
```

```
    - name: stopping httpd
```

```
      yum: name=httpd state=stopped
```

```
:wq
```

```
ansible-playbook my-playbook.yml
```

ANSIBLE PLAYBOOK



ANSIBLE

```
> vim my-playbook.yml
---
- hosts: all
  tasks:
    - name: removing user
      user: name=sunny state=absent
:wq
ansible-playbook my-playbook.yml
```

ANSIBLE TAGS



ANSIBLE

Ansible tags are used to execute a specific task or they can be used to skip a specific task.

ANSIBLE TAGS



ANSIBLE

> vim my-playbook.yml

- hosts: all

- tasks:

- name: installing docker

- yum: name=docker state=present

- tags: a

- name: installing docker

- yum: name=docker state=present

- tags: b

> ansible-playbook --tags a my-playbook.yml (only task a will execute)

> ansible-playbook --skip-tags b my-playbook.yml (task b will skip)

ANSIBLE VARIABLES



ANSIBLE

Static Variables

Static variables are those that are defined in the playbook or inventory files and do not change during playbook execution.

Dynamic Variables:

Dynamic variables in Ansible are variables whose values are determined during the execution of a playbook, based on the current environment, context, or specific conditions.

ANSIBLE VARIABLES (STATIC)



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  vars:
```

```
    a: git
```

```
    b: maven
```

```
    c: docker
```

```
    d: httpd
```

```
  tasks:
```

```
- name: installing pkg
```

```
  yum: name={{a}} state=present
```

```
- name: installing maven
```

```
  yum: name={{b}} state=present
```

```
- name: installing docker
```

```
  yum: name={{c}} state=present
```

```
- name: installing maven
```

```
  yum: name={{b}} state=present
```

```
> ansible-playbook my-playbook.yml
```

ANSIBLE VARIABLES

(DYNAMIC)



```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: adding user
```

```
      user: name={{a}} state=present
```

```
:wq
```

```
> ansible-playbook my-playbook.yml --extra-vars "a=umaira"
```

```
> ansible-playbook my-playbook.yml --extra-vars "a=sumaira"
```

ANSIBLE HANDLERS



ANSIBLE

Ansible handlers are special tasks in Ansible that are triggered by other tasks using the notify directive. Handlers are used to perform actions only when a change occurs in the tasks that notify them. This is useful for tasks that should only run when something specific has changed, like restarting a service after a configuration file has been updated.

ANSIBLE HANDLERS



ANSIBLE

```
> vim my-playbook.yml
```

```
---
```

```
- hosts: all
```

```
  tasks:
```

```
    - name: installing httpd
```

```
      yum: name=httpd state=present
```

```
      notify: installing httpd
```

```
  handlers:
```

```
    - name: starting httpd
```

```
      service: name=httpd state=started
```

```
:wq
```

```
> ansible-playbook my-playbook.yml
```

ANSIBLE LOOPS



ANSIBLE

Ansible loops allow you to repeat a task multiple times with different inputs. This is useful when you want to perform the same action on multiple items without having to write the task multiple times.

ANSIBLE LOOPS



ANSIBLE

```
---  
- hosts: all  
  tasks:  
    - name: adding users  
      user: name={{items}} state=present  
      with_items:  
        - tahira  
        - sumaira  
        - umaira  
        - saba  
        - sara  
  
:wq  
> ansible all my-playbook.yml
```

ANSIBLE DEBUG MODULE



ANSIBLE

The debug module in Ansible is used to display information, variables, or messages during the execution of a playbook. It is a useful tool for troubleshooting, verifying data, and providing context or feedback within a playbook.

ANSIBLE DEBUG MODULE



ANSIBLE

- hosts: all

- tasks:

- name: printing worker node information

- debug:

- msg: "The worker node name is: {{ansible_hostname}} , The total
memory is: {{ansible_memtotal_mb}} , Free memory is:
{{ansible_memfree_mb}} , The flavour is: {{ansible_os_family}} , Total no. of
cpu's: {{ansible_processor_vcpus}}"

- :wq

- > ansible all my-playbook.yml

ANSIBLE VAULT



ANSIBLE

Ansible Vault is a feature within Ansible that allows you to securely store and manage sensitive data, such as passwords, API keys, and other confidential information, by encrypting files and variables. This ensures that sensitive information is not exposed in plain text within your playbooks or version control systems.

ANSIBLE VAULT



ANSIBLE

- > `ansible-vault create my-playbook.yml` (to create encrypted playbook)
- > `username: tahira`
- > `password: password123`
- > `ansible-vault edit my-playbook.yml` (to edit vault file)
- > `ansible-vault rekey my-playbook.yml` (to change the password of vault file)
- > `ansible-vault decrypt my-playbook.yml` (to decrypt vault file)

ANSIBLE CONDITIONS



ANSIBLE

Ansible conditions allow you to control the execution of tasks based on specific criteria. By using conditions, you can make your playbooks more dynamic and flexible, ensuring that tasks are only executed when certain conditions are met.

ANSIBLE CONDITIONS



ANSIBLE

- ```

- hosts: all
 tasks:
 - name: installing git on redhat
 yum: name=git state=present
 when: ansible_os_family == "RedHat"

 - name: installing git on ubuntu
 apt: name=git state=present
 when: ansible_os_family == "Debian"
```

:wq

> ansible all my-playbook.yml

# ANSIBLE GALAXY



ANSIBLE

Ansible Galaxy is a community hub for discovering, sharing, and downloading Ansible roles and collections. It serves as a central repository where users can find reusable Ansible content created by others, as well as share their own.

# ANSIBLE GALAXY



ANSIBLE

- > ansible-galaxy (search on goggle)
- > ansible-galaxy search httpd (to search package from ansible galaxy)
- > ansible-galaxy install username-tomcat (to install role)
- > ansible-galaxy search --author <username> (to see all role by a specific user)

# ANSIBLE ROLES



ANSIBLE

- Ansible roles are used to divide the playbook into directory structure.
- We can organize the playbook through roles.
- We can reduce the length of playbook using roles.

# ANSIBLE ROLES



ANSIBLE

```
> mkdir -p roles/one/tasks
> vim roles/one/tasks/main.yml
- name: adding user
 user: name=David state=present
:wq
> vim my-playbook.yml
- hosts: all
 roles:
 - one
:wq
> vim roles/two/tasks/main.yml
- name: installing pkg
 yum: name=httpd state=present
:wq
> vim my-playbook.yml
- hosts:
 roles:
 - one
 - two
```



# ANSIBLE LOOKUPS



ANSIBLE

Ansible Lookups are a feature that allows you to retrieve data from external sources during the execution of a playbook. They are used to fetch information from files, databases, APIs, or other sources and make it available within your playbooks.

# ANSIBLE LOOKUPS



ANSIBLE

```
> vim file.txt
Hello, My name is Tahira.
> vim my-playbook.yml

- hosts: all
 vars:
 a: "{{ lookup ('file' , '/root/file.txt') }}"
 tasks:
 - debug:
 msg: "welcom all {{a}}"
:wq
> vim my-playbook.yml
```