**\*\*\*\* Ansible \*\*\*\***

Ansible is an open-source automation tool used for configuration management,

application deployment, and orchestration.

Ansible uses SSH for communication with target systems, making it agentless and easy to set up.

Inventory: The inventory file lists the target hosts on which Ansible will run

tasks. It can be a static file or generated dynamically.

Playbooks: Playbooks are YAML files that define a set of tasks and

configurations to be executed on target hosts.

Tasks: Tasks are the individual units of work in Ansible. They represent actions

to be performed on target hosts.

Modules: Ansible provides a wide range of modules for various tasks, such as

package installation, file manipulation, service management, etc.

Roles: Roles are reusable units of playbooks. They encapsulate related tasks,

handlers, variables, and files into a directory structure.

**To Install Ansible on our Ubuntu server we need to install some additional packages on server**

sudo apt-addrepository ppa:ansible/ansible

sudo apt update

sudo apt install ansible

Here's a brief overview of how the hosts file works in Ansible:

1. \*\*Location of the Hosts File:\*\*

- By default, the hosts file is located at `/etc/ansible/hosts` on the Ansible control node. However, you can specify a different location using the `-i` option when running Ansible commands.

2. \*\*Format of the Hosts File:\*\*

- The hosts file is a plain text file that contains a list of server hostnames or IP addresses. It can also include groupings to organize your servers logically.

Example of a basic hosts file:

```

[web\_servers]

web1.example.com

web2.example.com

[db\_servers]

db1.example.com

db2.example.com

```

In this example, there are two groups: `web\_servers` and `db\_servers`. Each group contains a list of server hostnames.

3. \*\*Groups and Group Variables:\*\*

- Groups allow you to organize your servers based on their roles, environments, or any other criteria. You can also define group-specific variables in separate YAML files.

Example of a hosts file with groups and variables:

```

[web\_servers]

web1.example.com ansible\_ssh\_user=ubuntu

[db\_servers]

db1.example.com ansible\_ssh\_user=postgres

```

In this example, we've also defined a specific `ansible\_ssh\_user` variable for each server.

4. \*\*Specifying Additional Information:\*\*

- You can include additional information about your servers in the hosts file, such as connection details, SSH keys, and more.

Example:

```

[web\_servers]

web1.example.com ansible\_ssh\_user=ubuntu ansible\_ssh\_private\_key\_file=/path/to/your/private\_key.pem

```

5. \*\*Dynamic Inventories (Optional):\*\*

- While the hosts file is a static way to define your inventory, Ansible also supports dynamic inventories that can automatically discover and group hosts based on various sources (e.g., cloud providers, external databases, etc.).

Example of a dynamic inventory source (AWS):

```

plugin: aws\_ec2

regions:

- us-west-2

```

6. \*\*Using Hosts in Playbooks:\*\*

- When you run Ansible playbooks, you can specify which group or individual hosts to target. This can be done using patterns or by explicitly specifying hosts.

Example playbook targeting the `web\_servers` group:

```yaml

---

- name: Example Playbook

hosts: web\_servers

tasks:

- name: Example Task

debug:

msg: "Hello from {{ inventory\_hostname }}"

```

In summary, the hosts file is a crucial component in Ansible that allows you to define the servers you want to manage. It's where you organize and categorize your servers for effective management with Ansible playbooks.

sudo nano /etc/ansible/hosts

ansible-inventory --list -y

[servers]

server1 ansible\_host=<Public IP>

server2 ansible\_host=<Public IP>

server3 ansible\_host=<Public IP>

[all:vars]

ansible\_python\_interpreter=/usr/bin/python3

ansible\_user=ubuntu

ansible\_ssh\_private\_key\_file=/<file>

Examples:

-

name: Date playbook

hosts: servers

tasks:

- name: this will show the

date

command: date

To install Nginx:

-

name: This playbook will

install nginx

hosts: servers

become: yes

tasks:

- name: install nginx

apt:

name: nginx

state: latest

- name: start nginx

service:

name: nginx

state: started

enabled: yes

Conditional Example:

-

name: this will install based on os

hosts: servers

become: yes

tasks:

- name: install Docker

apt:

name: docker.io

state: latest

- name: install aws cli

apt:

name: awscli

state: latest

when: ansible\_distribution == 'Debian'

or ansible\_distribution == 'Ubuntu'