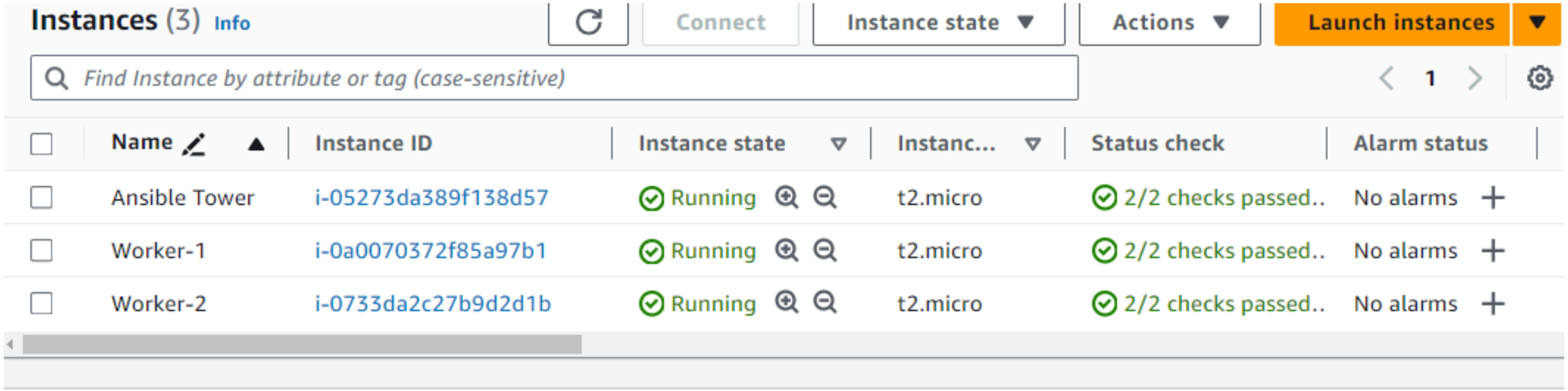
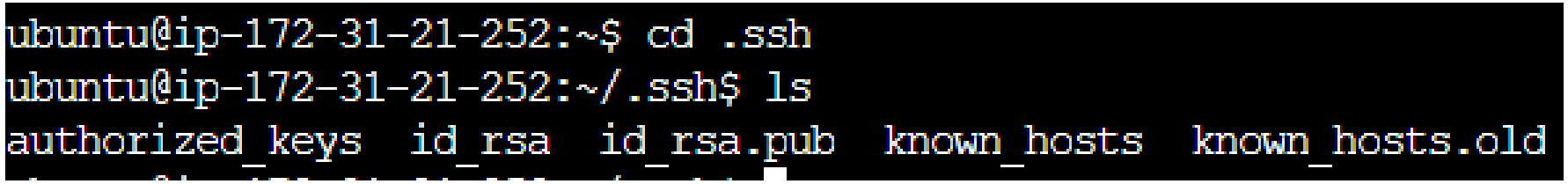
Ansible case study

First we need to setup two management server groups. One for Apache and another one for Nginx server



Connect to Ansible tower Ec2 instance
run Command :- ssh-keygen –t ras
(The ssh-keygen command is used to generate SSH key pairs, which consist of a public key and a private key.)

After run “ssh-keygen –t ras ” this command generate

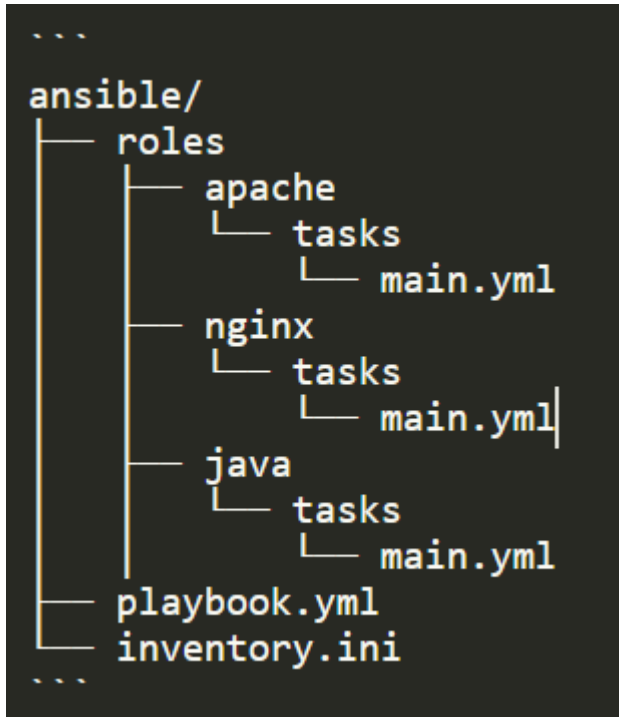


This “id_rsa.pub” copy the content inside the key and paste in “authorized_keys” of Worker node.

A detailed solution using Ansible roles to achieve the tasks outlined for configuring Apache and Nginx server groups and installing Java on the Apache server group.

1. Directory Structure:

First, let's set up the directory structure for Ansible roles and playbook:



/etc/ansible

Playbook for install apache

```
ansible/
├── roles
│   ├── apache
│   │   └── tasks
│   │       └── main.yml
```

```yaml
---
- name: Install Apache
  apt:
    name: apache2
    state: present

- name: Push Apache HTML file
  copy:
    content: "<html><body>Apache Server</body></html>"
    dest: /var/www/html/index.html

- name: Start Apache service
  systemd:
    name: apache2
    state: started

- name: Post-installation message for Apache
  debug:
    msg: "Apache installation and setup completed."
```

Playbook for install nginx

```
ansible/
├── roles
│   │   ├── nginx
│   │   └── tasks
│   │       └── main.yml
```

```yaml
---
- name: Install Nginx
  apt:
    name: nginx
    state: present

- name: Push Nginx HTML file
  copy:
    content: "<html><body>Nginx Server</body></html>"
    dest: /usr/share/nginx/html/index.html

- name: Start Nginx service
  systemd:
    name: nginx
    state: started

- name: Post-installation message for Nginx
  debug:
    msg: "Nginx installation and setup completed."
```

Playbook for install Java

```
ansible/
├── roles
│   ├── java
│   │   └── tasks
│   │       └── main.yml
```

```yaml
---
- name: Install Java
  apt:
    name: default-jdk
    state: present

- name: Post-installation message for Java
  debug:
    msg: "Java installation completed."
```

Create a Ansible inventory.ini

In Ansible, the inventory.ini file is used to define the inventory of hosts that Ansible will manage. The inventory file is a simple text file that lists the hostnames or IP addresses of the target servers or devices you want to manage with Ansible. You can also define groups of hosts, set variables for hosts or groups, and more in the inventory file.

```
ubuntu@ip-172-31-21-252:/etc/ansible$ cat inventory.ini
[apache_servers]
apache_server1 ansible_host=54.163.75.153

[nginx_servers]
nginx_server1 ansible_host=54.82.80.108
```

This Ansible playbook is a YAML file that defines a series of tasks to be executed on different groups of servers.

```
ubuntu@ip-172-31-21-252:/etc/ansible$ cat playbook.yml
---
- name: Configure Apache and Nginx server groups
  hosts: apache_servers
  become: yes
  roles:
    - apache

- name: Configure Nginx server group
  hosts: nginx_servers
  become: yes
  roles:
    - nginx

- name: Install Java on Apache server group
  hosts: apache_servers
  become: yes
  roles:
    - java
```

Run this command to execute a automation on “**ansible-playbook -i inventory.ini playbook.yml**”

The `Ansible-playbook` command is used to execute Ansible playbooks. In your command:

- `-i inventory.ini`: This specifies the inventory file (`inventory.ini`) that contains information about the hosts and groups you want to target with your playbook.

- `playbook.yml`: This is the Ansible playbook you want to execute. The playbook contains a series of tasks and configurations that will be applied to the hosts defined in the inventory.

When you run this command, Ansible will use the inventory file to identify which hosts to target and then execute the tasks defined in the playbook on those hosts. This allows you to automate various configuration and management tasks on remote servers.

```
TASK [apache : Post-installation message for Apache] ********
*
ok: [apache_server1] => {
    "msg": "Apache installation and setup completed."
}
```

```
TASK [java : Post-installation message for Java]
*
ok: [apache_server1] => {
    "msg": "Java installation completed."
}
```

```
TASK [nginx : Post-installation message for Nginx] ****
*
ok: [nginx_server1] => {
    "msg": "Nginx installation and setup completed."
}
```

```
*
apache_server1             : ok=8    changed=3
nginx_server1              : ok=5    changed=2
```

Nginx run on Worker node 1

```
Worker1 $systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-10-19 03:42:43 UTC; 1h 14min ago
       Docs: man:nginx(8)
    Process: 11817 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 11829 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 11933 (nginx)
      Tasks: 2 (limit: 1121)
     Memory: 4.7M
        CPU: 25ms
     CGroup: /system.slice/nginx.service
             ├─11933 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─11936 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
```

Apache run on Worker node 2

```
Worker 2 $sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2023-10-19 03:42:29 UTC; 1h 12min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 3542 (apache2)
      Tasks: 55 (limit: 1121)
     Memory: 5.3M
        CPU: 244ms
     CGroup: /system.slice/apache2.service
             ├─3542 /usr/sbin/apache2 -k start
             ├─3544 /usr/sbin/apache2 -k start
             └─3545 /usr/sbin/apache2 -k start
```

Java install on Apache server

```
Worker 2 $java -version
openjdk version "11.0.20.1" 2023-08-24
OpenJDK Runtime Environment (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04)
OpenJDK 64-Bit Server VM (build 11.0.20.1+1-post-Ubuntu-0ubuntu122.04, mixed mode, sharing)
```

Thanks you….