**Ansible Assignment**

1. **Create the file on the target machines or servers as mentioned in the inventory file and the master’s group, save the below code with. yml extension and run the playbook.**

We will create a playbook for the above task

vi files.yml



Contents of the Playbook

---

- hosts: Master

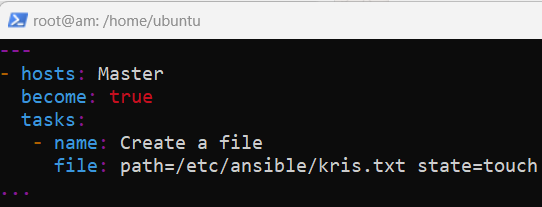
become: true

tasks:

- name: Create a file

file: path=/etc/ansible/kris.txt state=touch

...



In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook files.yml

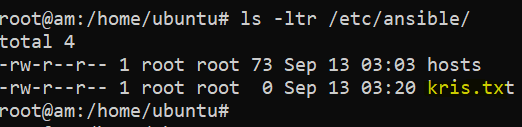


We see the playbook ran successfully.

**Output:**

**Master**

ls -ltr /etc/ansible/



1. **Create a directory with the mode as 775 and owner/group as Ansible.**

We will create a playbook for the above task

vi dir.yml



Contents of the Playbook

---

- hosts: Master

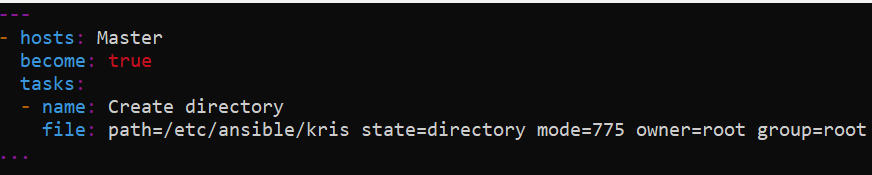
become: true

tasks:

- name: Create directory

file: path=/etc/ansible/kris state=directory mode=775 owner=root group=root

...

****

In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook dir.yml

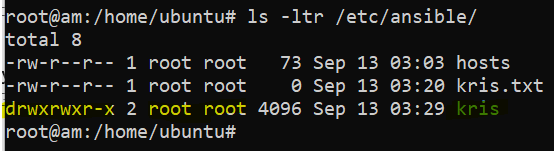


We see the playbook ran successfully.

**Output:**

**Master**

ls -ltr /etc/ansible/

f

1. **Create a user. Let’s look at the user module to create and delete users in the playbook.**

We will create a playbook for the above task

vi useradd.yml

****

Contents of the Playbook

---

- hosts: Worker1

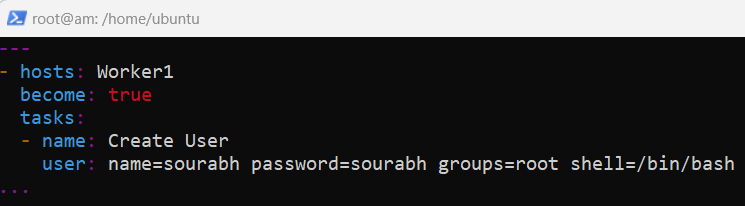
become: true

tasks:

- name: Create User

user: name=sourabh password=sourabh groups=root shell=/bin/bash

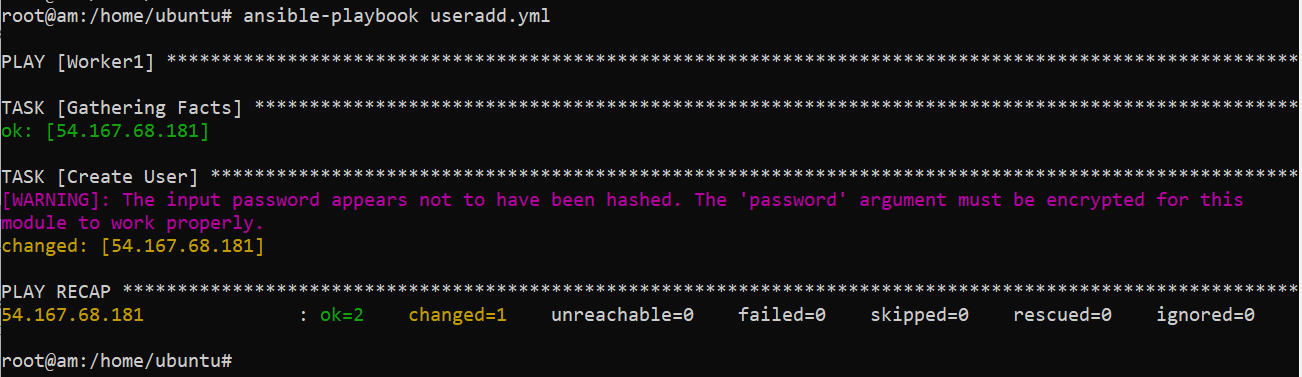
...

****

In our case, we have taken hosts as the Worker1.

Now we will run the playbook

ansible-playbook useradd.yml

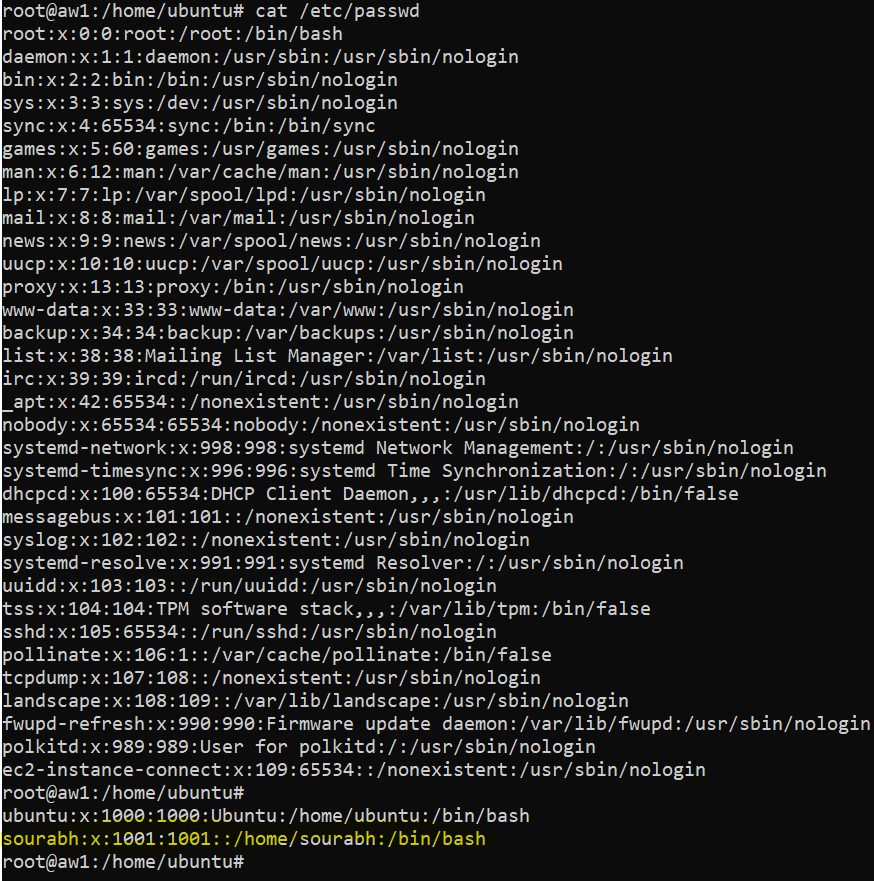
****

We see the playbook ran successfully. It also gave the warning to keep password encrypted.

**Output:**

**Worker1**

cat /etc/passwd

****

1. **Remove user. Removing a user is very easy and it will need the state to be set to absent. This is equivalent to the userdel command in Linux.**

We will create a playbook for the above task

vi userdel.yml

****

Contents of the Playbook

---

- hosts: Worker1

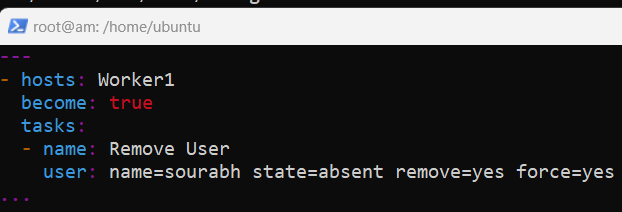
become: true

tasks:

- name: Remove User

user: name=sourabh state=absent remove=yes force=yes

…

****

In our case, we have taken hosts as the Worker1.

Now we will run the playbook

ansible-playbook userdel.yml

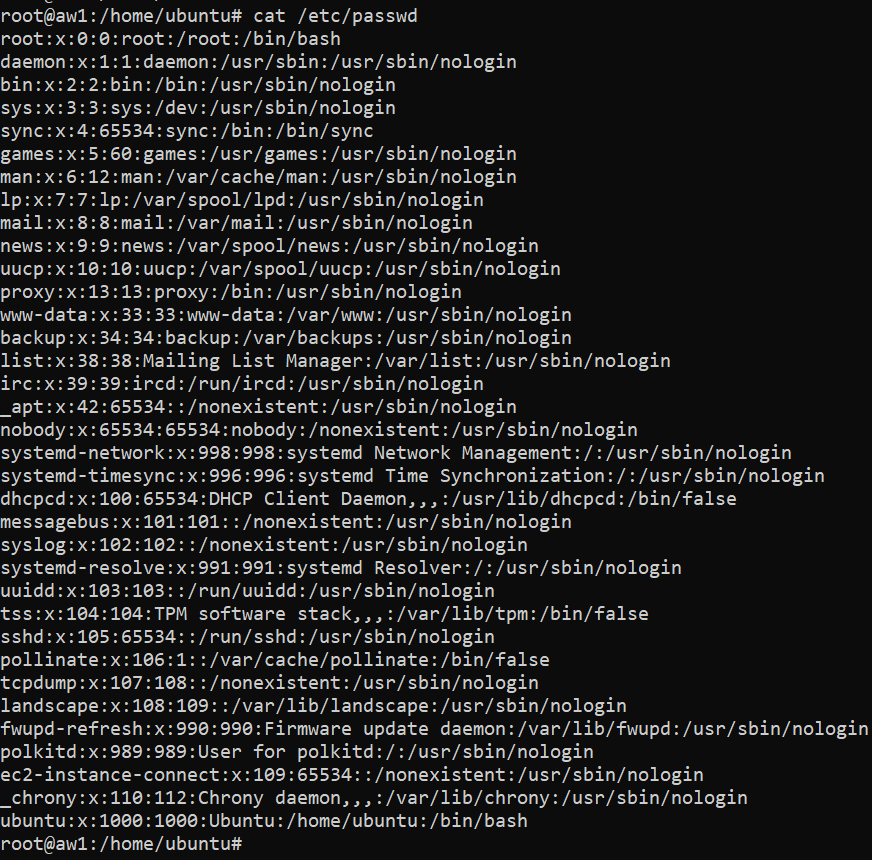
****

We see the playbook ran successfully.

**Output:**

**Worker1**

cat /etc/passwd

****

1. **Copy content to a file using the copy module. If you need to copy a file to the target machines or servers use the src and dest in the copy module.**

We will create a playbook for the above task

vi copy.yml



Contents of the Playbook

---

- hosts: Master

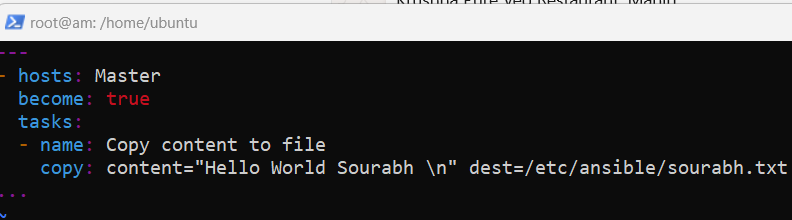
become: true

tasks:

- name: Copy content to file

copy: content="Hello World Sourabh \n" dest=/etc/ansible/sourabh.txt

...

****

In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook copy.yml

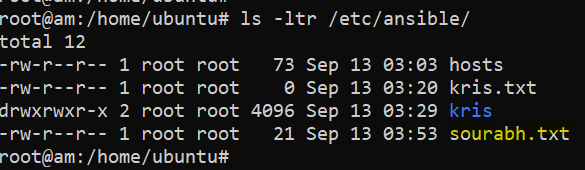


We see the playbook ran successfully.

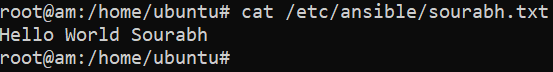
**Output:**

**Master**

ls -ltr /etc/ansible/

****

cat /etc/ansible/sourabh.txt



1. **Replace all instances of a string. Using replace module we can replace a word with another word. The replace module will need 3 parameters i.e. ‘path’, ‘regexp’ (to find the particular word) and ‘replace’ (providing another word for replacement).**

We will create a playbook for the above task

vi replace.yml



Contents of the Playbook

---

- hosts: Master

tasks:

- name: Replace example

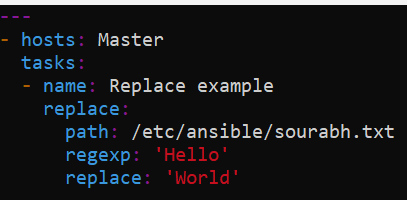
replace:

path: /etc/ansible/sourabh.txt

regexp: 'Hello'

replace: ‘World’

...

****

In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook replace.yml

****

We see the playbook ran successfully.

**Output:**

**Master**

cat /etc/ansible/sourabh.txt

****

1. **Archive or ZIP files and Folders. Using the Ansible archive module you can compress files or folders to ‘zip’, ‘.gz’, or ‘bz2’ format.**

We will create a playbook for the above task

vi archive.yml



Contents of the Playbook

---

- hosts: Master

become: true

tasks:

- name: Ansible zip file example

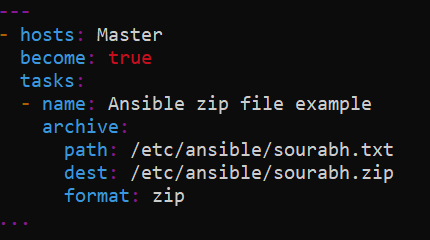
archive:

path: /etc/ansible/sourabh.txt

dest: /etc/ansible/sourabh.zip

format: zip

...

****

In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook archive.yml

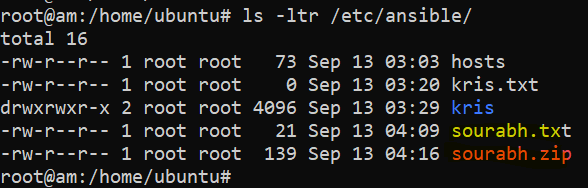


We see the playbook ran successfully.

**Output:**

**Master**

ls -ltr /etc/ansible/



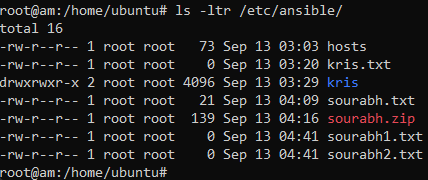
1. **Playbook to zip multiple files to sourabh.zip file.**

In this we will create two more files at the location /etc/ansible/

touch /etc/ansible/sourabh1.txt

touch /etc/ansible/sourabh2.txt

ls -ltr /etc/ansible/



We will create a playbook for the above task

vi multiple.yml



Contents of the Playbook

---

- hosts: Master

tasks:

- name: Ansible zip multiple files example

archive:

path:

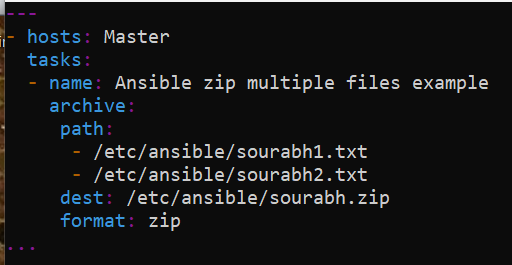
- /etc/ansible/sourabh1.txt

- /etc/ansible/sourabh2.txt

dest: /etc/ansible/sourabh.zip

format: zip

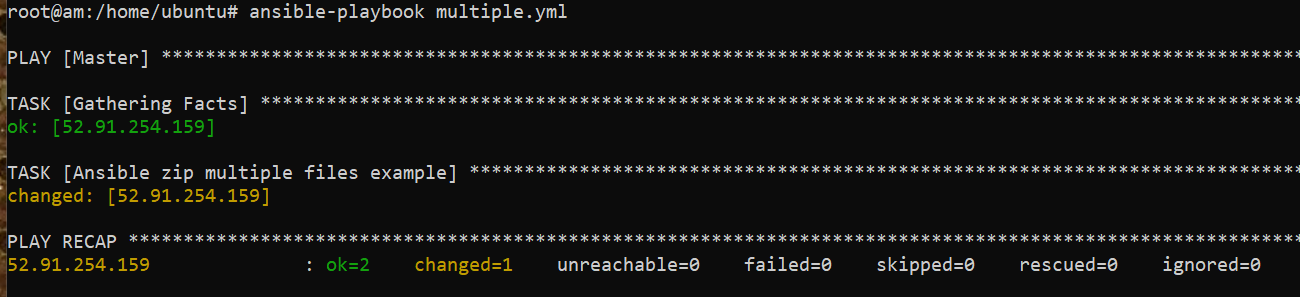
...



In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook multiple.yml

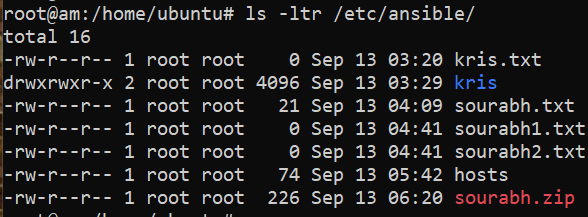
****

We see the playbook ran successfully.

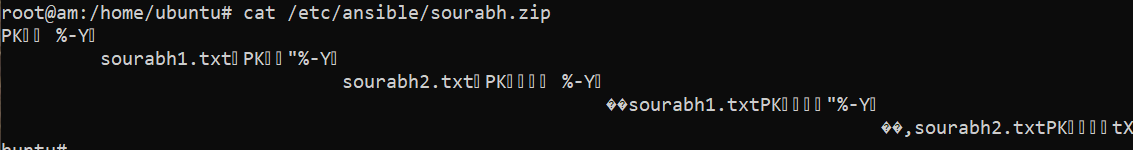
**Output:**

**Master**

ls -ltr /etc/ansible/

****

cat /etc/ansible/sourabh.zip

****

1. **Playbook to zip all files in the /etc/ansible directory**

We will create a playbook for the above task

vi all.yml



Contents of the Playbook

---

- hosts: Master

tasks:

- name: Ansible zip directory example

archive:

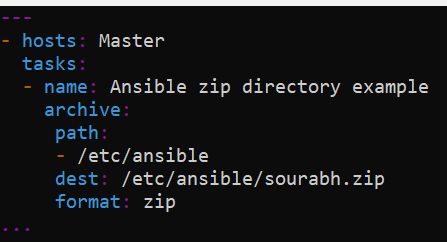
path:

- /etc/ansible

dest: /etc/ansible/sourabh.zip

format: zip

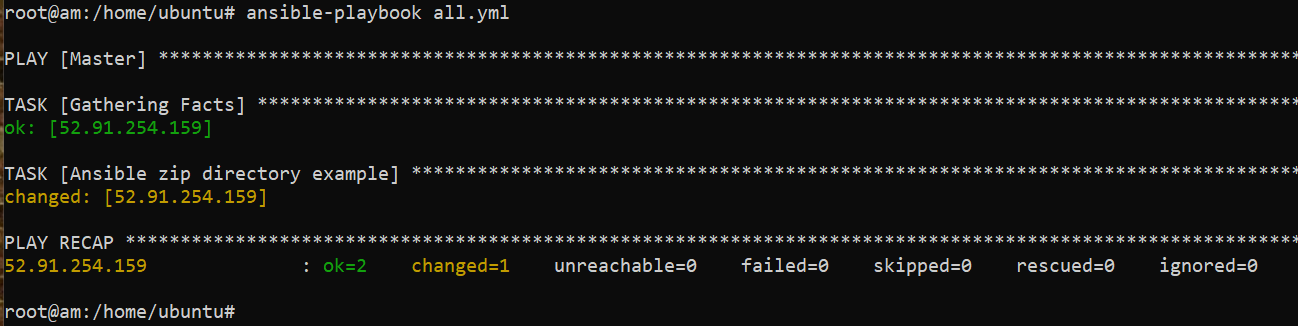
...

****

In our case, we have taken hosts as the master.

Now we will run the playbook

ansible-playbook all.yml

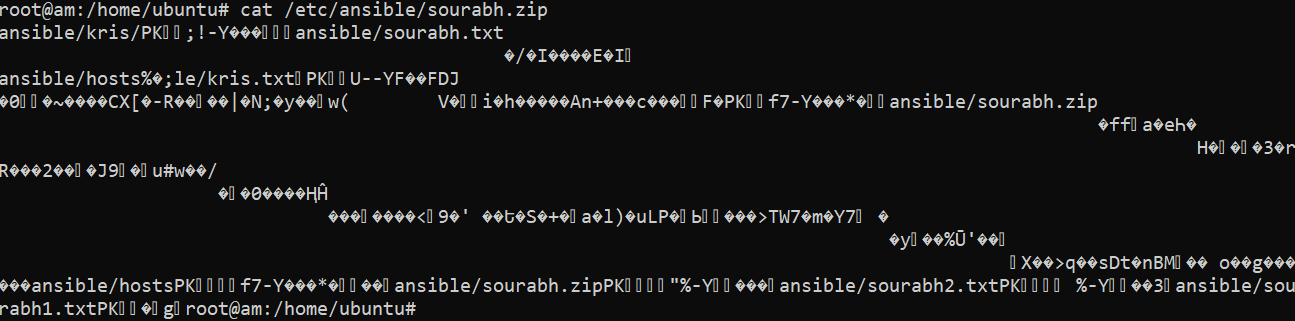


We see the playbook ran successfully.

**Output:**

**Master**

cat /etc/ansible/sourabh.zip

****

1. **Working with date. Using the system date and timestamp helps in certain status or logging purposes. The Ansible facts provide access to remote or target servers date and time. So, we can use the debug module to print the output along with the var attribute as shown below.**

We will create a playbook for the above task

vi date.yml



Contents of the Playbook

---

- hosts: all

become: true

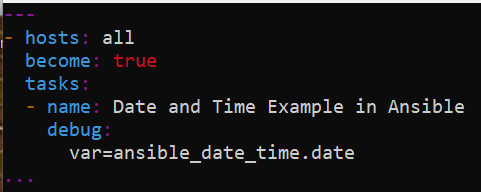
tasks:

- name: Date and Time Example in Ansible

debug:

var=ansible\_date\_time.date

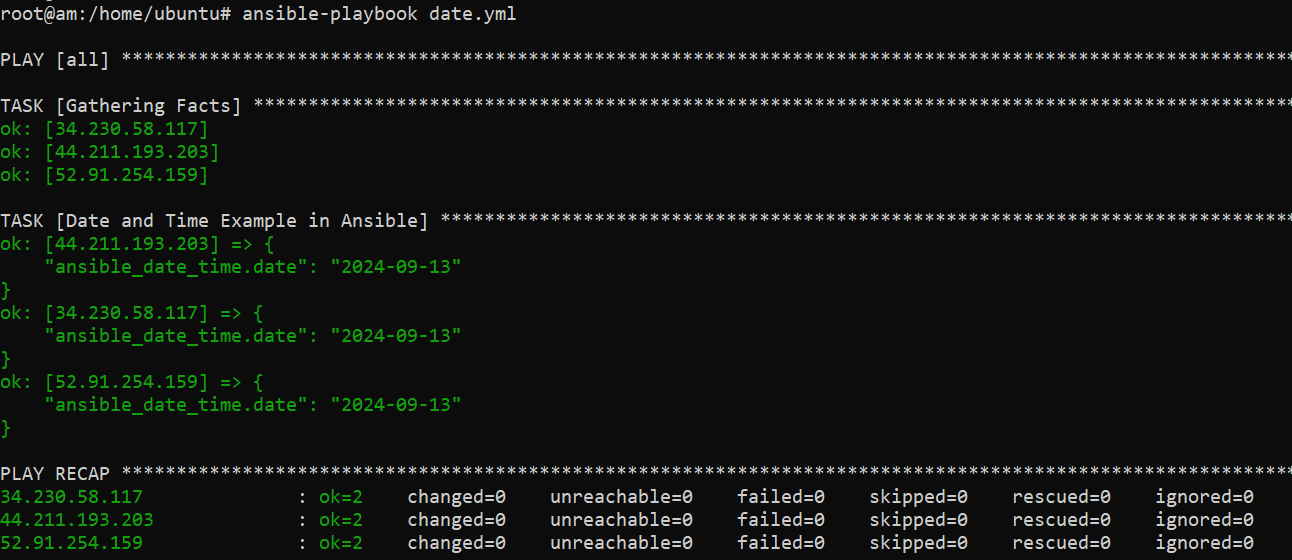
...

****

In our case, we have taken hosts as all.

Now we will run the playbook

ansible-playbook date.yml



We see the playbook ran successfully.

**Output:**

Date is displayed in the output of the playbook itself

1. **Working with time. Using the system date and timestamp helps in certain status or logging purposes. The Ansible facts provide access to remote or target servers date and time. So, we can use the debug module to print the output along with the var attribute as shown below.**

We will create a playbook for the above task

vi time.yml



Contents of the Playbook

---

- hosts: all

become: true

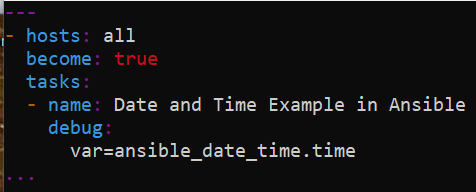
tasks:

- name: Date and Time Example in Ansible

debug:

var=ansible\_date\_time.time

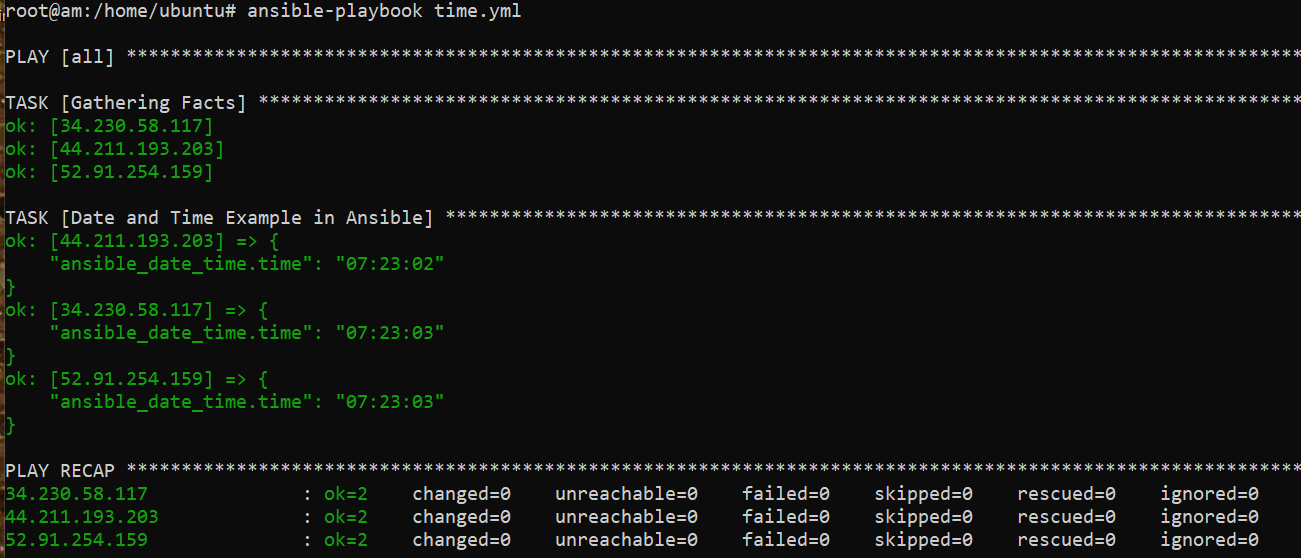
...

****

In our case, we have taken hosts as all.

Now we will run the playbook

ansible-playbook time.yml



We see the playbook ran successfully.

**Output:**

Time is displayed in the output of the playbook itself

1. **Variables Example. Variables are used to store values. In the below Example I am declaring the variable name with value Sourabh. The output will be Sourabh.**

We will create a playbook for the above task

vi variable.yml



Contents of the Playbook

---

- hosts: Worker2

vars:

- name: sourabh

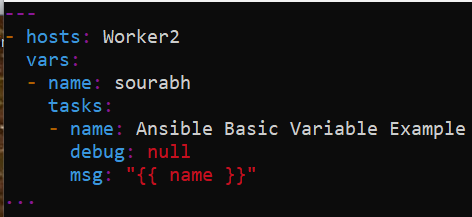
tasks:

- name: Ansible Basic Variable Example

debug: null

msg: "{{ name }}"

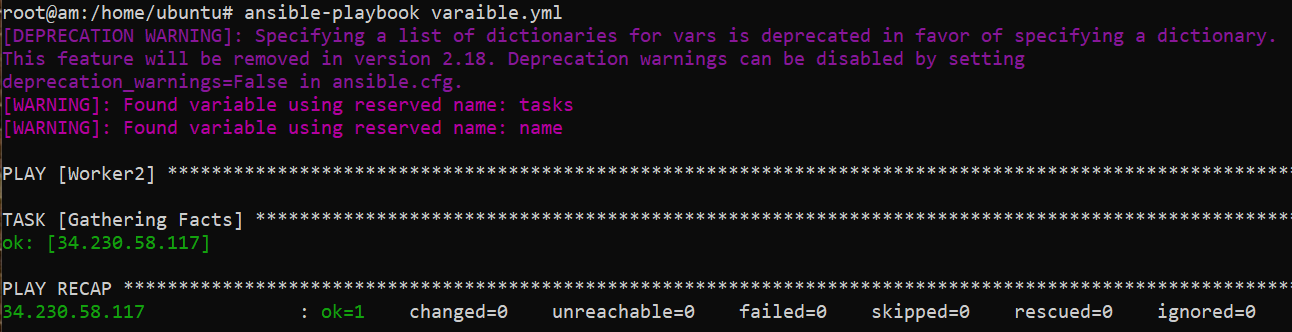
...

****

In our case, we have taken hosts as Worker2.

Now we will run the playbook

ansible-playbook variable.yml



We see the playbook ran successfully.

**Output:**

It is displayed in the output of the playbook itself

1. **An array or a list of variables**

We will create a playbook for the above task

vi array.yml



Contents of the Playbook

---

- hosts: all

vars:

- username: ubuntu

- home: /home/ubuntu

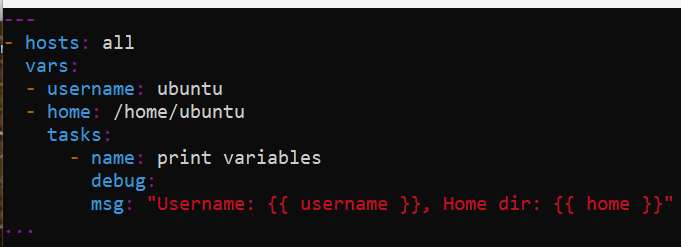
tasks:

- name: print variables

debug:

msg: "Username: {{ username }}, Home dir: {{ home }}"

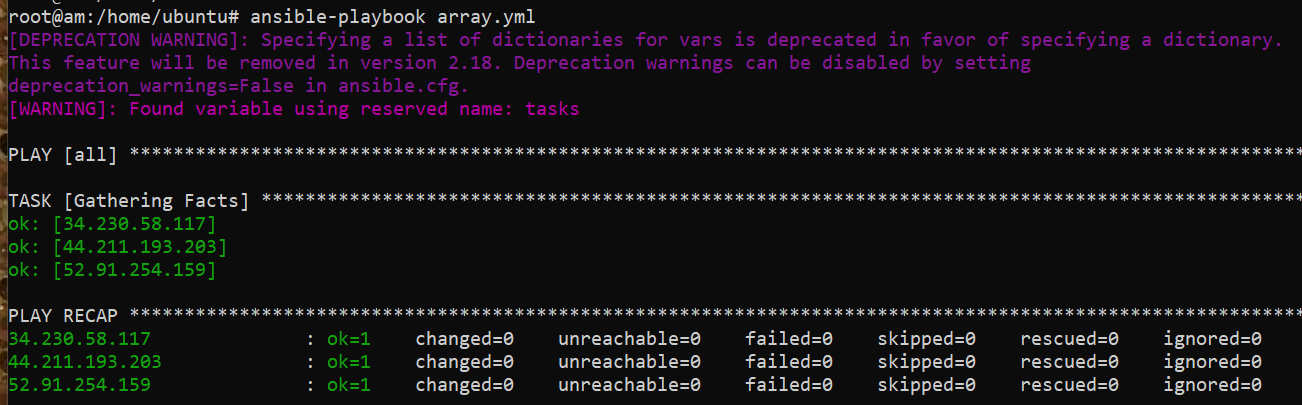
...

****

In our case, we have taken hosts as all.

Now we will run the playbook

ansible-playbook array.yml



We see the playbook ran successfully.

**Output:**

It is displayed in the output of the playbook itself

1. **Playbook to install vim editor and GIT on the target servers or machines. In this playbook, we have made use of the apt module to install the latest version of the software packages.**

We will create a playbook for the above task

vi gitvim.yml



Contents of the Playbook

---

- hosts: all

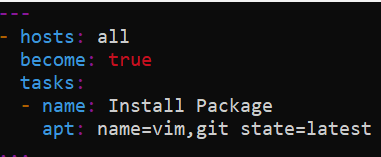
become: true

tasks:

- name: Install Package

apt: name=vim,git state=latest

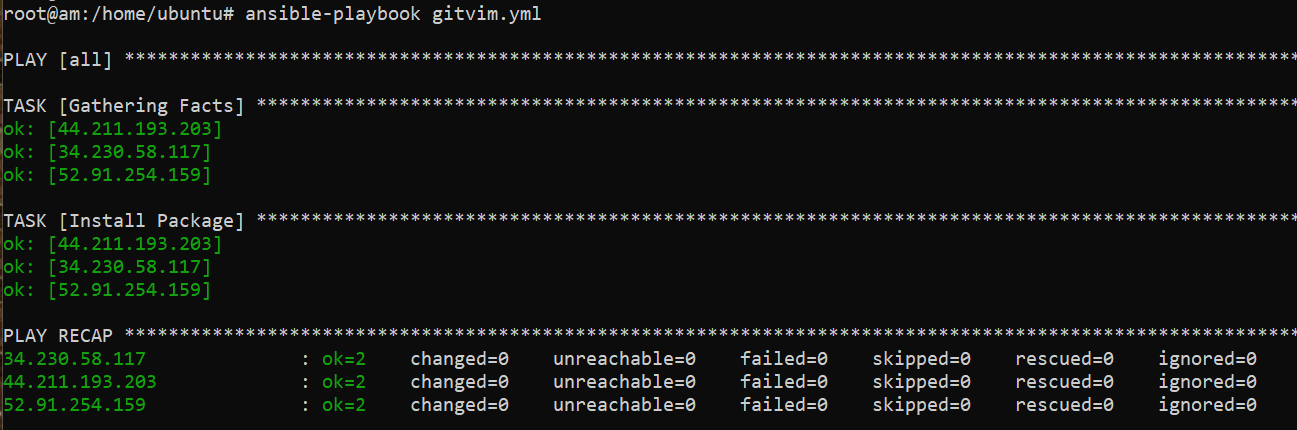
...

****

In our case, we have taken hosts as all.

Now we will run the playbook

ansible-playbook gitvim.yml



We see the playbook ran successfully without any change as package are already installed.

**Output:**

It is displayed in the output of the playbook itself as it ran without any change as package are already installed.

1. **Playbook to install mini-httpd on the target servers or machines. In this playbook, we have made use of the service module to install the latest version of the software packages.**

We will create a playbook for the above task

vi mini-httpd.yml



Contents of the Playbook

---

- hosts: Worker1

become: true

tasks:

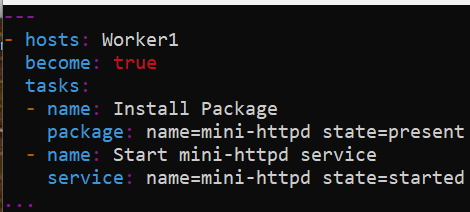
- name: Install Package

package: name=mini-httpd state=present

- name: Start mini-httpd service

service: name=mini-httpd state=started

...

****

In our case, we have taken hosts as Worker1.

Now we will run the playbook

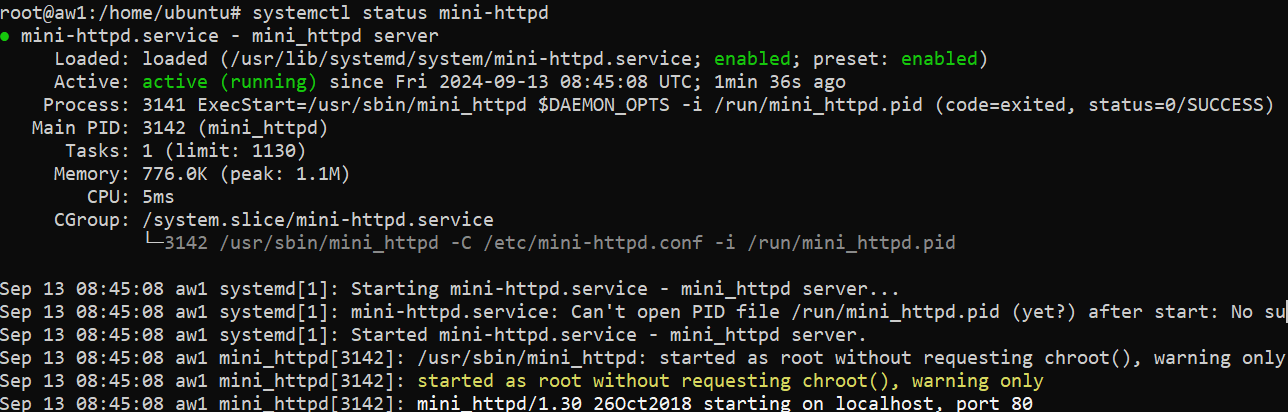
ansible-playbook mini-httpd.yml



We see the playbook ran successfully.

**Output:**

**Worker1**



1. **Install JDK. The following playbook will automate to install JDK 8 on all target machines or servers. JDK is a pre-requisite for most of the other software packages like Maven or Tomcat.**

We will create a playbook for the above task

vi jdk18.yml



Contents of the Playbook

- name: Java installation

hosts: all

become: true

tasks:

- name: Download the JDK binaries

get\_url:

url: https://download.java.net/java/GA/jdk18.0.2.1/db379da656dc47308e138f21b33976fa/1/GPL/openjdk-18.0.2.1\_linux-x64\_bin.tar.gz

dest: /opt/openjdk-18.0.2.1\_linux-x64\_bin.tar.gz

- name: Extract the downloaded file

command: tar xvf /opt/openjdk-18.0.2.1\_linux-x64\_bin.tar.gz -C /opt

- name: Update profile

copy: content="export JAVA\_HOME=/opt/jdk-18.0.2.1 \n" dest=/etc/profile.d/java.sh

- name: Set the JAVA\_HOME in /etc/profile.d/java.sh file

lineinfile:

path: /etc/profile.d/java.sh

state: present

line: 'export PATH=${JAVA\_HOME}/bin:${PATH}'

- name: Reload /etc/profile file

shell:

cmd: source /etc/profile.d/java.sh

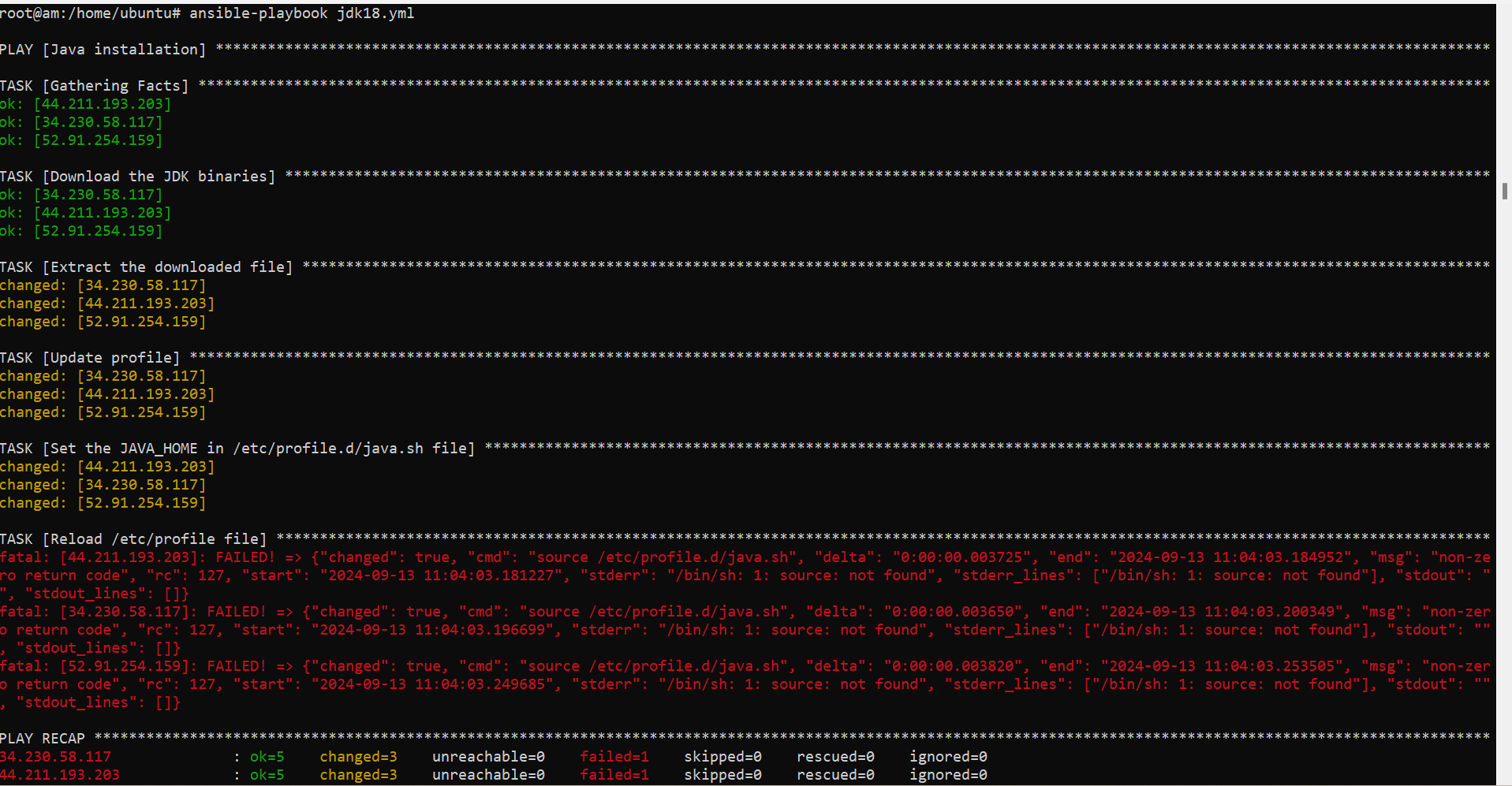
...

****

In our case, we have taken hosts as all.

Now we will run the playbook

ansible-playbook jdk18.yml

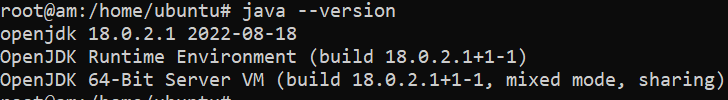


We see the playbook ran successfully.

**Output:**

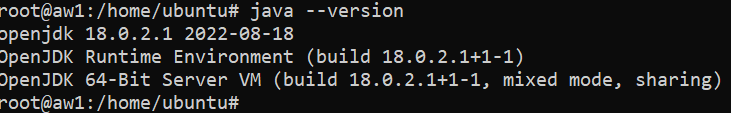
**Master**

java --version

****

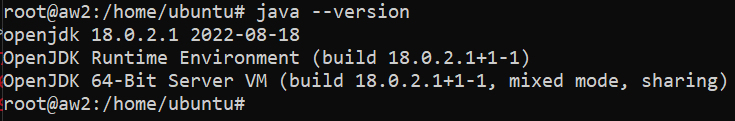
**Worker1**

java --version

****

**Worker2**

java --version

****

1. **Install Maven. The tasks performed are to download the maven file from the URL using the get\_url module, extract the file downloaded, move it to a smaller directory, update and run the profile where the maven is added to the path.**

We will create a playbook for the above task

vi mvn.yml



Contents of the Playbook

- name: Maven installation

hosts: Worker2

become: true

tasks:

- name: Install OpenJDK 17

apt:

name: openjdk-17-jdk

state: present

- name: Download the Maven binaries

get\_url:

url: https://dlcdn.apache.org/maven/maven-3/3.9.9/binaries/apache-maven-3.9.9-bin.tar.gz

dest: /opt/apache-maven-3.9.9-bin.tar.gz

- name: Extract Maven

command: tar xvf /opt/apache-maven-3.9.9-bin.tar.gz -C /opt

- name: Update profile

copy: content="export M2\_HOME=/opt/apache-maven-3.9.9 \n" dest=/etc/profile.d/maven.sh

- name: Set the M2\_HOME in /etc/profile.d/maven.sh file

lineinfile:

path: /etc/profile.d/maven.sh

state: present

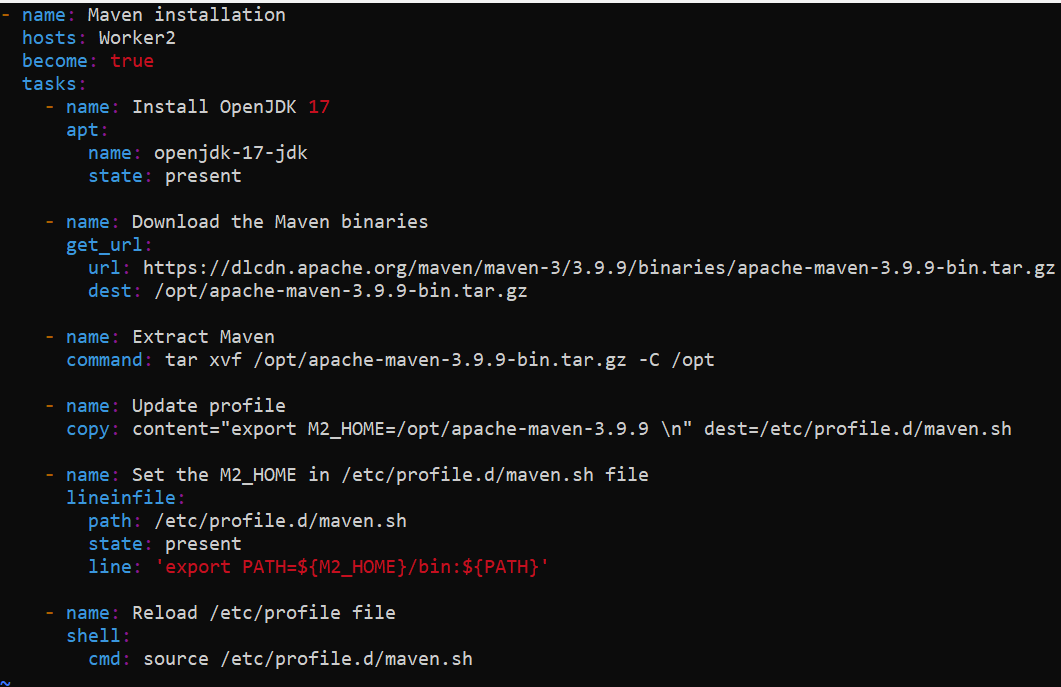
line: 'export PATH=${M2\_HOME}/bin:${PATH}'

- name: Reload /etc/profile file

shell:

cmd: source /etc/profile.d/maven.sh

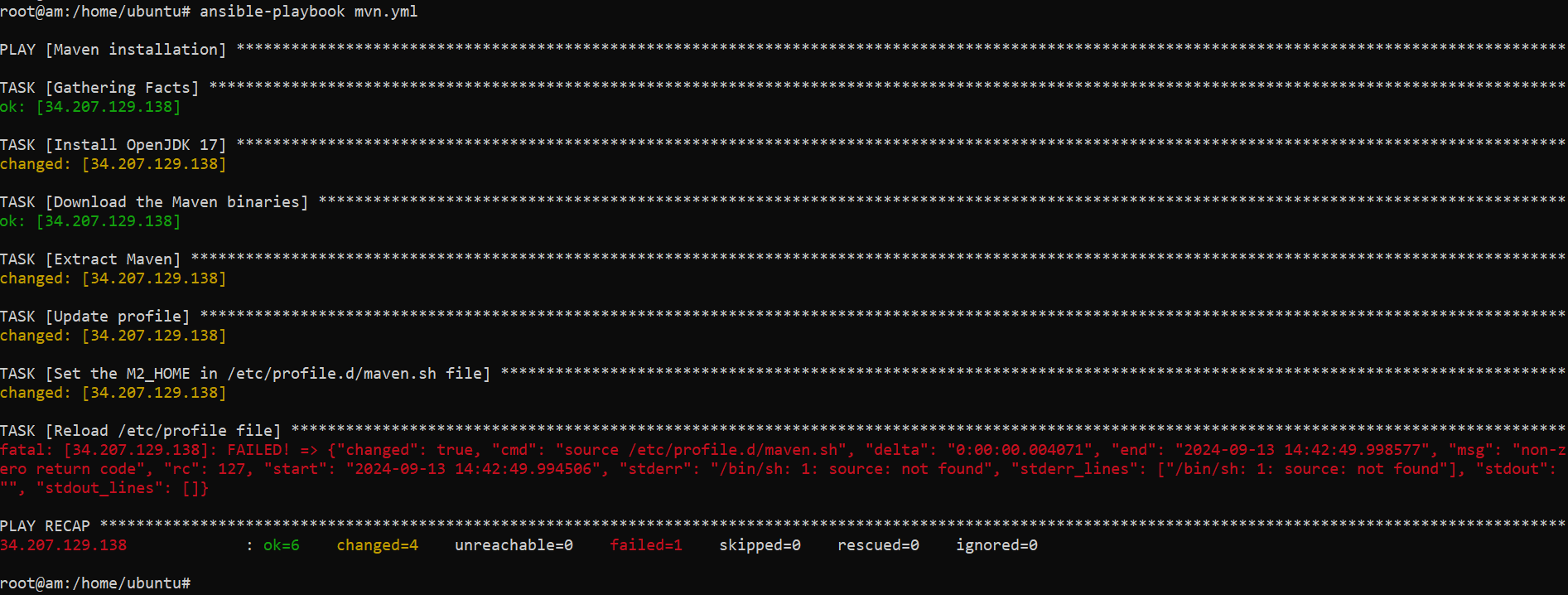
...

****

In our case, we have taken hosts as Worker2.

Now we will run the playbook

ansible-playbook mvn.yml

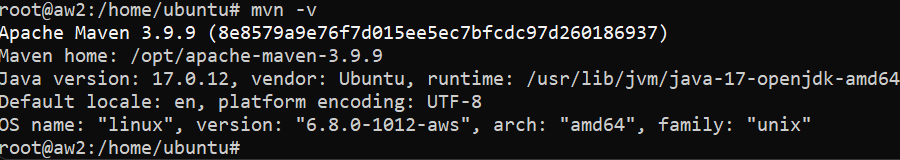


We see the playbook ran successfully.

**Output:**

**Worker2**

mvn -v

****

1. **Install Tomcat. The below playbook helps to install and start Tomcat 8 on to the target machines or servers.**

We will create a playbook for the above task

vi tomcat.yml



Contents of the Playbook

---

- hosts: Worker1

become: true

tasks:

- name: Install JDK and Maven

apt:

name:

- openjdk-17-jdk

- maven

state: present

- name: Download Tomcat 9 and unzip the folder

shell: |

cd /etc

wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz

tar xzfv apache-tomcat-9.0.93.tar.gz

mv apache-tomcat-9.0.93 tomcat

rm apache-tomcat-9.0.93.tar.gz

- name: Setup tomcat service

file: path=/etc/systemd/system/tomcat.service state=touch

- name: Add content in service file

blockinfile:

dest: /etc/systemd/system/tomcat.service

block: |

[Unit]

Description=Tomcat 9 Servlet Container

After=network.target

[Service]

Type=forking

Environment=\"JAVA\_HOME=/usr/lib/jvm/java-17-openjdk-amd64"

Environment=\"JAVA\_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment=\"CATALINA\_BASE=/etc/tomcat/"

Environment=\"CATALINA\_HOME=/etc/tomcat/"

Environment=\"CATALINA\_PID=/etc/tomcat/temp/tomcat.pid"

Environment=\"CATALINA\_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/etc/tomcat/bin/startup.sh

ExecStop=/etc/tomcat/bin/shutdown.sh

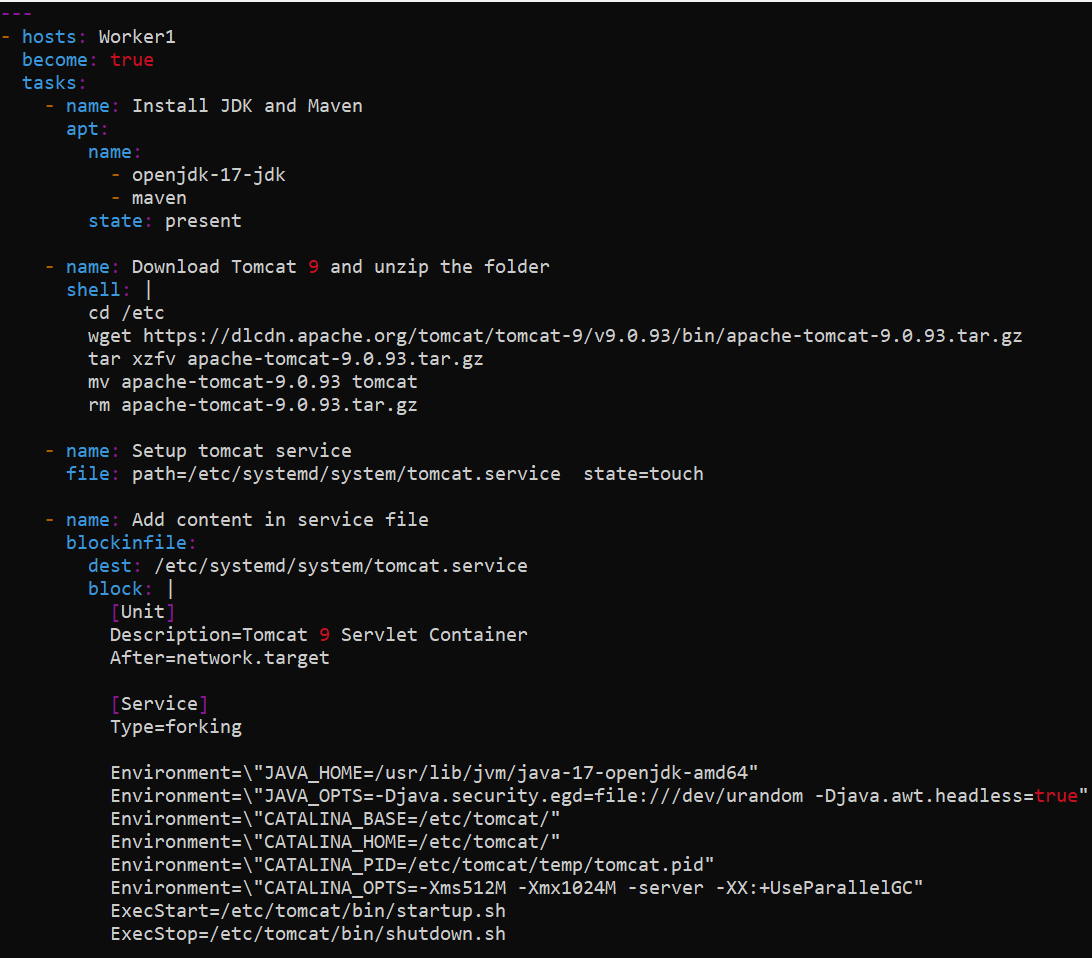
- name: Restart Services

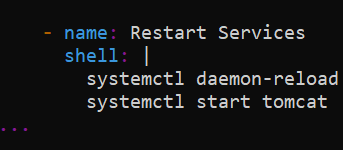
shell: |

systemctl daemon-reload

systemctl start tomcat

…

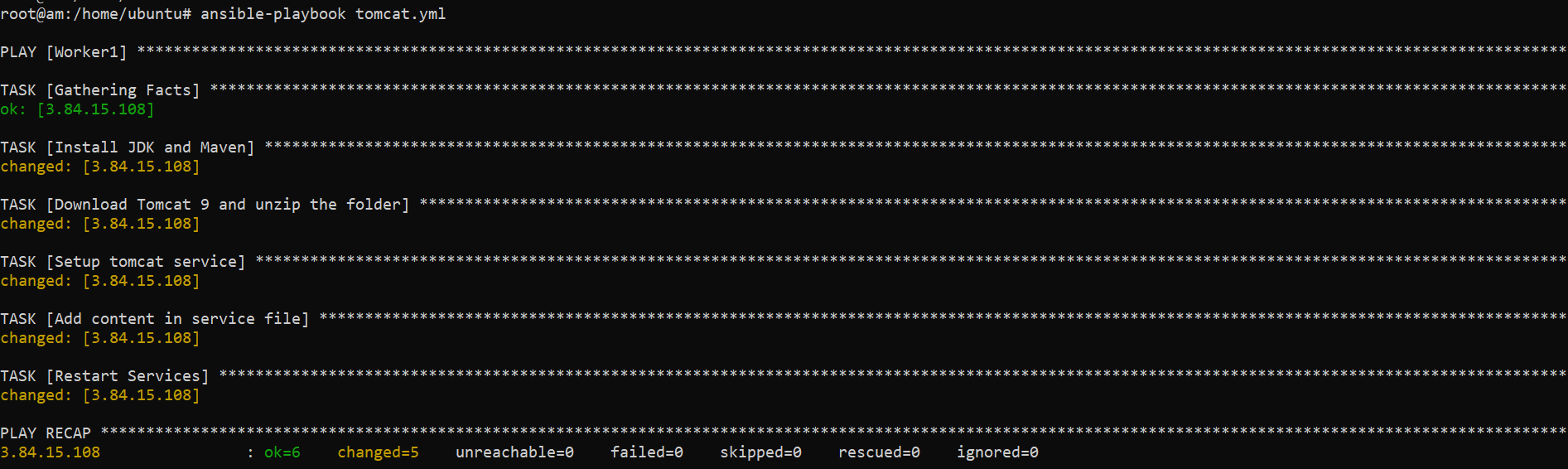
****

****

In our case, we have taken hosts as Worker 1.

Now we will run the playbook

ansible-playbook tomcat.yml

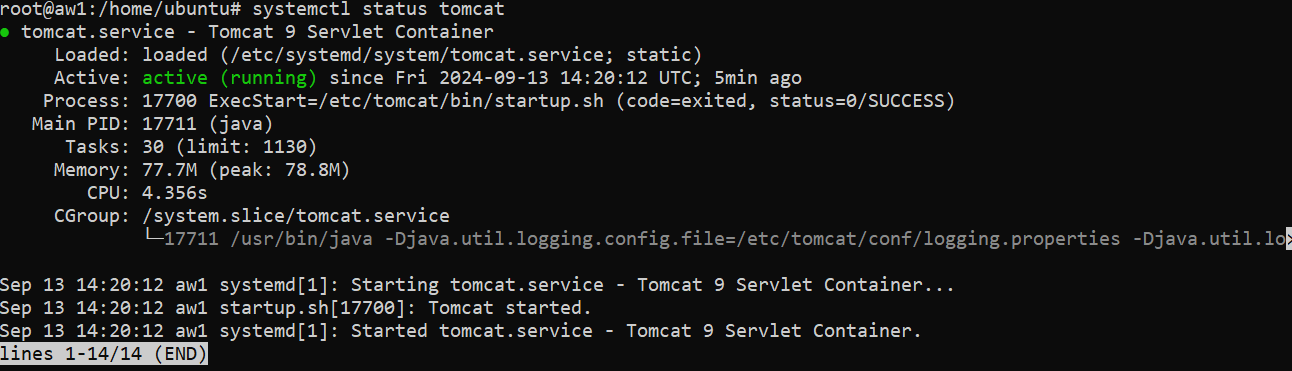


We see the playbook ran successfully.

**Output:**

**Worker1**

systemctl status tomcat



1. **Install Jenkins through ansible.**

We will create a playbook for the above task

vi jenkins.yml



Contents of the Playbook

**---**

- name: Install Jenkins and Java 17

hosts: Master

become: true

tasks:

- name: Update and upgrade apt packages

apt:

update\_cache: yes

upgrade: dist

- name: Install OpenJDK 17

apt:

name: openjdk-17-jdk

state: present

- name: Check Java version

command: java --version

register: java\_version

- name: Print Java version

debug:

msg: "{{ java\_version.stdout }}"

- name: Add Jenkins repository key

get\_url:

url: https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

dest: /usr/share/keyrings/jenkins-keyring.asc

- name: Add Jenkins repository

apt\_repository:

repo: deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/

state: present

- name: Update apt cache again after adding Jenkins repository

apt:

update\_cache: yes

- name: Install Jenkins

apt:

name: jenkins

state: present

- name: Start Jenkins service

systemd:

name: jenkins

state: started

enabled: yes

- name: Ensure Jenkins on port 8080

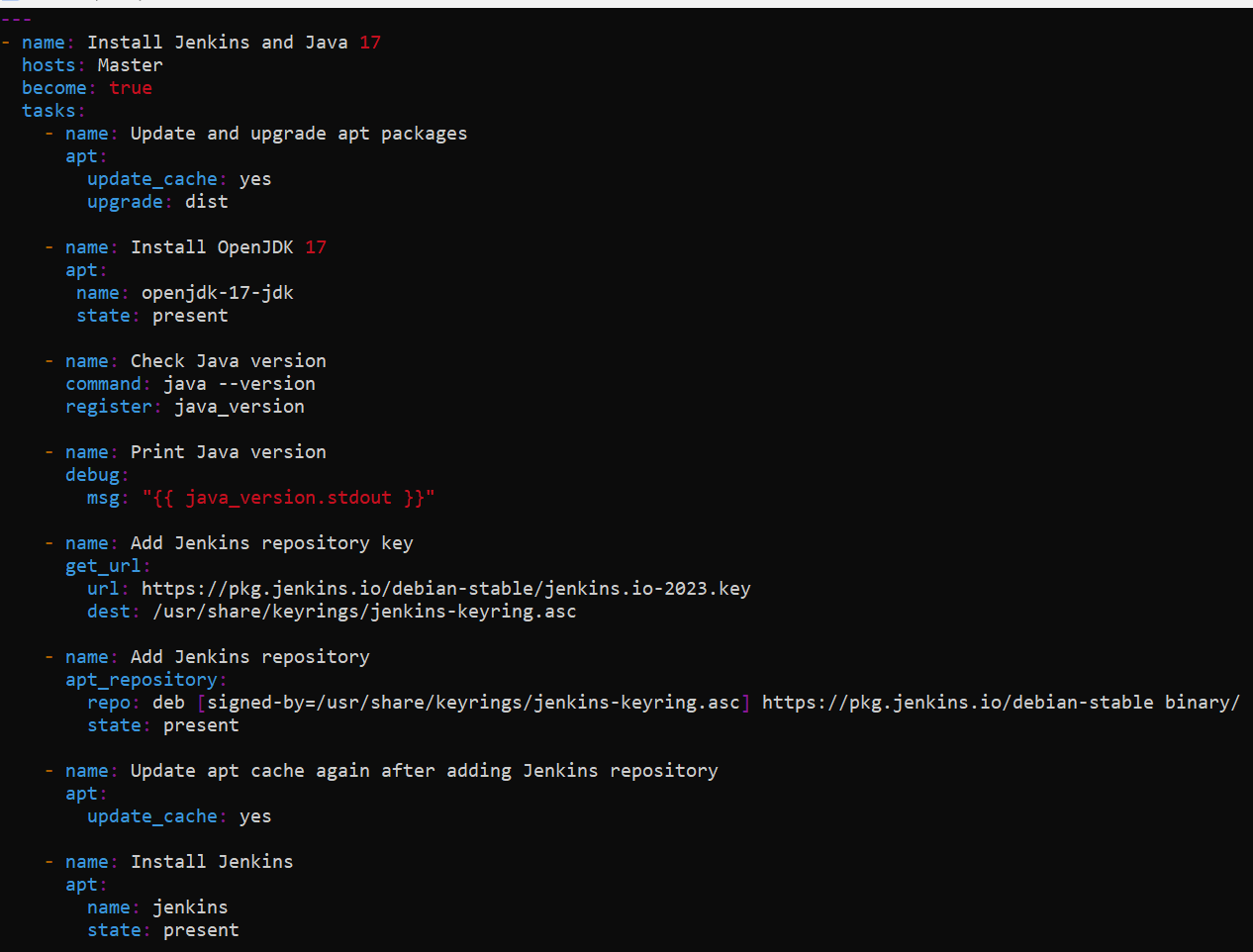
wait\_for:

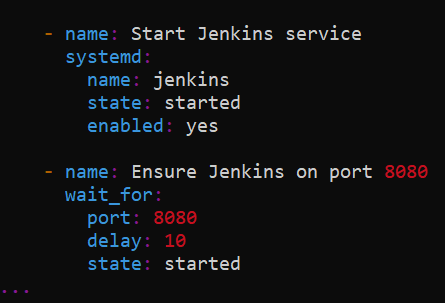
port: 8080

delay: 10

state: started

...

****



In our case, we have taken hosts as Master.

Now we will run the playbook

ansible-playbook jenkins.yml

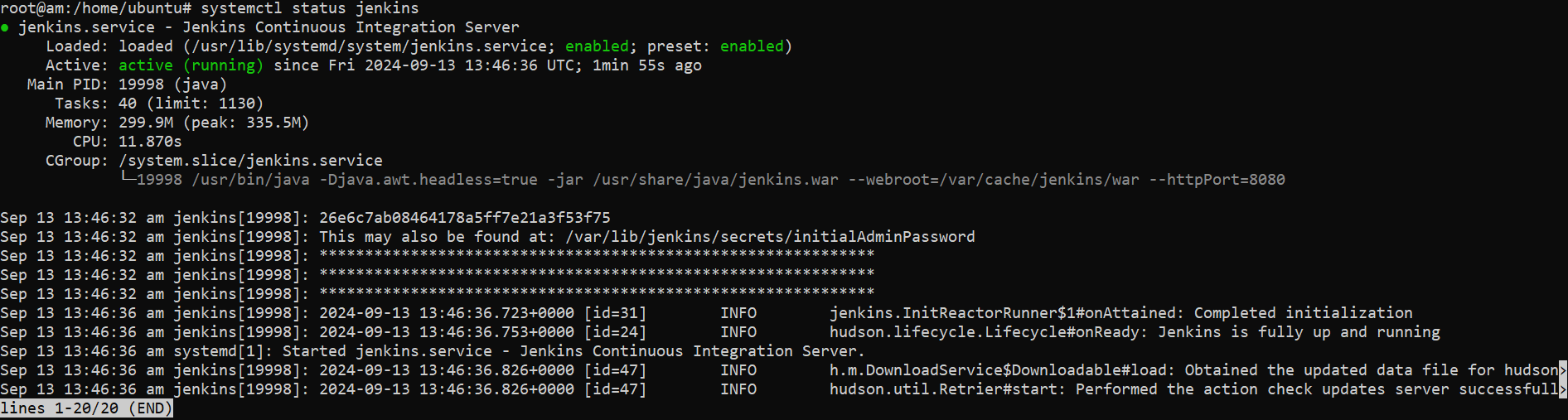


We see the playbook ran successfully.

**Output:**

**Master**

systemctl status jenkins

****

1. **Install Docker through ansible.**

We will create a playbook for the above task

vi docker.yml



Contents of the Playbook

---

- hosts: Worker1

become: true

tasks:

- name: Install aptitude

apt:

name: aptitude

state: latest

update\_cache: true

- name: Install required system packages

apt:

pkg:

- apt-transport-https

- ca-certificates

- curl

- software-properties-common

- python3-pip

- virtualenv

- python3-setuptools

state: latest

update\_cache: true

- name: Add Docker GPG apt Key

apt\_key:

url: https://download.docker.com/linux/ubuntu/gpg

state: present

- name: Add Docker Repository

apt\_repository:

repo: deb https://download.docker.com/linux/ubuntu focal stable

state: present

- name: Update apt and install docker-ce

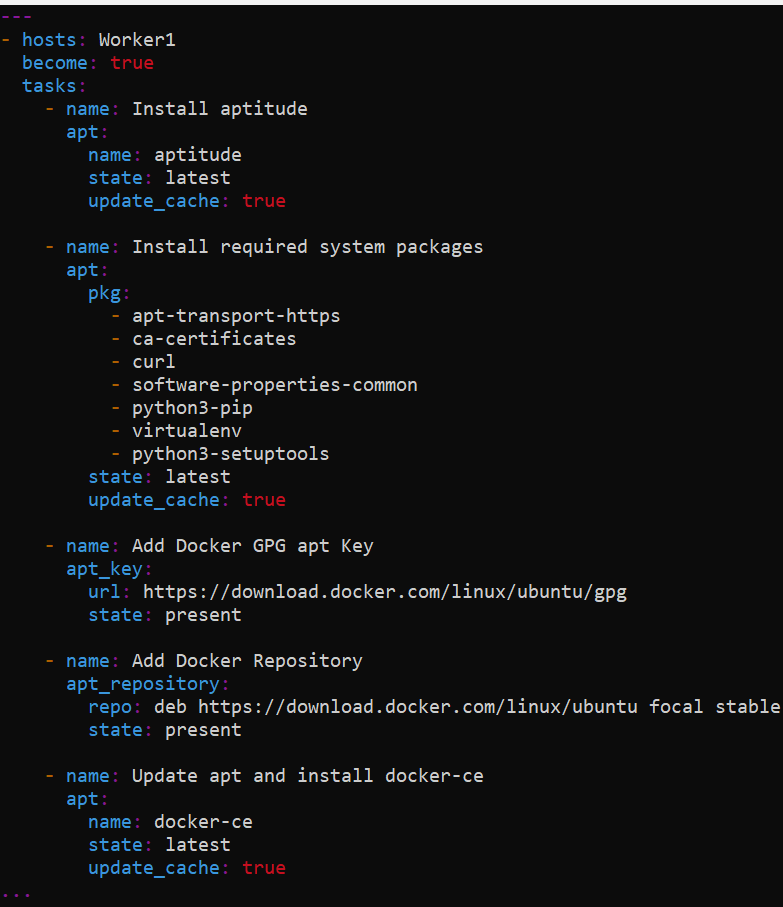
apt:

name: docker-ce

state: latest

update\_cache: true

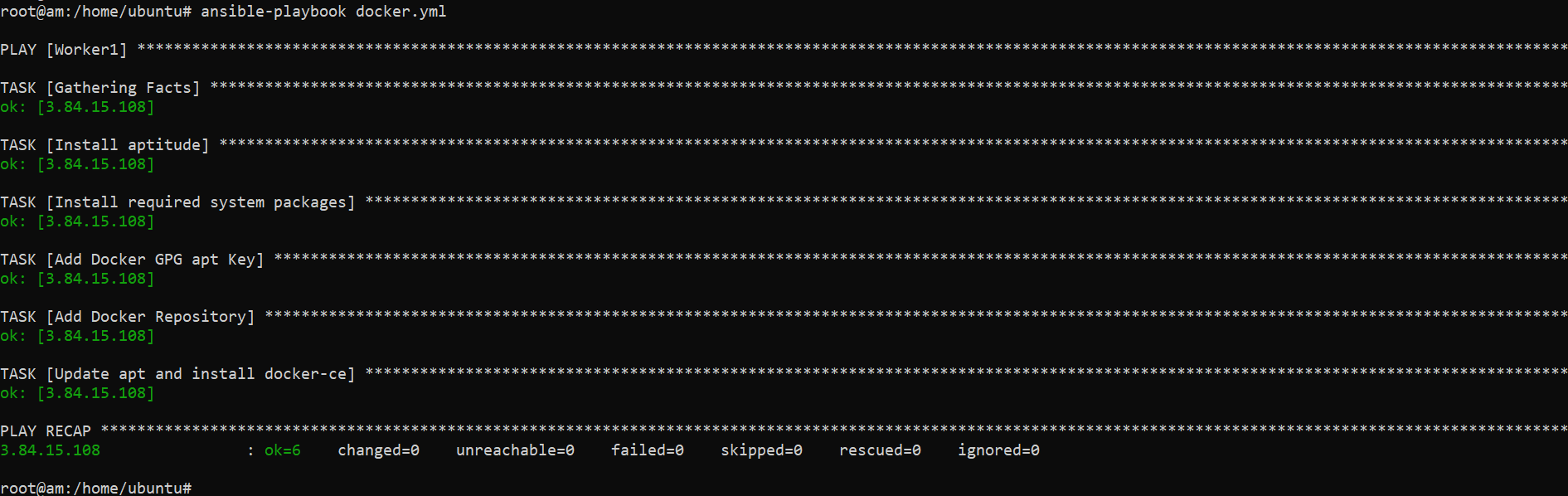
...



In our case, we have taken hosts as Worker1.

Now we will run the playbook

ansible-playbook docker.yml

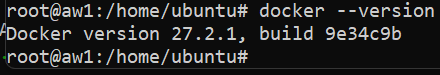


We see the playbook ran successfully.

**Output:**

**Worker1**

docker --version

****

1. **Install MySQL through ansible.**

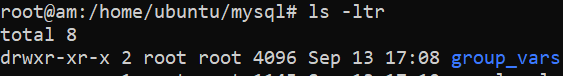
For the above task we need to create a directory. In our case, it will “mysql”. This is created to keep our credentials for my sql in it under subdirectory called group\_vars and there will be a file named as “all” which will have our credentials for mysql server.

mkdir mysql



mkdir mysql/group\_vars (directory which has the credentials)

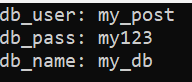
ls -ltr



vi mysql/group\_vars/all

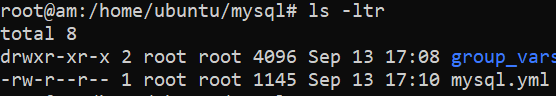


Our credentials



We will create a playbook for the above task in the parent directory

ls -ltr



vi mysql.yml



Contents of the Playbook

- name: setup Mysql with my\_db db and remote login

become: yes

hosts: Worker3

tasks:

- name: Installing Mysql and dependencies

package:

name: "{{item}}"

state: present

update\_cache: yes

loop:

- mysql-server

- mysql-client

- python3-mysqldb

- libmysqlclient-dev

become: yes

- name: start and enable mysql service

service:

name: mysql

state: started

enabled: yes

- name: creating mysql user (my\_post)

mysql\_user:

name: "{{db\_user}}"

password: "{{db\_pass}}"

priv: '\*.\*:ALL'

host: '%'

state: present

- name: creating medium\_db

mysql\_db:

name: "{{db\_name}}"

state: present

- name: Enable remote login to mysql

lineinfile:

path: /etc/mysql/mysql.conf.d/mysqld.cnf

regexp: '^bind-address'

line: 'bind-address = 0.0.0.0'

backup: yes

notify:

- Restart mysql

handlers:

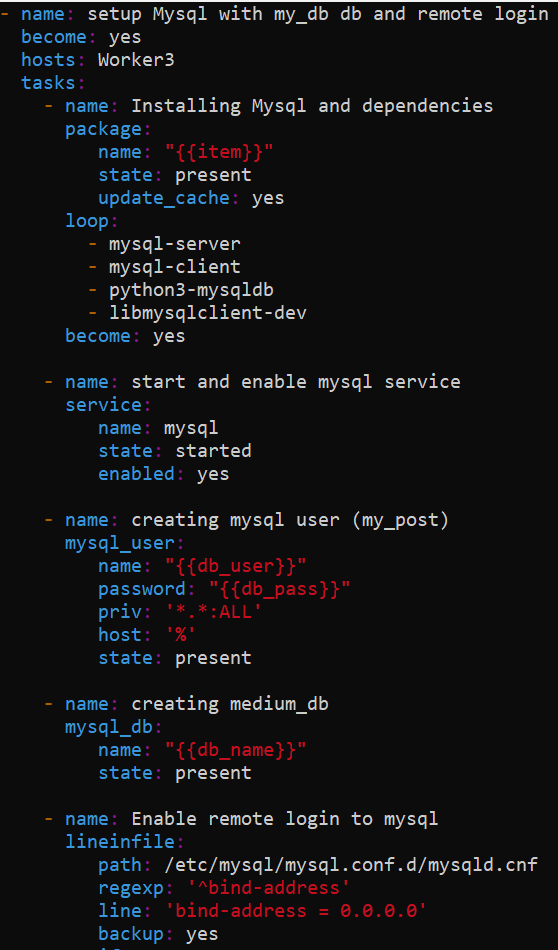
- name: Restart mysql

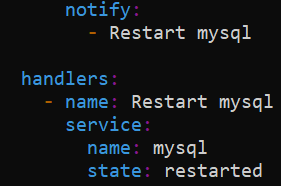
service:

name: mysql

state: restarted

...

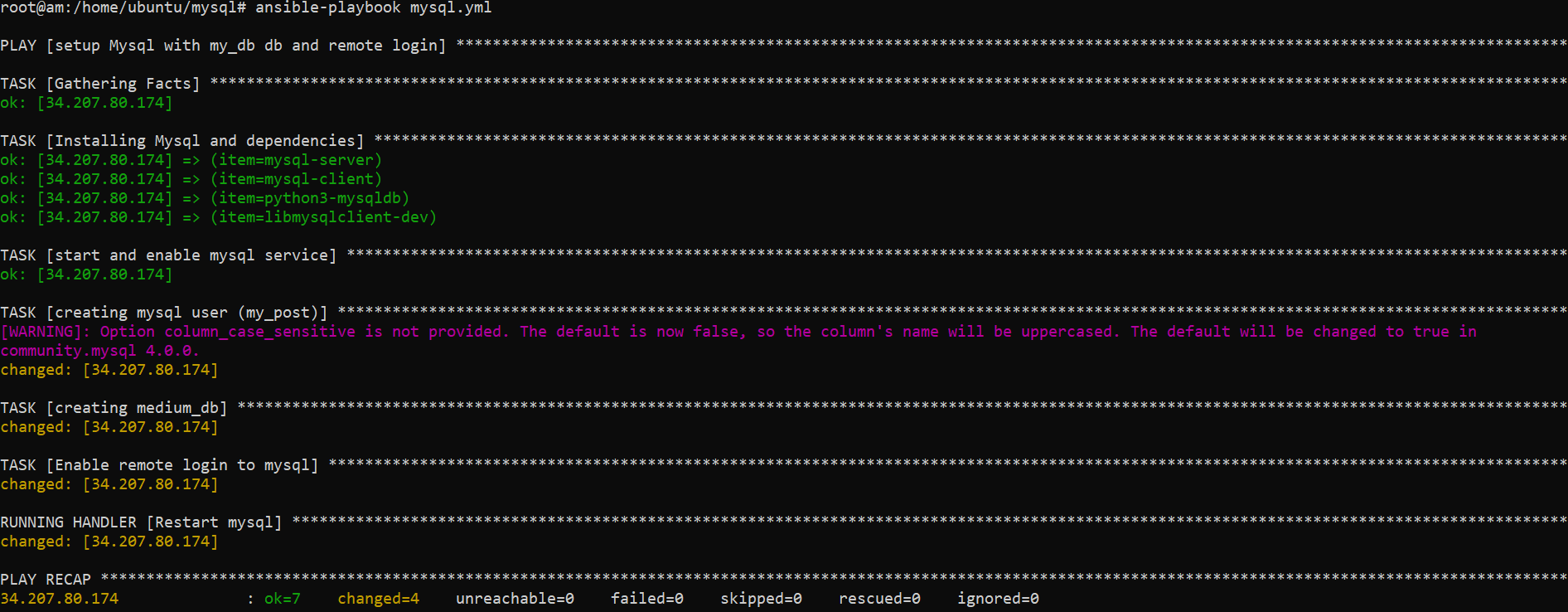




In our case, we have taken hosts as Worker3.

Now we will run the playbook

ansible-playbook mysql.yml



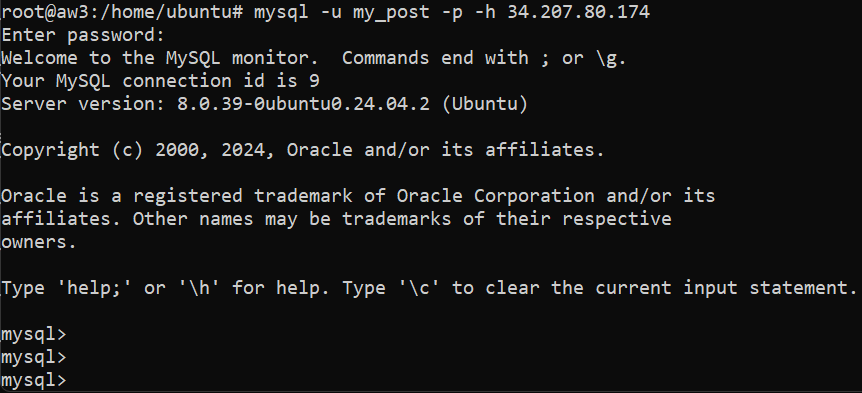
We see the playbook ran successfully.

**Output:**

**Worker3**

We need run the below command on worker3 as it has mysql client installed on it to connect to our database using the public IP address our EC2 instance.

mysql -u my\_post -p -h 34.207.80.174

****