Terraform Associate

Study Materials

1. Exam Details
2. Exam objectives
3. Learning path
4. Video Course
5. Alternate Video Course
6. Practice Exam

1. Understand Infrastructure as Code (IaC) Concepts

1a. Explain what IaC is

- Infrastructure as Code (IaC) is the practice of managing and provisioning computing infrastructure through machine-readable configuration files, rather than physical hardware configuration or interactive configuration tools.

1b. Describe the advantages of IaC patterns

- IaC provides consistency, reusability, and enables automation by allowing version control, reduced configuration drift, and easier disaster recovery.

2. Understand the Purpose of Terraform (vs Other IaC Tools)

2a. Explain multi-cloud and provider-agnostic benefits

- Means that you can use any cloud provider you want.
- You can also use non cloud resources and manage them
- Means you don't have to code thing individually

2b. Explain the benefits of state in Terraform

- Effencicy
- Can view what is created/destroyed

3. Understand Terraform Basics

3a. Install and version Terraform providers

- Terraform
  {
  Required providers
  {

3b. Describe plugin-based architecture

- It is all the different providers

3c. Write Terraform configuration using multiple providers

-

3d. Describe how Terraform finds and fetches providers

- Uses plugins to fetch API

4. Use Terraform Outside the Core Workflow

4a. Describe when to use terraform import

- When there is current infrastructure not created with terraform
- Create the resource block
- Then use terraform import (resource name)(resource id)

4b. Use terraform state to view Terraform state

- Terraform state show
- Terraform state list

4c. Describe when to enable verbose logging

- TF_LOG
- TF_LOG=TRACE

5. Interact with Terraform Modules

5a. Contrast and use different module source options

-

5b. Interact with module inputs and outputs

-

5c. Describe variable scope within modules/child modules

-

5d. Set module version

- Version ~>

6. Use the Core Terraform Workflow

6a. Describe Terraform workflow (Write -> Plan -> Create)

-

6b. Initialize a Terraform working directory (terraform init)

- Terraform init

6c. Validate a Terraform configuration (terraform validate)

- Terraform validate

6d. Generate and review an execution plan (terraform plan)

- Terraform plan
- Terraform plan -out
- Terraform plan -out=example

6e. Execute changes to infrastructure (terraform apply)

- Terraform apply -refresh-only
- Terraform apply
- Terraform apply -destroy
-

6f. Destroy Terraform-managed infrastructure (terraform destroy)

- Terraform destroy

6g. Apply formatting and style adjustments (terraform fmt)

- Terraform fmt

7. Implement and Maintain State

7a. Describe the default local backend

- Local backend is on the local machine.
- It is where things are stored

7b. Describe state locking

- State locking prevents multiple workspaces from using the state file at the same time
- Cannot have 2 terraform plans at the same time
-

7c. Handle backend and cloud integration authentication methods

-

7d. Differentiate remote state backend options

- s3

7e. Manage resource drift and Terraform state

- Terraform apply -refresh-only
-

7f. Describe backend block and cloud integration in configuration

-

7g. Understand secret management in state files

-

8. Read, Generate, and Modify Configuration

8a. Demonstrate use of variables and outputs

-

8b. Describe secure secret injection best practices

-

8c. Understand the use of collection and structural types

-

8d. Create and differentiate resource and data configuration

-

8e. Use resource addressing and resource parameters to connect resources

-

8f. Use HCL and Terraform functions to write configuration

-

8g. Describe built-in dependency management (order of execution)

-

9. Understand HCP Terraform Capabilities

9a. Explain how HCP Terraform helps to manage infrastructure

-

9b. Describe how HCP Terraform enables collaboration and governance

-