**3. Running Ad-Hoc Commands**

b. Write a document that details how to run ad-hoc commands using Ansible, including examples for managing instances in AWS, Azure, and GCP.

**Running Ad-Hoc Commands with Ansible for Cloud Instances**

**Introduction**

Ansible ad-hoc commands provide a quick way to execute tasks on remote systems without writing a playbook. This document outlines how to run ad-hoc commands for managing cloud instances in AWS, Azure, and GCP.

**Prerequisites**

Before executing ad-hoc commands, ensure you have:

* Ansible installed on your control node.
* Cloud provider credentials configured (AWS, Azure, GCP).
* The appropriate Ansible collections and modules installed.
* SSH access to the instances or cloud SDK authentication.

**General Syntax**

Ansible ad-hoc commands follow this structure:

ansible <host-pattern> -m <module> -a "<module-arguments>" -i <inventory>

* <host-pattern>: Target hosts or group.
* -m <module>: Ansible module to use.
* -a "<module-arguments>": Module arguments.
* -i <inventory>: Inventory file or dynamic inventory.

**Running Ad-Hoc Commands on AWS**

**1. List Running EC2 Instances**

ansible localhost -m amazon.aws.ec2\_instance\_info

**2. Start an EC2 Instance**

ansible localhost -m amazon.aws.ec2\_instance -a "instance\_ids=i-1234567890abcdef state=running"

**3. Stop an EC2 Instance**

ansible localhost -m amazon.aws.ec2\_instance -a "instance\_ids=i-1234567890abcdef state=stopped"

**Running Ad-Hoc Commands on Azure**

**1. List Virtual Machines**

ansible localhost -m azure.azcollection.azure\_rm\_virtualmachine\_info

**2. Start an Azure VM**

ansible localhost -m azure.azcollection.azure\_rm\_virtualmachine -a "name=myVM resource\_group=myResourceGroup state=running"

**3. Stop an Azure VM**

ansible localhost -m azure.azcollection.azure\_rm\_virtualmachine -a "name=myVM resource\_group=myResourceGroup state=stopped"

**Running Ad-Hoc Commands on GCP**

**1. List GCP Instances**

ansible localhost -m google.cloud.gcp\_compute\_instance\_info -a "auth\_kind=serviceaccount project=my-project"

**2. Start a GCP VM**

ansible localhost -m google.cloud.gcp\_compute\_instance -a "name=my-instance zone=us-central1-a project=my-project state=running"

**3. Stop a GCP VM**

ansible localhost -m google.cloud.gcp\_compute\_instance -a "name=my-instance zone=us-central1-a project=my-project state=stopped"

**Conclusion**

Ansible ad-hoc commands offer a quick and powerful way to manage cloud resources across multiple providers. By leveraging the appropriate modules, users can automate cloud instance management efficiently.

c. Include commands for tasks such as checking the status of a service and gathering facts from the managed nodes.

---

- name: Check service status

hosts: all

tasks:

- name: Check if the Apache service is running

service:

name: apache2

state: started