Table of Contents

==

1.0: Basic Info:

Before devops as a system admin:

- Hard for a single person to do configuration on a large number of servers.
- If there are 500 servers then you have to hire higher staff to manage the system administration
- Almost all tasks were manually based

After devops (all work can managed by devops engg.) :

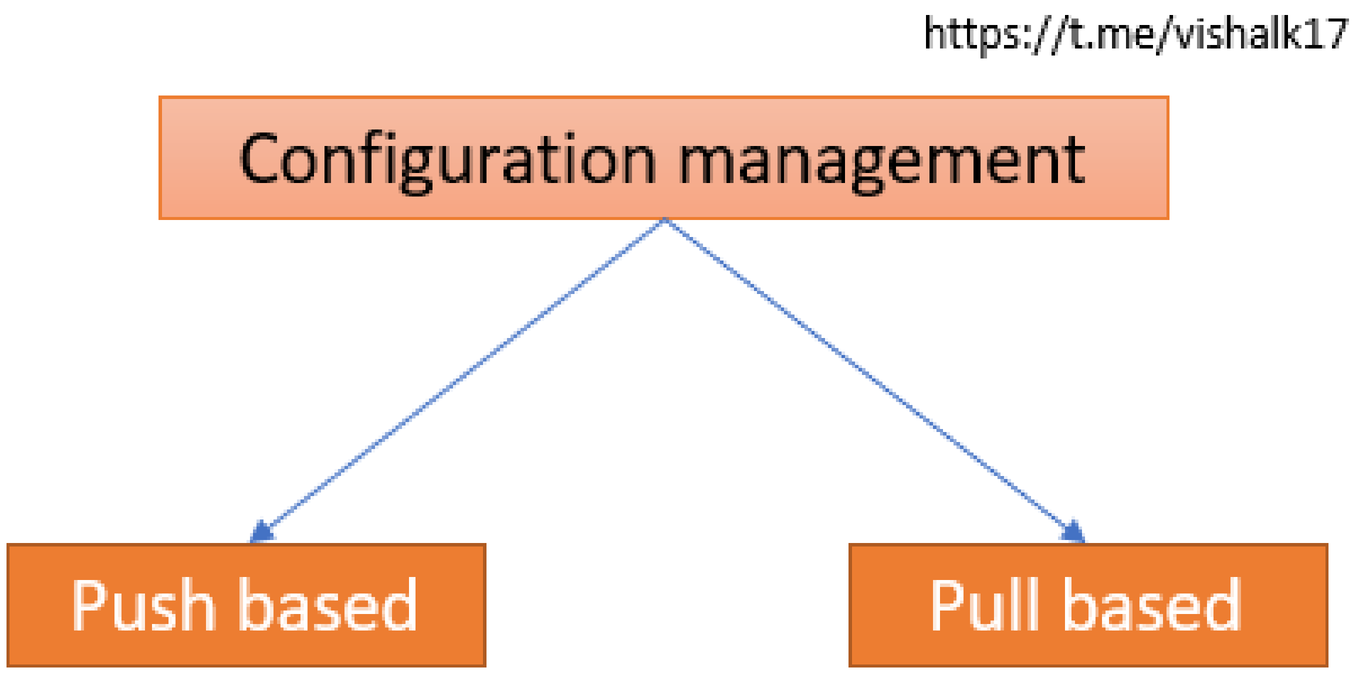
- Single server/person can handle number of servers
- Involves automation
- Infrastructure as a code
- Number of configuration management handle with the code

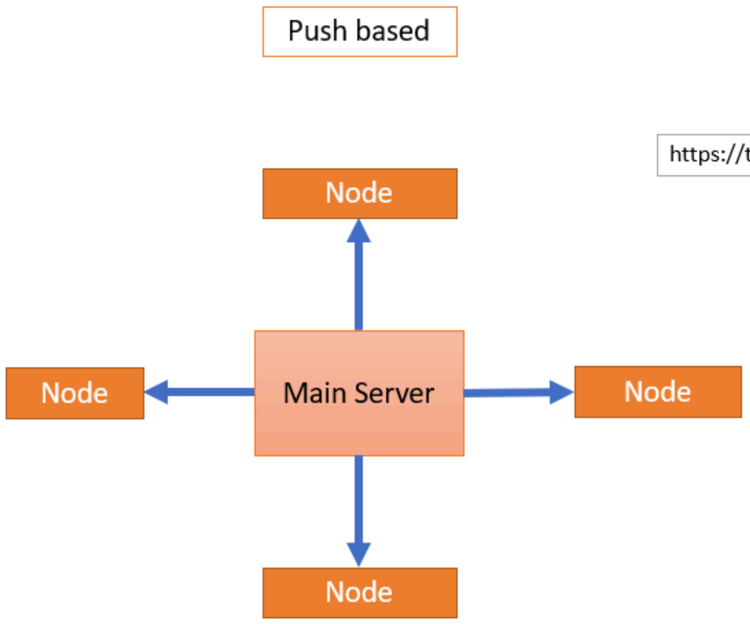
1.1: What is configuration management?

Configuration management is a process for maintaining computer systems, servers, and software in a desired, consistent state.

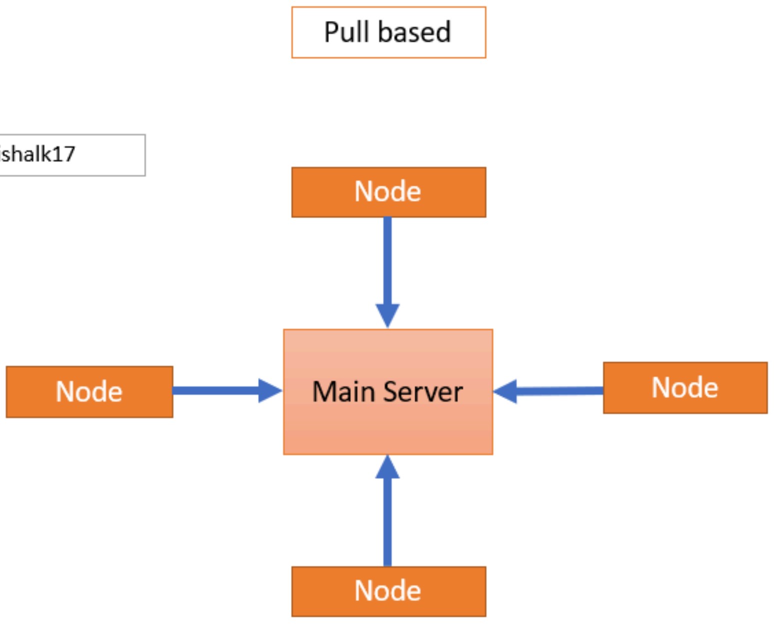
- Classify and manage systems by groups and subgroups.
- Centrally modify base configurations.
- Roll out new settings to all applicable systems.
- Automate system identification, patches, and updates
- Identify outdated, poor performing, and noncompliant configurations.
- Prioritize actions.
- Access and apply prescriptive remediation.

1.2: Types of configuration management?





https://t.me/vishalk17



Push based

- Centralized server pushes configs on the nodes.
- Server contacts nodes and send updates as needed.
- Since main server manages all things thus has overall control over nodes.
- Easier to use example ansible.

Pull based

- Well Suited but difficult to manage.
- Nodes runs an agent that ask central server if/when it has updates to run.
- Agent required to be install on nodes
- Example chef, puppet

2.0: Ansible

Ansible is a radically simple IT automation system. It handles configuration management, application deployment (CI / CD), cloud provisioning, ad-hoc task execution, network automation, and multi-node orchestration.

2.1: Why Do We Need Ansible?

Well before I tell you what is Ansible, it is of utmost importance to understand the problems that were faced before Ansible.

Let us take a little flashback to the beginning of networked computing when deploying and managing servers reliably and efficiently has been a challenge. Previously, system administrators managed servers by hand, installing software, changing configurations, and administering services on individual servers.
As data centers grew, and hosted applications became more complex, administrators realized they couldn't scale their manual systems management as fast as the applications they were enabling. It also hampered the velocity of the work of the developers since the development team was agile and released software frequently, but IT operations were spending more time configuring the systems. That's why server provisioning and configuration management tools came to flourish.

Consider the tedious routine of administering a server fleet. We always need to keep updating, pushing changes, copying files on them etc. These tasks make things very complicated and time consuming.

But let me tell you that there is a solution to the above stated problem. The solution is – ***Ansible*.**

Advantages :

- Ansible is free for everyone
- Very light weight.
- Very secure
- It directly communicates with node using ssh.
- Agentless
- Push mechanism
- Doesn't need any special system admin with skills to install and use it

- Ansible use YAML (yet another markup language

Disadvantages:

- Ansible is new therefore less support and less doc. available
- Cannot achieve full automation

2.2: Terms in Ansible:

- Ansible server : Then machine where ansible is installed and from which all task and playbook will run
- Module : Basically, a module is a command or set of similar commands meant to be executed on the client side
- Task: A task is a section that consist of a single procedure to be completed.
- Role: A way of organizing tasks and related files to be later called in a playbook
- Fact: Information fetch from the client system form the global variables with the gather-facts operations.
- Inventory: File containg data about the ansible client servers
- Play : execution of playbook.
- Handler: Task which is called only if a notifier is present
- Notifier: Section attributed to a task which calls a handler if the output is changed.
- Playbooks: It consist code in YAML format, which describes task to be executed.

3.0: Installation of ansible:

OS I'm using : Amz Linux 2023
Other Info : (ansible version I'm currently on :)

```
[root@vishalk17-node1 ~]# ansible --version
ansible [core 2.15.3]
  config file = None
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.9/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.9.16 (main, Apr 24 2024, 00:00:00) [GCC 11.4.1 20230605 (Red Hat 11.4.1-2)]
(/usr/bin/python3.9)
  jinja version = 3.1.4
  libyaml = True
```

3.1: Installation on all nodes except ansible Master Node.

I made a script for a simple installation. Execute below script :

https://github.com/vishalk17/devops/blob/main/ansible/general_install_amz_linux2023.sh

```
#!/bin/bash

# Function to display usage information
usage() {
    echo "Usage: $0 {master|node}"
    echo "example : $0 master   ... for master node"
    exit 1
}

# Check if the script is run with root privileges
if [[ $EUID -ne 0 ]]; then
    echo "This script must be run as root. Exiting..."
    exit 1
fi

# Check if a valid argument is provided
if [[ $# -ne 1 ]]; then
    usage
fi

# Determine the installation type based on the argument
INSTALL_TYPE=$1

# Function for common installation steps
common_installation() {
    ### Add user with password start ###
    echo "Adding ansible user with specified password..."
    useradd -m ansible
    echo "ansible:ansible" | chpasswd

    ### Provide ansible user with root privileges start ###
    echo "Granting sudo privileges to ansible user..."
    echo 'ansible ALL=(ALL)       NOPASSWD: ALL' >> /etc/sudoers

    ### Edit SSH configuration start ###
    echo "Modifying SSH configuration..."
    sed -i "s%#PermitRootLogin prohibit-password%PermitRootLogin yes%g" /etc/ssh/sshd_config
    sed -i "s%PasswordAuthentication no%PasswordAuthentication yes%g" /etc/ssh/sshd_config

    echo "Restarting SSH service..."
    service sshd restart
}

# Function for Ansible master installation
master_installation() {
    common_installation

    ## Install required packages ##  make sure python installed on all nodes and control plane
    echo "Installing required packages..."
    yum install -y ansible
    sleep 4

    ### Configure Ansible start ###
    echo "Configuring Ansible..."
    ansible-config init --disabled -t all > ~/.ansible/ansible.cfg  # Create ansible.cfg
    sed -i "s%;inventory=/etc/ansible/hosts%inventory=/etc/ansible/hosts%g" ~/.ansible/ansible.cfg
    sed -i "s%;become_user=root%become_user=root%g" ~/.ansible/ansible.cfg
}

# Function for Ansible node installation
node_installation() {
    common_installation
}

# Execute the appropriate function based on the argument
case $INSTALL_TYPE in
```

```
    master)
        master_installation
        ;;
    node)
        node_installation
        ;;
    *)
        usage
        ;;
esac
```

Download above script on all nodes:

Execute following command for installation on all nodes except ansible master node:

```
[root@vishalk17-node3 ~]# ls
general_install_amz_linux2023.sh

[root@vishalk17-node3 ~]# bash general_install_amz_linux2023.sh
Usage: general_install_amz_linux2023.sh {master|node}
example : general_install_amz_linux2023.sh master   ... for master node

[root@vishalk17-node3 ~]# bash general_install_amz_linux2023.sh node
```

3.2: Installation on Ansible Master Node:

Download Same Script as discussed Previously, & Execute following command

Execute following command for installation on all nodes except ansible master node:

```
[root@vishalk17-node1 ~]# ls
general_install_amz_linux2023.sh

[root@vishalk17-node1 ~]# bash general_install_amz_linux2023.sh
Usage: general_install_amz_linux2023.sh {master|node}
example : general_install_amz_linux2023.sh master   ... for master node

[root@vishalk17-node1 ~]# bash general_install_amz_linux2023.sh master
```

4.0 Configure Ansible Master Node:

We will See following things

- Create Host file
- Generate SSH keys on the Ansible Master node
- Copy the public SSH key to all remote hosts one by one

4.1 : Create Host file for our Ansible Server(Master)-Node setup with groups

Mention all the servers ip in `/etc/ansible/hosts` in Group like format

```
[ansible@vishalk17-node1 ~]$ cat /etc/ansible/hosts
[webservers]
web1 ansible_host=172.31.44.163
web2 ansible_host=172.31.37.118
web3 ansible_host=172.31.37.235

[dbservers]
db1 ansible_host=172.31.37.118
db2 ansible_host=172.31.37.235

[dev]
172.31.44.163
172.31.37.118
172.31.37.235
```

4.2: Generate SSH keys on the Ansible Master node

Change user to ansible on master node

```
[ec2-user@vishalk17-node1 ~]$ sudo su ansible
[ansible@vishalk17-node1]$
```

On the Ansible Master node , generate a pair of SSH keys using the following command:

```
[ansible@vishalk17-node1]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): ansible-master
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ansible-master
Your public key has been saved in ansible-master.pub
The key fingerprint is:
SHA256:F7UcXM3BtEZff0bhqxQn0WeRgFM1ROdwWCdMNKjSe/Q root@vishalk17-node1
The key's randomart image is:
+---[RSA 3072]----+
|            .=X//%|
|            =oo*@&|
|          ...+o *B|
|         . o.. =.o|
|        S..o o . |
```

4.3: Copy the public SSH key to all remote hosts one by one

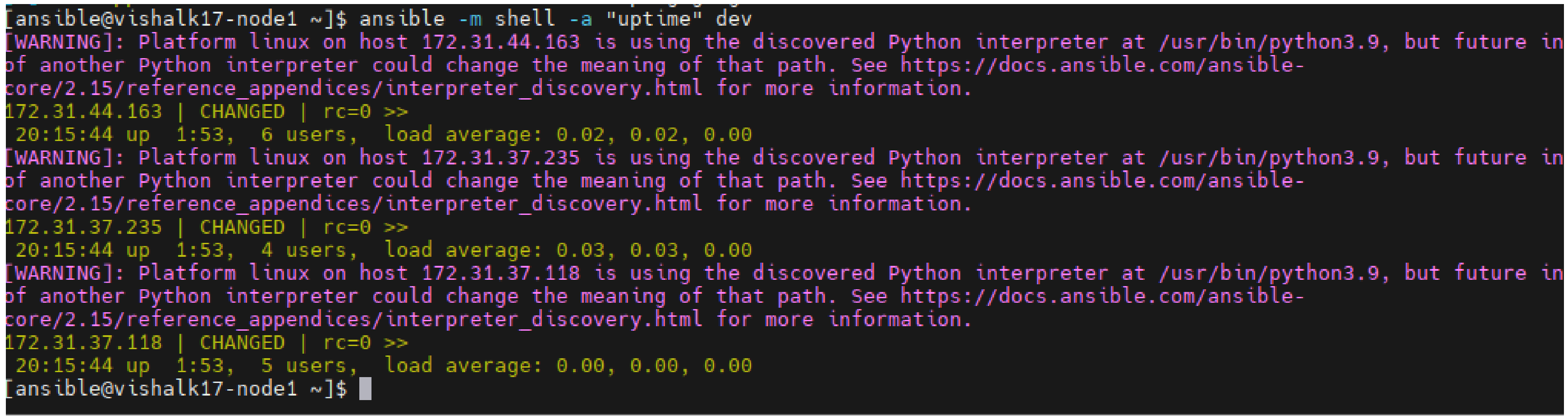
Copy the public SSH key (`~/.ssh/id_rsa.pub`) to all remote hosts, You can use the `ssh-copy-id` command to simplify this step:

ssh-copy-id -i ~/.ssh/id_rsa.pub user@remote_host

```
[ansible@vishalk17-node1 ~]$ ssh-copy-id -i ~/.ssh/id_rsa.pub ansible@172.31.44.163
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ansible/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
ansible@172.31.44.163's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'ansible@172.31.44.163'"
and check to make sure that only the key(s) you wanted were added.
```

Verify :

It should return output from servers:

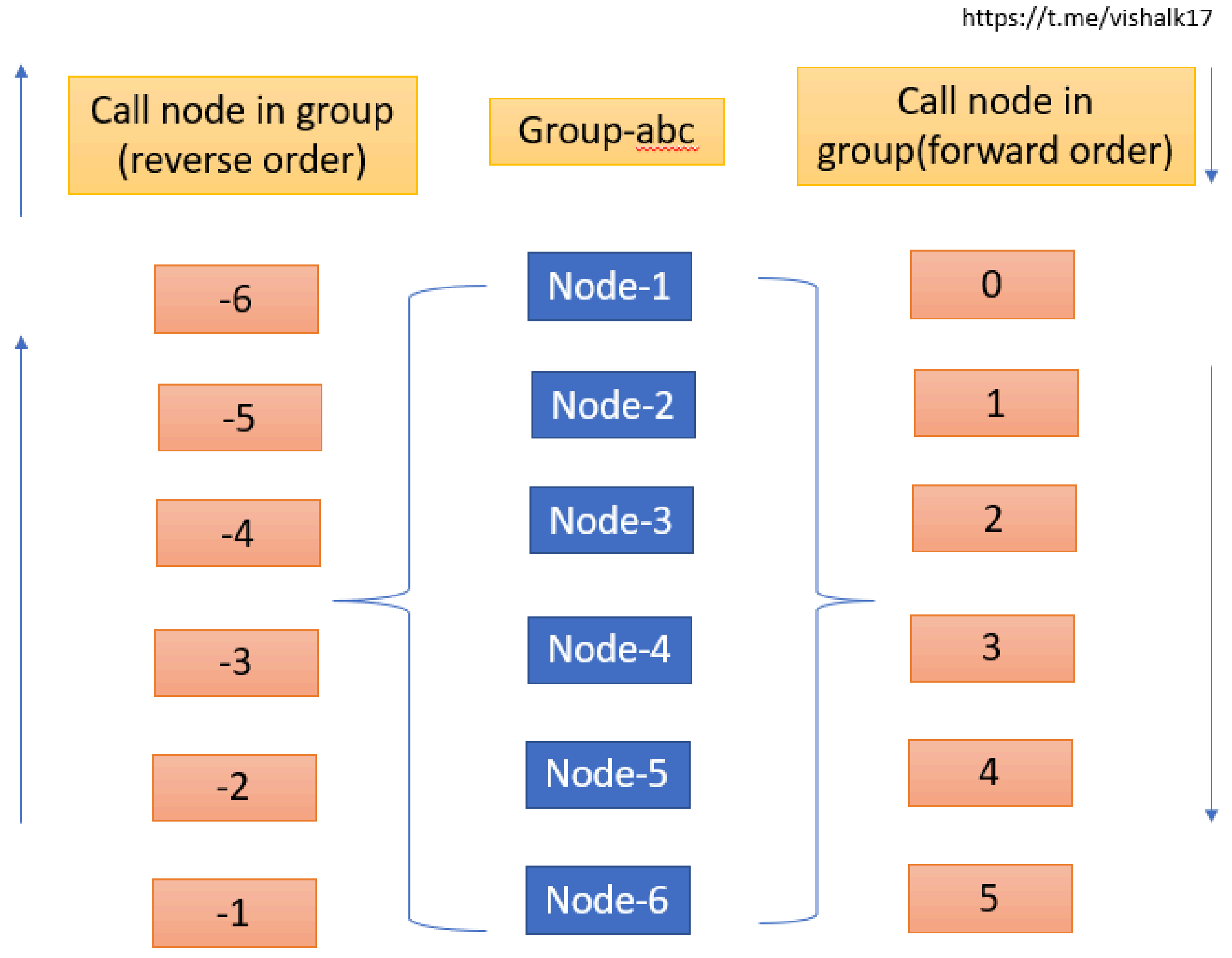


```
[ansible@vishalk17-node1 ~]$ ansible -m shell -a "uptime" dev
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future in
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.44.163 | CHANGED | rc=0 >>
 20:15:44 up  1:53,  6 users,  load average: 0.02, 0.02, 0.00
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future in
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.235 | CHANGED | rc=0 >>
 20:15:44 up  1:53,  4 users,  load average: 0.03, 0.03, 0.00
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future in
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.118 | CHANGED | rc=0 >>
 20:15:44 up  1:53,  5 users,  load average: 0.00, 0.00, 0.00
[ansible@vishalk17-node1 ~]$
```

==

5.0: Ansible Commands:

Host pattern:



Basic Patterns:

ansible all --list-hosts … all: Refers to all hosts in the inventory file.

ansible webservers --list-hosts … <group-name>: Refers to all hosts in a specific group.

Indexed Patterns:

ansible webservers[-1] --list-hosts … <group-name>[-1]: Refers to the last host in the group. (Reverse order)

ansible webservers[0] --list-hosts … <group-name>[0]: Refers to the first host in the group. (Forward order)

ansible webservers[0:1] --list-hosts … <group-name>[0:1]: Refers to the first and second hosts in the group. (Forward order)

ansible webservers[2:5] --list-hosts … <group-name>[2:5]: Refers to the third, fourth, fifth, and sixth hosts in the group. (Forward order)

Combining Patterns:

ansible webservers:dbservers --list-hosts … <group-name-1>:<group-name-2>: Refers to hosts in both groups.

ansible webservers[0:2]:dbservers[4:7] --list-hosts … <group-name-1>[0:2]:<group-name-2>[4:7]: Refers to the first, second, and third hosts in group-name-1 and the fifth, sixth, seventh, and eighth hosts in group-name-2.

Wildcard Patterns:

ansible 'web*' --list-hosts … web*: Refers to hosts with names starting with "web".

Verify :

Make sure you are executing all the commands as the ansible user on your Ansible master node.

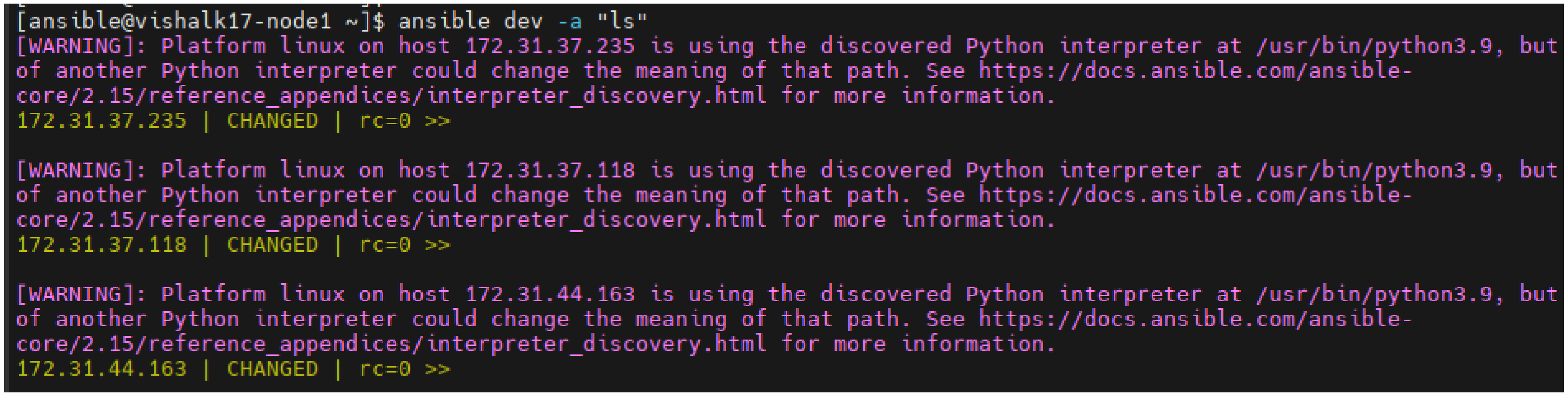
`vishalk17-node1  : It’s my Ansible Master Node`

```
[ansible@vishalk17-node1 ~]$ ansible web* --list-hosts
  hosts (3):
    web1
    web2
    web3
[ansible@vishalk17-node1 ~]$ ansible dev[0:1] --list-hosts
  hosts (2):
    172.31.44.163
    172.31.37.118
[ansible@vishalk17-node1 ~]$ ansible webservers[0:2]:dbservers[0] --list-hosts
  hosts (4):
    web1
    web2
    web3
    db1
[ansible@vishalk17-node1 ~]$ ansible dev[-1] --list-hosts
  hosts (1):
    172.31.37.235
```

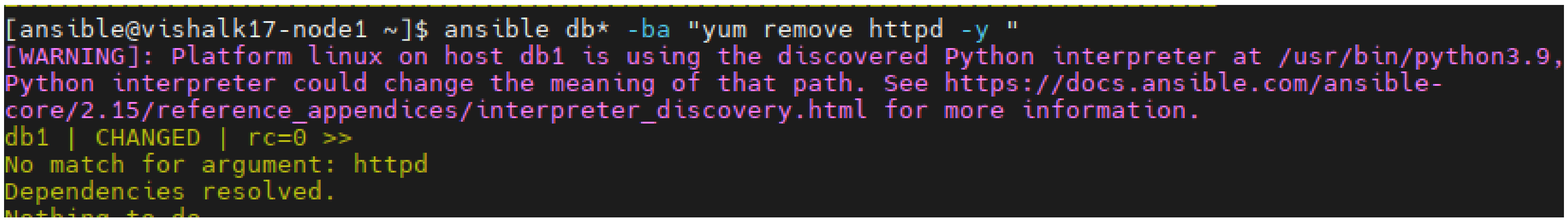
5.1: Ad-hoc commands (temporary)

- These commands can be run individually to perform quick functions.
- There is no idempotency (i.e., the same command can be used multiple times).
- These commands are not used for configuration management and deployment because they are intended for one-time use.
- Ansible ad-hoc commands use the /usr/bin/ansible command-line tool to automate a single task.
- You can use Linux commands as they are.

```
ansible dev -a "ls"
ansible all -a "touch vish"
ansible dev -a "sudo yum install httpd -y"
ansible dev -ba "yum remove httpd -y "                ... -b = become root user
```



```
[ansible@vishalk17-node1 ~]$ ansible dev -a "ls"
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.235 | CHANGED | rc=0 >>

[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.118 | CHANGED | rc=0 >>

[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.44.163 | CHANGED | rc=0 >>
```



```
[ansible@vishalk17-node1 ~]$ ansible db* -ba "yum remove httpd -y "
[WARNING]: Platform linux on host db1 is using the discovered Python interpreter at /usr/bin/python3.9,
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
db1 | CHANGED | rc=0 >>
No match for argument: httpd
Dependencies resolved.
```

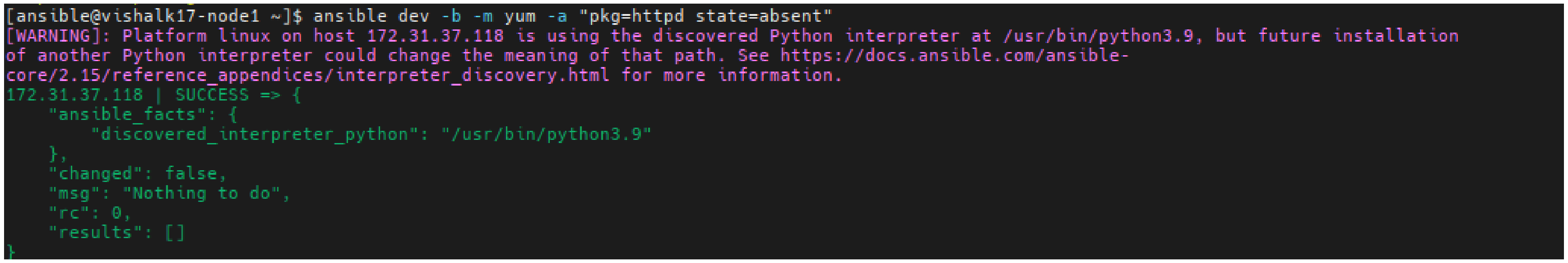
5.2: Module

- Ansible ships with a variety of modules that can be executed directly on remote hosts or through playbooks.
- Your library of modules can reside on any machine; no separate server or database is required.
- Idempotency is inherent in Ansible modules.

5.2.1 : Package (pkg) module

- Install: `present`
- Uninstall / Remove: `absent`
- Update: `latest`

```
ansible dev -b -m yum -a "pkg=httpd state=present"

where,

-b = sudo privilege
-m = module
-a = execute whatever in inverted comma's
Yum = yum module
dev = group name mentioned in /etc/ansible/hosts

ansible dev -b -m yum -a "pkg=httpd state=absent"
ansible dev -b -m yum -a "pkg=httpd state=latest"
```



```
[ansible@vishalk17-node1 ~]$ ansible dev -b -m yum -a "pkg=httpd state=absent"
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.118 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.9"
    },
    "changed": false,
    "msg": "Nothing to do",
    "rc": 0,
    "results": []
}
```

5.2.2: Other Modules:

Service module:

```
ansible dev -b -m service -a "name=httpd state=started"
```

Create user :

```
ansible dev -m user -ba "name=vishalk17 state=absent"
```

Copy :

```
ansible dev -m copy -ba "src=file1.txt dest=/mnt/"
```

Setup :

```
ansible dev -m setup
ansible dev -m setup -a "filter=*ipv4*"
```

Whenever we execute any Ansible command, Ansible first checks the current status of the target system or resource. It then proceeds to execute the specified modules based on the desired tasks or configurations defined in the playbook or ad-hoc command.

6.0 : Playbook

Written in YAML format, a playbook is a file where we define tasks, variables, handlers, files, templates, and roles to automate infrastructure configuration and management.

In Ansible playbooks, the `---` at the start is used to indicate the beginning of a YAML document.

- **Structure:**
    - **Target Section:** Defines hosts against which playbook tasks are executed.
    - **Variable Section:** Defines variables used throughout the playbook.
    - **Task Section:** Lists modules (tasks) executed sequentially.
    - **Handlers:** Tasks triggered only if notified and if a change occurs (e.g., service restart).

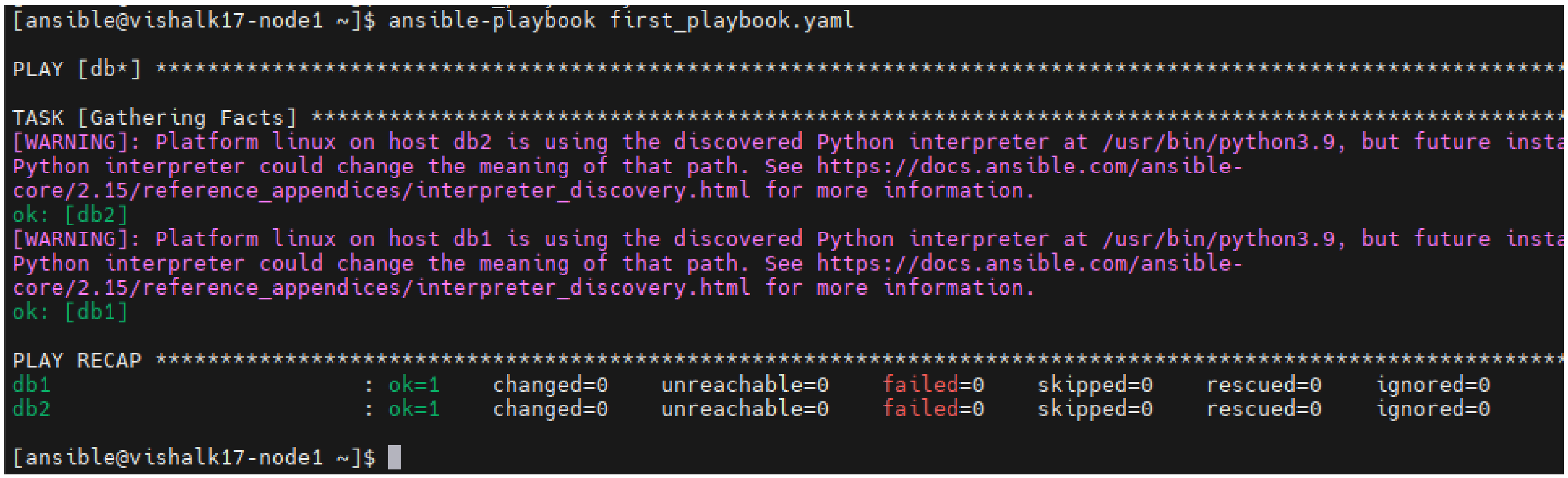
6.1: Task 1

vi first_playbook.yaml

```
--- # mytarget playbook
- hosts: db*
  user: ansible
  become: yes
  connection: ssh
  gather_facts: yes
```

- Save and Exit

Execute Playbook:

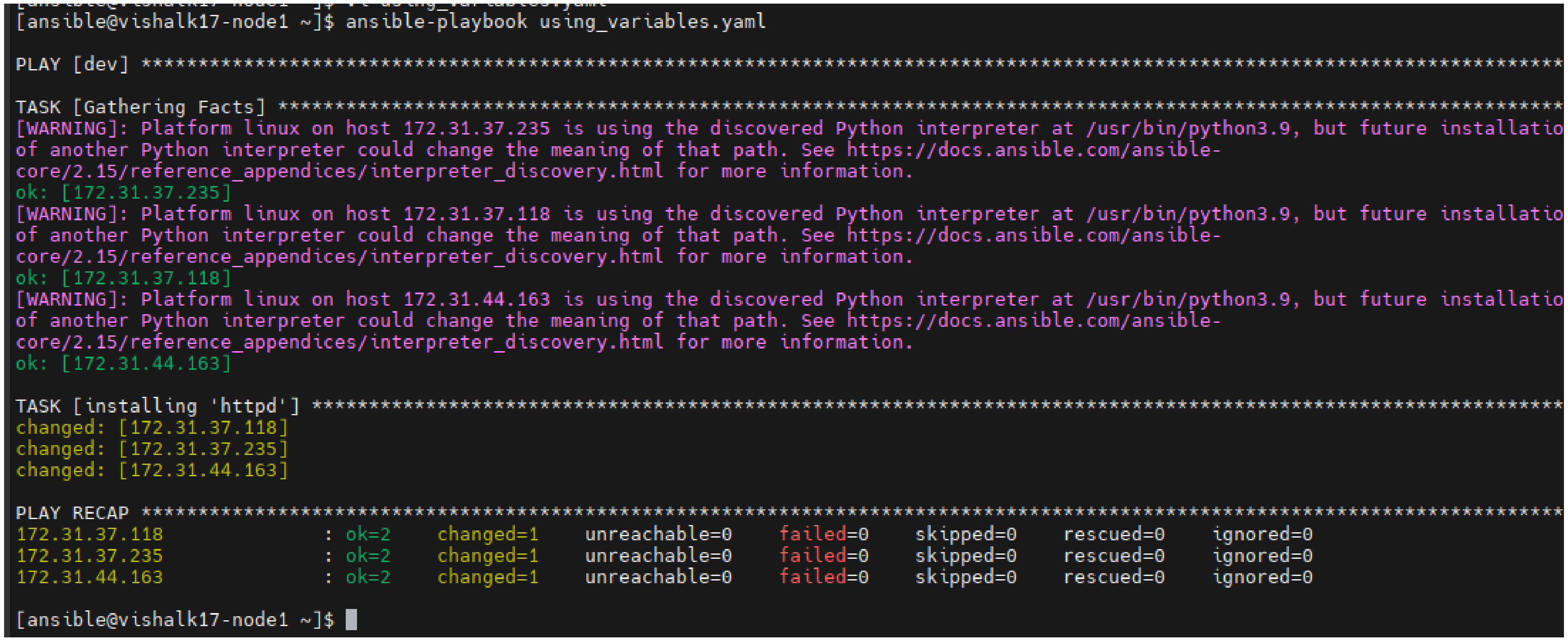


```
[ansible@vishalk17-node1 ~]$ ansible-playbook first_playbook.yaml

PLAY [db*] *************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************
[WARNING]: Platform linux on host db2 is using the discovered Python interpreter at /usr/bin/python3.9, but future insta
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [db2]
[WARNING]: Platform linux on host db1 is using the discovered Python interpreter at /usr/bin/python3.9, but future insta
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [db1]

PLAY RECAP *************************************************************************************************************
db1                        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
db2                        : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ansible@vishalk17-node1 ~]$
```

6.2: Task 2: Use Variables

vi using_variables.yaml

```
---
- hosts: dev
  user: ansible
  connection: ssh
  become: yes #root permission

  vars: #variables
    pkgn: httpd

  tasks:
    - name: installing '{{pkgn}}'
      yum: name='{{pkgn}}' state=present
```



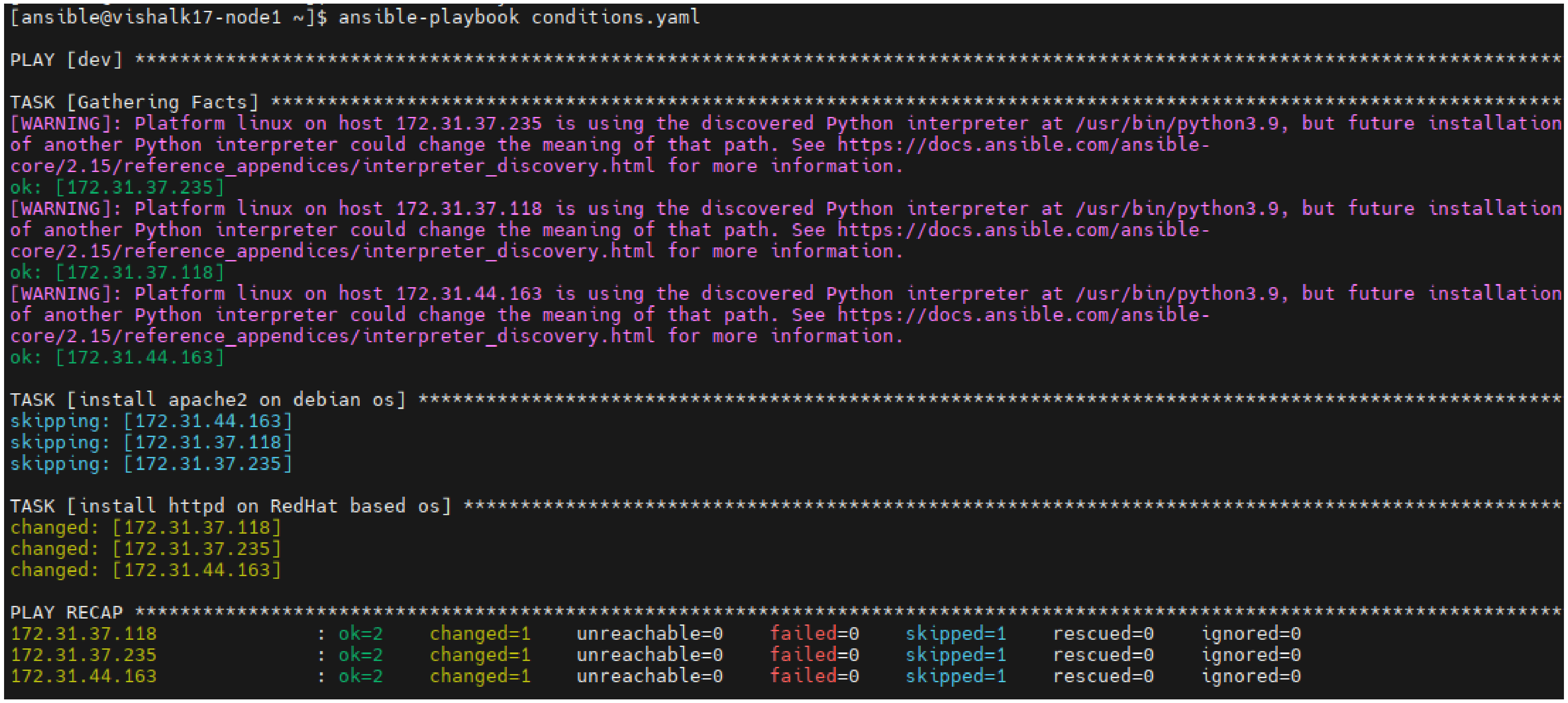
```
[ansible@vishalk17-node1 ~]$ vi using_variables.yaml
[ansible@vishalk17-node1 ~]$ ansible-playbook using_variables.yaml

PLAY [dev] *************************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************************
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future installatio
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.235]
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future installatio
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.118]
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future installatio
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.44.163]

TASK [installing 'httpd'] **********************************************************************************************************
changed: [172.31.37.118]
changed: [172.31.37.235]
changed: [172.31.44.163]

PLAY RECAP *************************************************************************************************************************
172.31.37.118              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.37.235              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.44.163              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[ansible@vishalk17-node1 ~]$
```

6.3: Task 3: Using Conditions

vi conditions.yml

```
--- # condition playbook
- hosts: dev
  user: ansible
  become: yes
  connection: ssh

  tasks:
    - name: install apache2 on debian os
```

```
      command: apt-get install apache2 -y
      when: ansible_os_family == "Debian"   # install when Debian distribution found

    - name: install httpd on RedHat based os
      command: yum install httpd -y
      when: ansible_os_family == "RedHat"
```



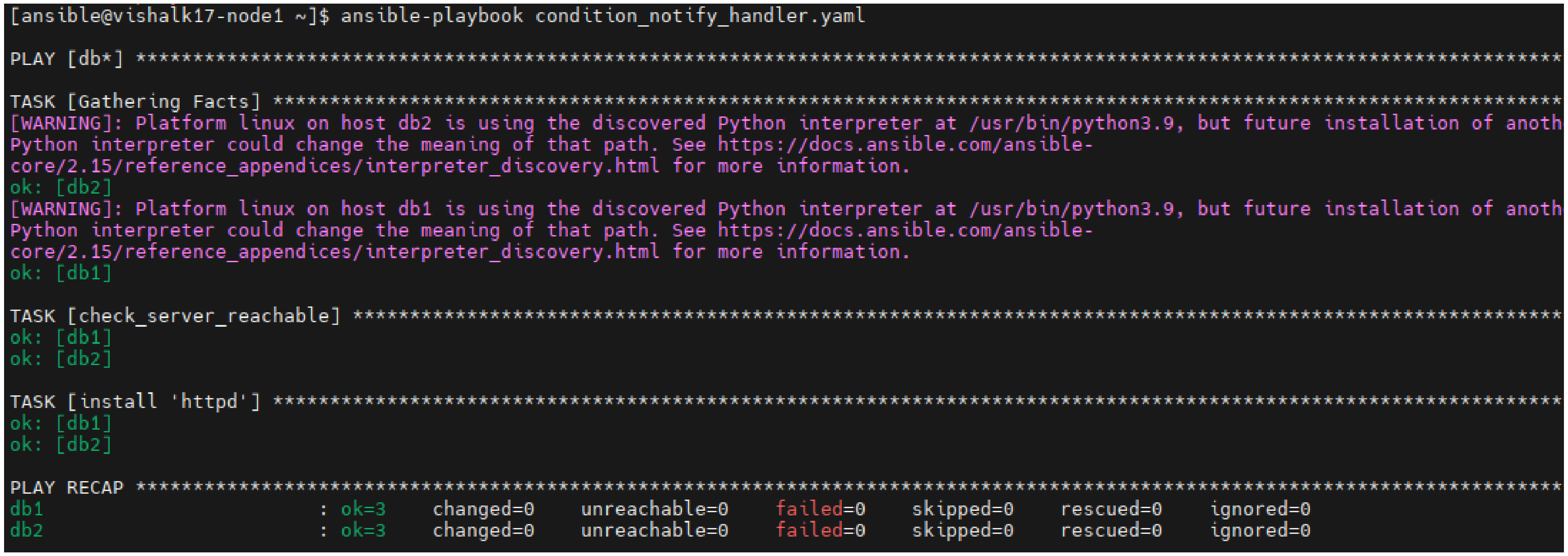
```
[ansible@vishalk17-node1 ~]$ ansible-playbook conditions.yaml

PLAY [dev] ************************************************************************************************************************

TASK [Gathering Facts] ************************************************************************************************************
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.235]
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.118]
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.44.163]

TASK [install apache2 on debian os] ***********************************************************************************************
skipping: [172.31.44.163]
skipping: [172.31.37.118]
skipping: [172.31.37.235]

TASK [install httpd on RedHat based os] *******************************************************************************************
changed: [172.31.37.118]
changed: [172.31.37.235]
changed: [172.31.44.163]

PLAY RECAP ************************************************************************************************************************
172.31.37.118              : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
172.31.37.235              : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
172.31.44.163              : ok=2    changed=1    unreachable=0    failed=0    skipped=1    rescued=0    ignored=0
```

6.4: Task 4: Using Conditions , Notify , handlers

- Handler: Task which is called only if a notifier is present
- Notifier: Section attributed to a task which calls a handler if the output is changed.

vi condition_notify_handler.yaml

```
---
- hosts: db*
  user: ansible
  become: yes
  connection: ssh
  gather_facts: yes

  vars:
    pkg: httpd

  tasks:
    - name: check server reachability
      ping:
      register: reachability_test   # dash - is not allowed only _
                                    # store output in reachability_test variable

    - name: install '{{pkg}}'
      yum: name='{{pkg}}' state=present
      when: reachability_test.changed == false  # false meaning reachable
                                                # install pkg only if reachability_test is false
      notify: start '{{pkg}}' service

  handlers:
    - name: start '{{pkg}}' service
      service: name='{{pkg}}' state=started
```

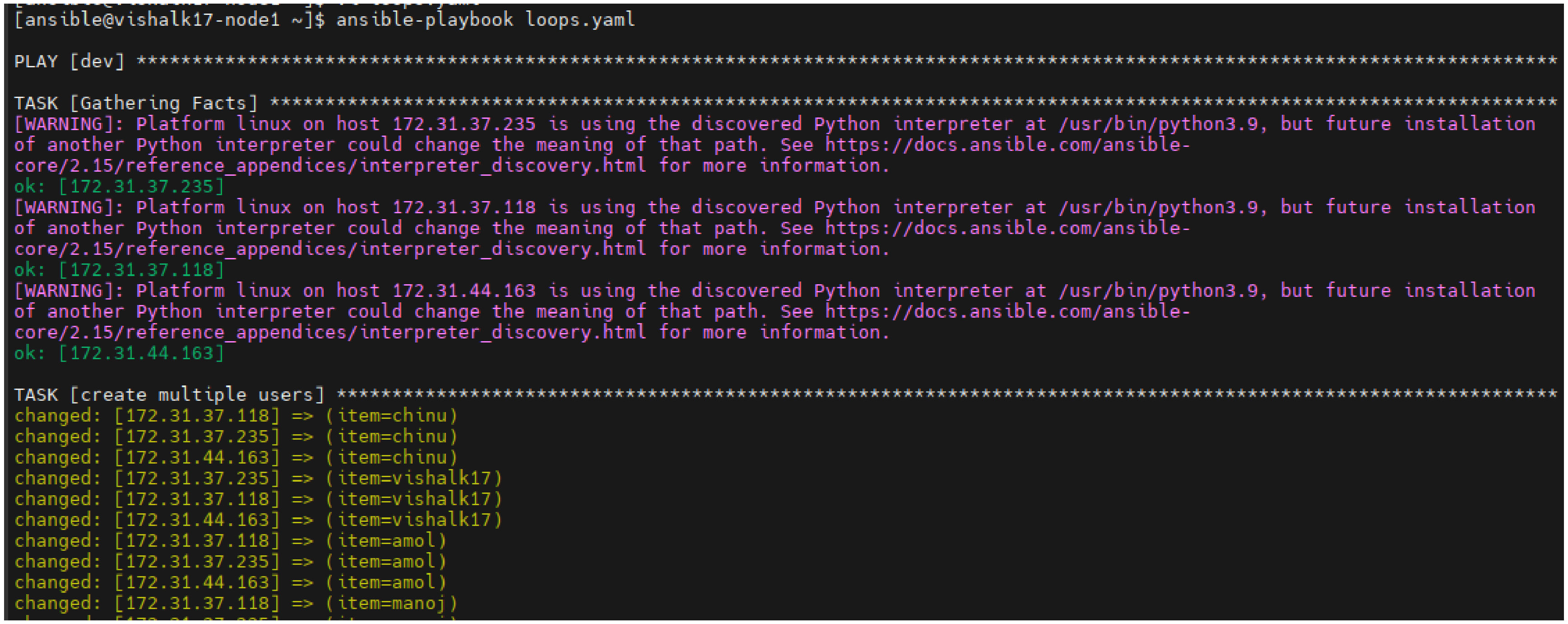


```
[ansible@vishalk17-node1 ~]$ ansible-playbook condition_notify_handler.yaml

PLAY [db*] ****************************************************************************************************************************

TASK [Gathering Facts] ****************************************************************************************************************
[WARNING]: Platform linux on host db2 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation of anoth
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [db2]
[WARNING]: Platform linux on host db1 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation of anoth
Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [db1]

TASK [check_server_reachable] *********************************************************************************************************
ok: [db1]
ok: [db2]

TASK [install 'httpd'] ****************************************************************************************************************
ok: [db1]
ok: [db2]

PLAY RECAP ****************************************************************************************************************************
db1                        : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
db2                        : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

6.5: Task 5: Using loops

vi loops.yaml

```
--- # create multiple users using user module with loop
- hosts: dev
  user: ansible
  become: yes
  connection: ssh

  tasks:
    - name: create multiple users
      user: name='{{item}}' state=present # see syntax '{{item}}' and with_items
      with_items:
        - chinu
        - vishalk17
        - amol
        - manoj
```



```
[ansible@vishalk17-node1 ~]$ ansible-playbook loops.yaml

PLAY [dev] *************************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************************
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.235]
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.118]
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.44.163]

TASK [create multiple users] *******************************************************************************************************
changed: [172.31.37.118] => (item=chinu)
changed: [172.31.37.235] => (item=chinu)
changed: [172.31.44.163] => (item=chinu)
changed: [172.31.37.235] => (item=vishalk17)
changed: [172.31.37.118] => (item=vishalk17)
changed: [172.31.44.163] => (item=vishalk17)
changed: [172.31.37.118] => (item=amol)
changed: [172.31.37.235] => (item=amol)
changed: [172.31.44.163] => (item=amol)
changed: [172.31.37.118] => (item=manoj)
```

For more playbook yamls samples:

https://github.com/vishalk17/devops/tree/main/ansible

7.0 : Ansible Vault

- Ansible allows keeping sensitive data such as passwords or keys in encrypted files,rather than a plaintext in your playbooks.
- Encryption based on AES256 algorithem.

```
ansible-vault create xyz.yml    ...... creating a new encrypted playbook
ansible-vault edit xyz.yml      ...... edit the encrypted playbook
ansible-vault rekey xyz.yml     ...... to change the password of playbook
ansible-vault encrypt xyz.yml   ...... to encrypt the existing playbook
ansible-vault decrypt xyz.yml   ...... to decrypt an encrypted playbook
```

```
[ansible@vishalk17-node1 ~]$ ls
condition_notify_handler.yaml  conditions.yaml  first_playbook.yaml  loops.yaml  using_variables.yaml

[ansible@vishalk17-node1 ~]$ ansible-vault create condition.yaml
New Vault password:
Confirm New Vault password:

# Verify if file encrypted or not
[ansible@vishalk17-node1 ~]$ cat condition.yaml
$ANSIBLE_VAULT;1.1;AES256
64633064353934393935373639326262663362343038303935353236373937636433613863303934
6337326163393061656234663634383363646463366366330a633131383664643564663137383736
33623334663039646161646439383766623861623637626563353735313561383133396131666338
6539323831333035380a353062316366613436633839303130353637326462633636663365643964
6439
```

```
f# If wrong password provided
[ansible@vishalk17-node1 ~]$  ansible-vault edit condition.yaml
Vault password:
ERROR! Decryption failed (no vault secrets were found that could decrypt) for
/home/ansible/condition.yaml

# If correct password provided
[ansible@vishalk17-node1 ~]$  ansible-vault edit condition.yaml
Vault password:
[ansible@vishalk17-node1 ~]$
```

```
[ansible@vishalk17-node1 ~]$ ansible-vault rekey condition.yaml
Vault password:
New Vault password:
Confirm New Vault password:
Rekey successful
```

```
[ansible@vishalk17-node1 ~]$ ansible-vault encrypt loops.yaml
New Vault password:
Confirm New Vault password:
Encryption successful

[ansible@vishalk17-node1 ~]$ ansible-vault decrypt loops.yaml
Vault password:
Decryption successful
```

8.0 : Ansible Roles

Roles are a way to organize and encapsulate tasks, variables, handlers, and other elements needed to accomplish those tasks. They provide a structured approach to managing Ansible playbooks, especially as the complexity and number of tasks increase.

Key Points about Roles

- **Organization Techniques**: We can use two techniques for organizing tasks: includes and roles. Roles are preferred for organizing tasks and encapsulating the necessary data.
- **Structure**: Roles allow us to organize playbooks into a directory structure, making it easier to manage and maintain complex configurations.
- **Maintainability**: Adding more functionality to playbooks can make them difficult to maintain in a single file. Roles help break down these complexities.

Directory Structure of a Role

1. **Defaults**:
   - Stores default variables for the role.
   - Example: Default variables for setting port numbers like 80 or 8080.
2. **Files**:
   - Contains static files that need to be transferred to the remote VM.
3. **Handlers**:
   - Contains tasks that are triggered by other tasks.
   - Handlers are typically used for actions like restarting services after a configuration change.
4. **Meta**:
   - Contains metadata about the role.
   - Includes information such as author name, supported platforms, and dependencies.
5. **Tasks**:
   - Contains the main list of tasks to be executed by the role.
   - Tasks are equivalent to what is typically found in a playbook, such as installing packages and copying files.
6. **Vars**:

- Stores variables specific to the role.
- Both `vars` and `defaults` directories store variables that can be used in configuration files.

8.1 : Task 1: Create role to install nginx in dev group

Create `nginx-webserver.yaml` that will use `nginx-webserver/` role directory

```
[ansible@vishalk17-node1 roles]$ touch  nginx-webserver.yaml
[ansible@vishalk17-node1 roles]$ ansible-galaxy init --force --offline  nginx-webserver
- Role nginx-webserver was created successfully
[ansible@vishalk17-node1 roles]$ tree
.
├── nginx-webserver
│   ├── README.md
│   ├── defaults
│   │   └── main.yml
│   ├── files
│   ├── handlers
│   │   └── main.yml
│   ├── meta
│   │   └── main.yml
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   ├── tests
│   │   ├── inventory
│   │   └── test.yml
│   └── vars
│       └── main.yml
└── nginx-webserver.yaml

9 directories, 9 files
```

```
[ansible@vishalk17-node1 roles]$ rm -rf tests
```

nginx-webserver.yaml

```
--- # roles
- hosts: dev
  user: ansible
  become: yes
  connection: ssh
  roles:
    - nginx-webserver  # role directory name
```

meta/main.yaml

```
---
galaxy_info:
  role_name: nginx-webserver     # The name of the role
  author: vishal-chinu           # The author of the role
  description: Install Nginx     # A brief description of what the role does
  company: vishalk17 group of Business     # The company or organization that created the role
  license: MIT                        # The license under which the role is distributed
```

vars/main.yml

```
---
nginx_package: nginx
nginx_service: nginx
```

tasks/main.yml

```
---
- name: Install Nginx
  apt:
    name: "{{ nginx_package }}"
    state: present
  when: ansible_os_family == 'Debian'

- name: Install Nginx
  yum:
    name: "{{ nginx_package }}"
    state: present
  when: ansible_os_family == 'RedHat'

- name: Copy customized index.html
  copy:
    src: files/index.html
    dest: /usr/share/nginx/html/index.html
    mode: '0644'

- name: Start Nginx
  service:
    name: "{{ nginx_service }}"
    state: started
    enabled: yes

- name: Configure Nginx
  template:
    src: templates/nginx.conf.j2
    dest: /etc/nginx/nginx.conf
    mode: '0644'
  notify: restart nginx

- name: Restart Nginx
  service:
    name: "{{ nginx_service }}"
    state: restarted
```

handlers/main.yml

```
---
- name: restart nginx
  service:
    name: "{{ nginx_service }}"
    state: restarted
```

templates/nginx.conf.j2

```
# /etc/nginx/nginx.conf

events {
    worker_connections  1024;  # Default setting
}

http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    server {
        listen       80;
        server_name  localhost;

        location / {
            root   /usr/share/nginx/html;
            index  index.html index.htm;
        }
    }
}
```

files/index.html

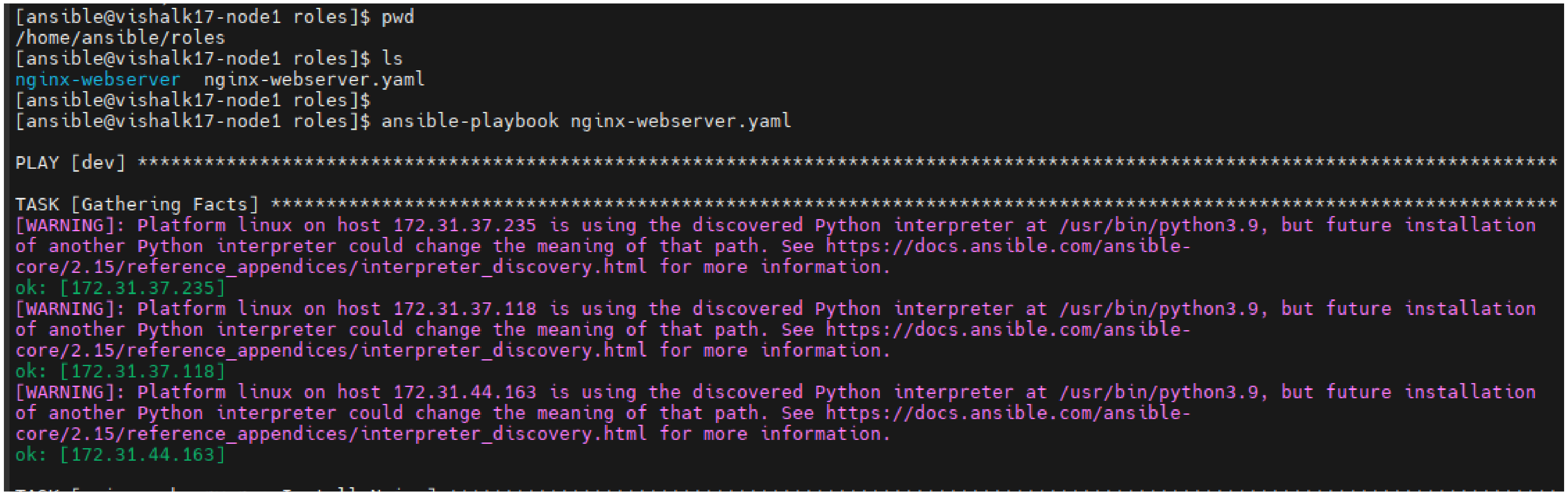
```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Nginx</title>
</head>
<body>
    <h1>Welcome to Nginx</h1>
</body>
</html>
```

Final structure looking like this :

```
[ansible@vishalk17-node1 roles]$ tree
.
├── nginx-webserver
│   ├── defaults
│   │   └── main.yml
│   ├── files
│   │   └── index.html
│   ├── handlers
│   │   └── main.yml
│   ├── meta
│   │   └── main.yml
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   │   └── nginx.conf.j2
│   ├── tests
│   │   ├── inventory
│   │   └── test.yml
│   └── vars
│       └── main.yml
└── nginx-webserver.yaml

9 directories, 10 files
```

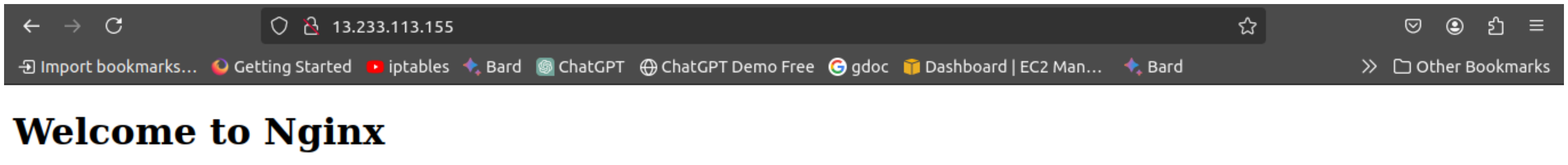
Verify : (Execute Role based playbook)

```
[ansible@vishalk17-node1 roles]$ ansible-playbook nginx-webserver.yaml
```



```
[ansible@vishalk17-node1 roles]$ pwd
/home/ansible/roles
[ansible@vishalk17-node1 roles]$ ls
nginx-webserver  nginx-webserver.yaml
[ansible@vishalk17-node1 roles]$
[ansible@vishalk17-node1 roles]$ ansible-playbook nginx-webserver.yaml

PLAY [dev] ****************************************************************************************************************************

TASK [Gathering Facts] ****************************************************************************************************************
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.235]
[WARNING]: Platform linux on host 172.31.37.118 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.37.118]
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future installation
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
ok: [172.31.44.163]
```

Access nginx on one of node using public ip of that node:



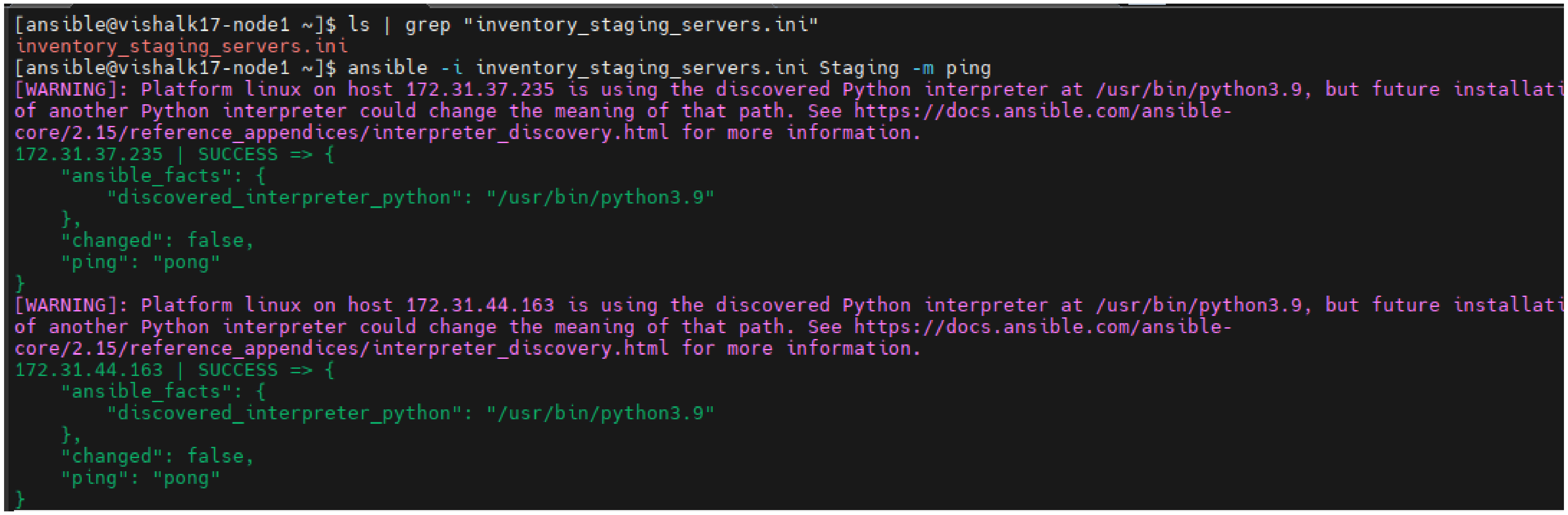
Welcome to Nginx

9.0 : Task: Pass custom inventory file and install httpd

inventory_staging_servers.ini

```
[ansible@vishalk17-node1 ~]$ cat inventory_staging_servers.ini
[Staging]
172.31.44.163
172.31.37.235
```

```
ansible -i inventory_staging_servers.ini Staging -m ping
```



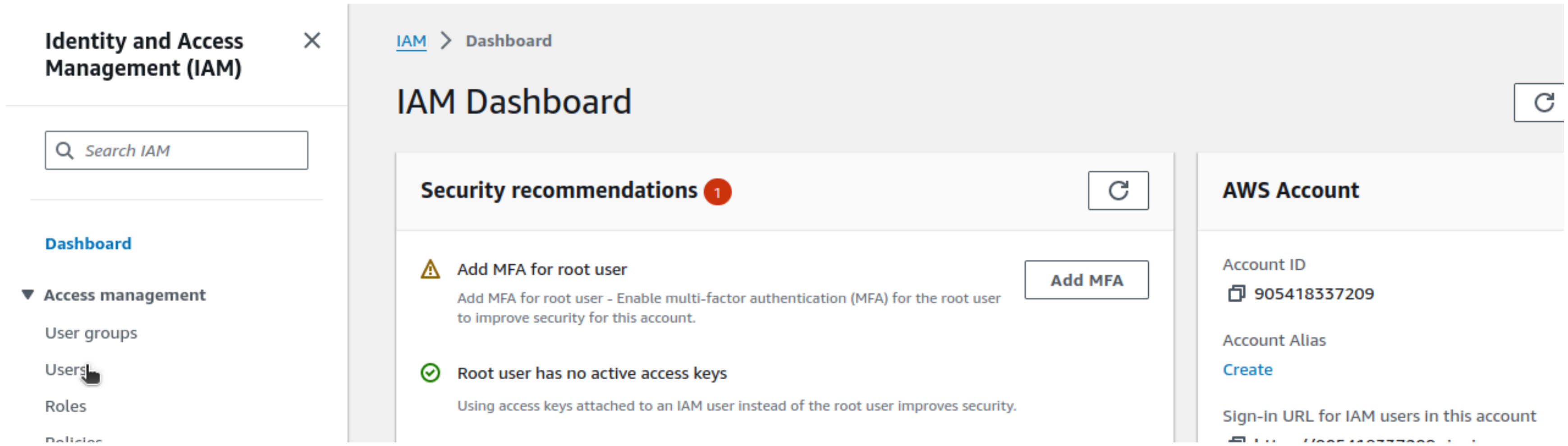
```
[ansible@vishalk17-node1 ~]$ ls | grep "inventory_staging_servers.ini"
inventory_staging_servers.ini
[ansible@vishalk17-node1 ~]$ ansible -i inventory_staging_servers.ini Staging -m ping
[WARNING]: Platform linux on host 172.31.37.235 is using the discovered Python interpreter at /usr/bin/python3.9, but future installati
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.37.235 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.9"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform linux on host 172.31.44.163 is using the discovered Python interpreter at /usr/bin/python3.9, but future installati
of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.15/reference_appendices/interpreter_discovery.html for more information.
172.31.44.163 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3.9"
    },
    "changed": false,
    "ping": "pong"
}
```

10: Task: : Create ec 2 instances with different os , Write ansible playbook role

- Create user which has permission ec2 full access
- 4 ec2 instance with (2 with amazon linux + 2 with ubuntu)
- Instance type : t2.micro
- Region: us-east-1

10.1: Create user which has permission ec2 full access :

- Search for I'am service in aws console >> click on users



Add MFA for root user

Add MFA for root user - Enable multi-factor authentication (MFA) for the root user to improve security for this account.

Root user has no active access keys

Using access keys attached to an IAM user instead of the root user improves security.

Account ID

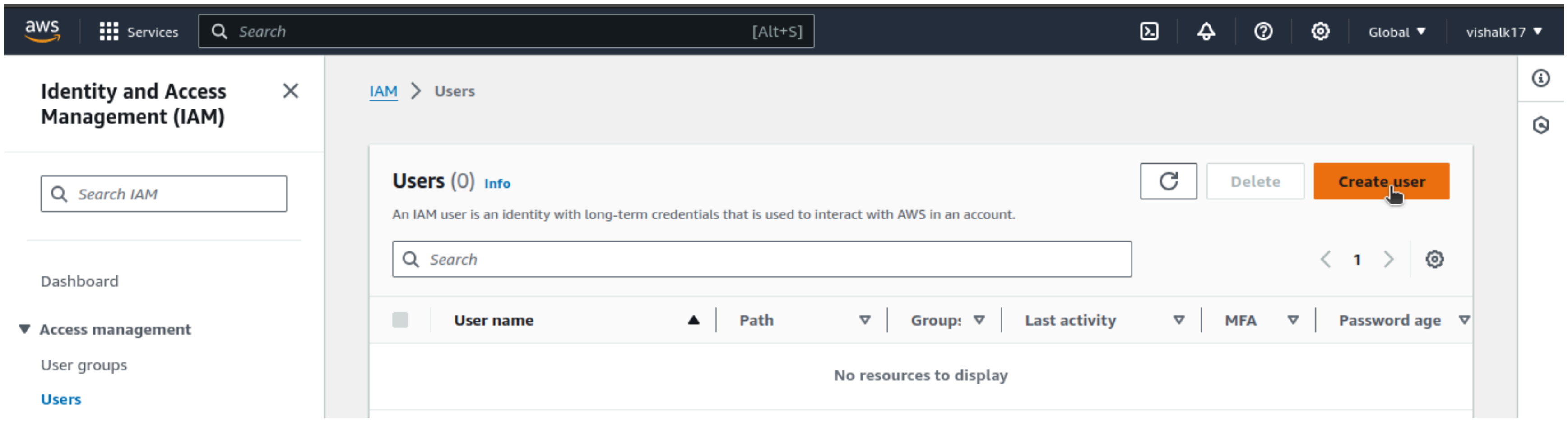
905418337209

Account Alias

Create

Sign-in URL for IAM users in this account

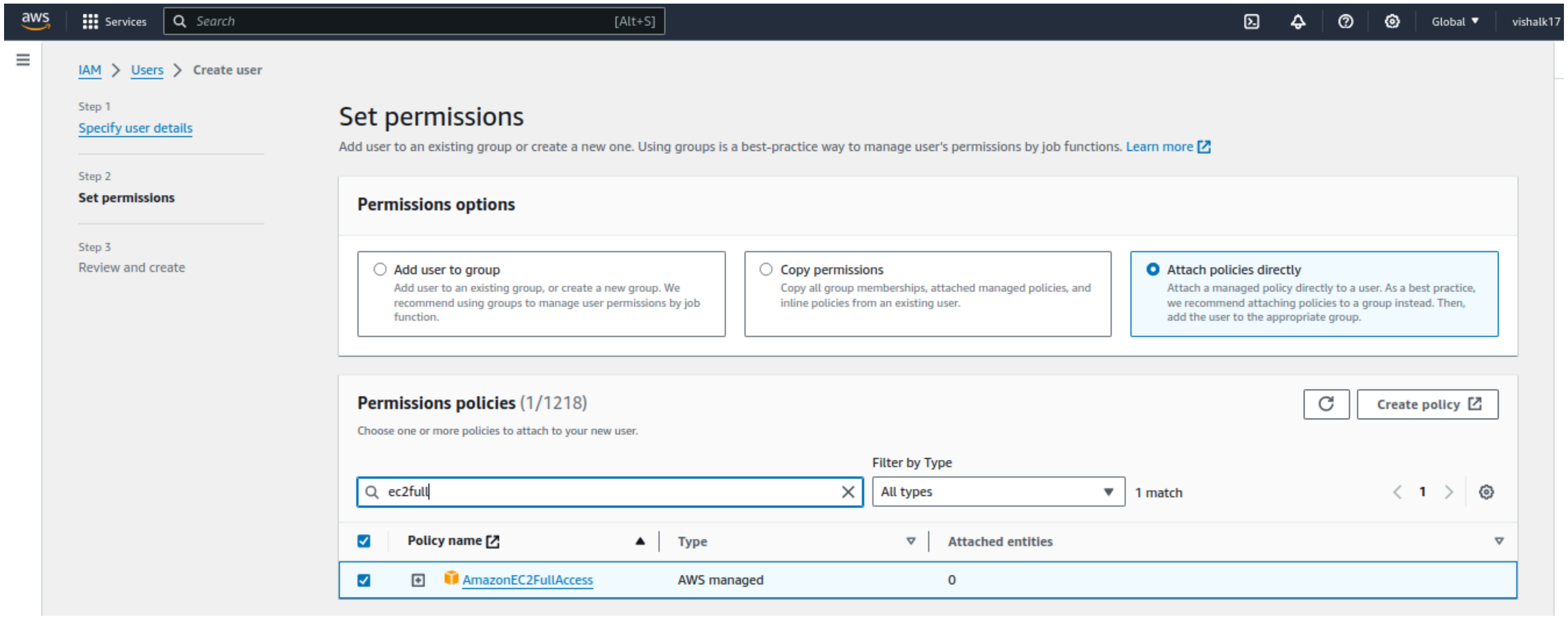
- Click on create user > user name : manoj >> click on attach policies directly >> select ec2 full access >> Next >> create user



An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

No resources to display

-



Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

- Add user to group — Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions — Copy all group memberships, attached managed policies, and inline policies from an existing user.
- [✓ Attach policies directly] — Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1218)

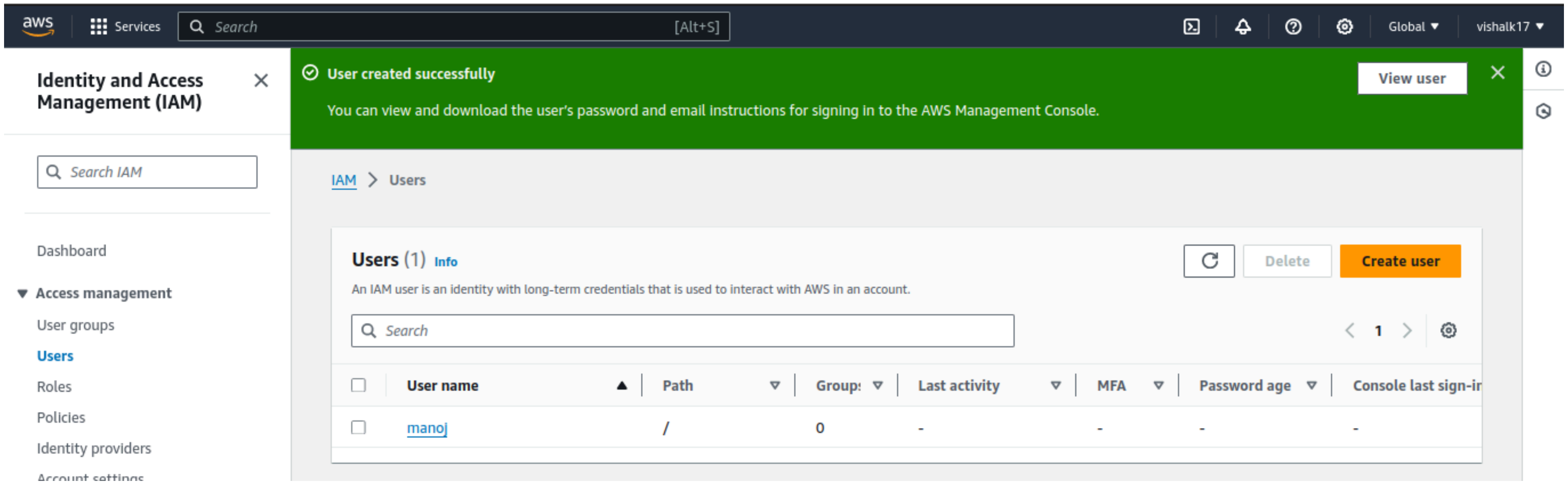
Choose one or more policies to attach to your new user.

ec2full

	Policy name	Type	Attached entities
✓	AmazonEC2FullAccess	AWS managed	0

==

Click on manoj



User created successfully

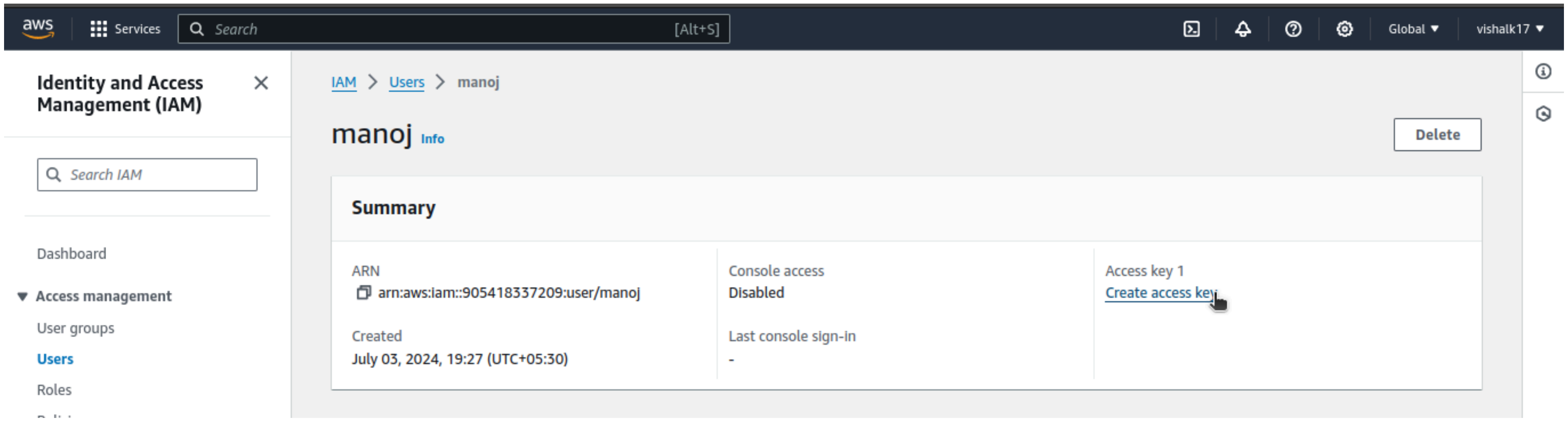
You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in
manoj	/	0	-	-	-	-

- Click on create access key >> select command-line-interface >> and then create



Summary

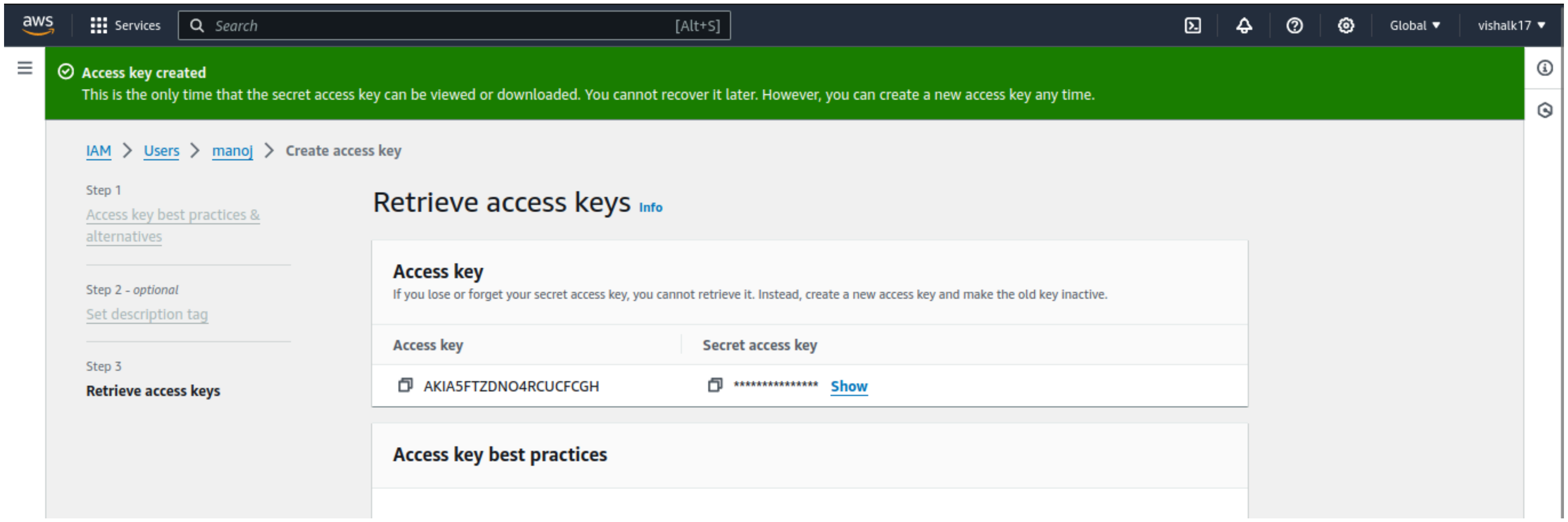
ARN
arn:aws:iam::905418337209:user/manoj

Created
July 03, 2024, 19:27 (UTC+05:30)

Console access
Disabled

Last console sign-in
-

Access key 1
Create access key



Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

IAM > Users > manoj > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Retrieve access keys Info

Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIA5FTZDNO4RCUCFCGH	*************** Show

Access key best practices

- Note down above generated keys

    - Access key : AKIA5FTZDNO4RCUCFCGH
    - Secret Access Key: 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

10.2: Create Ansible role

Ref. this doc.
https://docs.ansible.com/ansible/latest/collections/amazon/aws/ec2_instance_module.html

The below requirements are needed on the host that executes this module.

- python >= 3.6
- boto3 >= 1.26.0
- botocore >= 1.29.0

My ansible server installed on: Amazon linux 2023

```
yum install pip -y
pip install boto3 botocore
```

10.2.1: Install all modules and plugins

Ref.

https://galaxy.ansible.com/ui/repo/published/amazon/aws/?extIdCarryOver=true&sc_cid=701f2000001OH7YAAW

```
[root@vishalk17-node1 ~]# ansible-galaxy collection install amazon.aws --force
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Downloading
https://galaxy.ansible.com/api/v3/plugin/ansible/content/published/collections/artifacts/amazon-aws-8.1.0
.tar.gz to /root/.ansible/tmp/ansible-local-2629i1jyaf7t/tmpn3vh7li4/amazon-aws-8.1.0-lcqhjln9
Installing 'amazon.aws:8.1.0' to '/root/.ansible/collections/ansible_collections/amazon/aws'
amazon.aws:8.1.0 was installed successfully
```

10.2.2 : Initialize Sample structure of ec2_create role

```
[ansible@vishalk17-node1 ~]$ cd roles

[ansible@vishalk17-node1 roles]$ ansible-galaxy init ec2_create
- Role ec2_create was created successfully
```

==

```
[ansible@vishalk17-node1 roles]$ ls
ec2_create  nginx-webserver  nginx-webserver.yaml
```

10.2.3: Create or Edit Yaml files of ec2_create role

Ref. Below examples from the documentation and copy required contents.

https://docs.ansible.com/ansible/latest/collections/amazon/aws/ec2_instance_module.html#examples

ec2_create/tasks/main.yml

```
---
- name: start an instance with a public IP address
 amazon.aws.ec2_instance:
   name: "{{ item.name }}"
   instance_type: "{{ type }}"  # pick from /vars/main.yaml
   region: ap-south-1
   security_group: default
   network:
     assign_public_ip: true
   image_id: "{{ item.image }}"
   aws_access_key: "{{ aws_access_key }}"  # to access aws # from ansible vault
   aws_secret_key: "{{ aws_secret_key }}"
   tags:
     Environment: Testing
 with_items:
   - { image: ami-01376101673c89611, name: amz-1 }
   - { image: ami-01376101673c89611, name: amz-2 }
   - { image: ami-0ad21ae1d0696ad58, name: ubuntu-1}
   - { image: ami-0ad21ae1d0696ad58, name: ubuntu-2}
```

```
# Ansible won't create multiple instances with the same parameters unless specified differently.
# Since the AMI ID is the same for two amz instances and two ubuntu instances,
# if I execute the playbook without providing unique names, it may not create separate instances.
# By providing unique names for each instance, I ensure the creation of 4 distinct instances, even if the
AMI IDs are the same.
```

Key Points:

1. **Ansible's Idempotency**: Ansible is designed to ensure that the desired state is achieved. If you run the same task again and the state has already been achieved, Ansible will not repeat the task. However, when creating instances, specifying unique names helps in distinguishing them as separate entities.
2. **Instance Creation**: When you want to create multiple instances with the same AMI ID, providing unique names is a good practice to ensure that Ansible treats each as a distinct task.

ec2_create/vars/main.yml

```
---
type: t2.micro
```

Create ansible vault password protected pass.yaml for our secret_key and access_key

`group_vars/all/pass.yml`

```
---
aws_access_key: AKIA5FTZDNO4RCUCFCGH
aws_secret_key: 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
```

```
[ansible@vishalk17-node1 roles]$ ls
ec2_create/  ec2_create_main.yaml group_vars/  nginx-webserver  nginx-webserver.yaml

[ansible@vishalk17-node1 roles]$ tree group_vars/
group_vars/
└── all
    └── pass.yml

1 directory, 1 file
[ansible@vishalk17-node1 roles]$ ansible-vault encrypt group_vars/all/pass.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

Create **ec2_create_main.yaml**

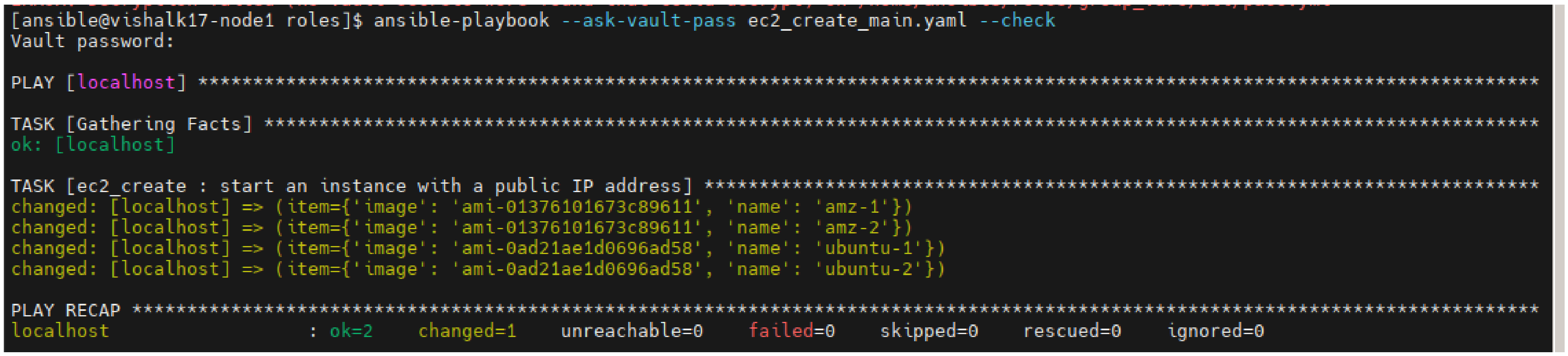
```
---
- hosts: localhost
  become: yes
  gather_facts: true
  connection: local

  roles:
    - ec2_create
```

Final Structure:

```
[ansible@vishalk17-node1 roles]$ pwd
/home/ansible/roles

[ansible@vishalk17-node1 roles]$ ls
ec2_create/  ec2_create_main.yaml group_vars/  nginx-webserver  nginx-webserver.yaml

[ansible@vishalk17-node1 roles]$ tree group_vars/
group_vars/
└── all
    └── pass.yml

1 directory, 1 file

[ansible@vishalk17-node1 roles]$ tree ec2_create
ec2_create
├── tasks
│   └── main.yml
└── vars
    └── main.yml

2 directories, 2 files
```

Verify (Dry-run) :

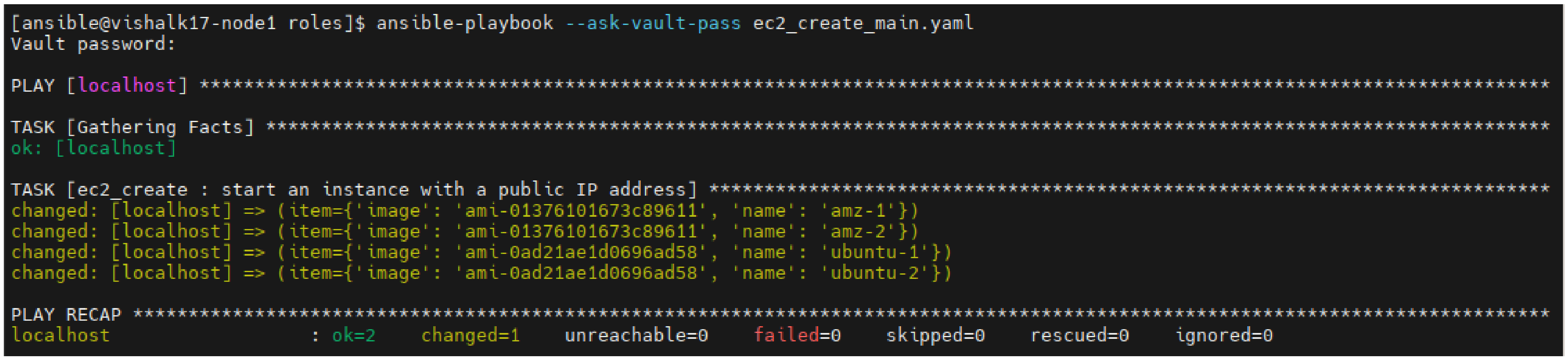
```
[ansible@vishalk17-node1 roles]$ ansible-playbook --ask-vault-pass ec2_create_main.yaml --check
```



```
[ansible@vishalk17-node1 roles]$ ansible-playbook --ask-vault-pass ec2_create_main.yaml --check
Vault password:

PLAY [localhost] ************************************************************************************************************

TASK [Gathering Facts] ******************************************************************************************************
ok: [localhost]

TASK [ec2_create : start an instance with a public IP address] **************************************************************
changed: [localhost] => (item={'image': 'ami-01376101673c89611', 'name': 'amz-1'})
changed: [localhost] => (item={'image': 'ami-01376101673c89611', 'name': 'amz-2'})
changed: [localhost] => (item={'image': 'ami-0ad21ae1d0696ad58', 'name': 'ubuntu-1'})
changed: [localhost] => (item={'image': 'ami-0ad21ae1d0696ad58', 'name': 'ubuntu-2'})

PLAY RECAP ******************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Test :

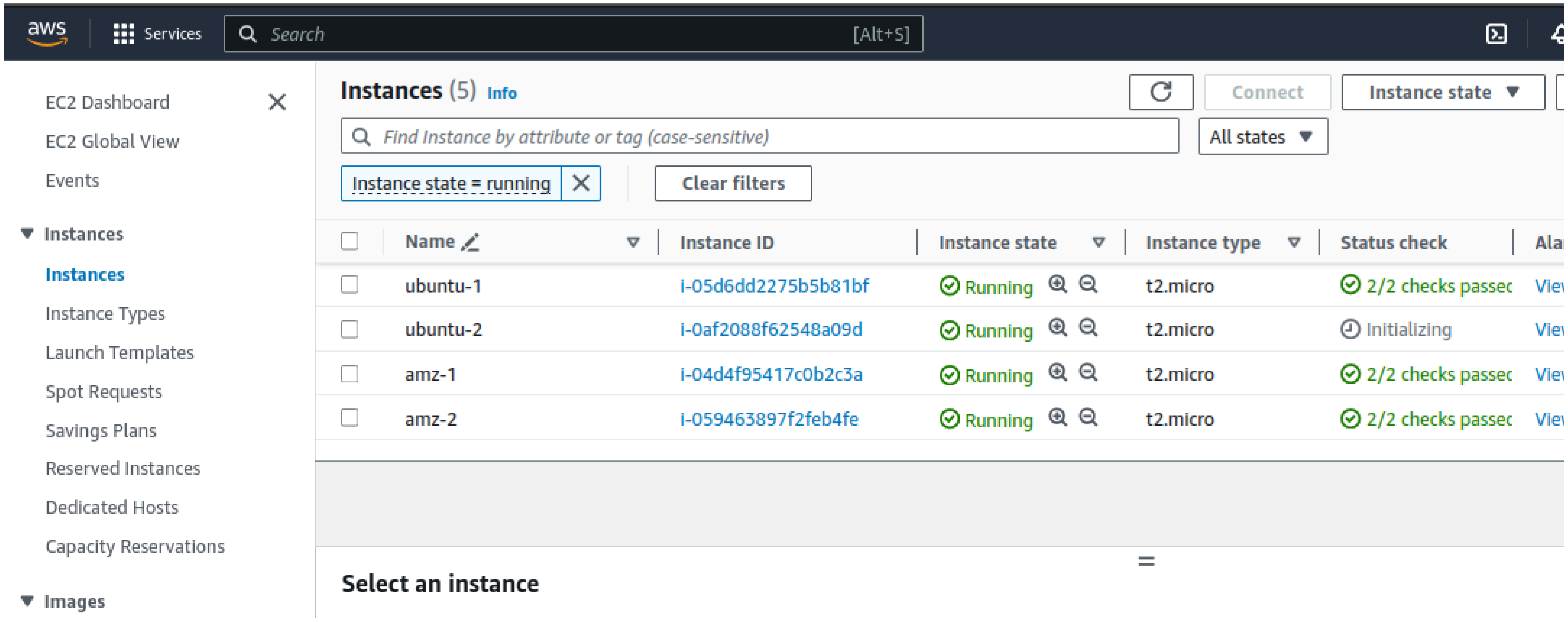
```
[ansible@vishalk17-node1 roles]$ ansible-playbook --ask-vault-pass ec2_create_main.yaml --check
```



```
[ansible@vishalk17-node1 roles]$ ansible-playbook --ask-vault-pass ec2_create_main.yaml
Vault password:

PLAY [localhost] ************************************************************************************************************

TASK [Gathering Facts] ******************************************************************************************************
ok: [localhost]

TASK [ec2_create : start an instance with a public IP address] **************************************************************
changed: [localhost] => (item={'image': 'ami-01376101673c89611', 'name': 'amz-1'})
changed: [localhost] => (item={'image': 'ami-01376101673c89611', 'name': 'amz-2'})
changed: [localhost] => (item={'image': 'ami-0ad21ae1d0696ad58', 'name': 'ubuntu-1'})
changed: [localhost] => (item={'image': 'ami-0ad21ae1d0696ad58', 'name': 'ubuntu-2'})

PLAY RECAP ******************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

==

Check the AWS Console to verify whether the instances have been created or not.



Instances (5) Info

Name	Instance ID	Instance state	Instance type	Status check	Ala
ubuntu-1	i-05d6dd2275b5b81bf	Running	t2.micro	2/2 checks passec	Viev
ubuntu-2	i-0af2088f62548a09d	Running	t2.micro	Initializing	Viev
amz-1	i-04d4f95417c0b2c3a	Running	t2.micro	2/2 checks passec	Viev
amz-2	i-059463897f2feb4fe	Running	t2.micro	2/2 checks passec	Viev

Result : Successfully Created EC2 instances using ansible role.

Ref. & SourceCode:

https://github.com/vishalk17/devops/tree/main/ansible/roles