HashiCorp Infrastructure Automation Certification

Cloud engineers can use the Terraform Associate exam from HashiCorp to verify their basic infrastructure automation skills.

Register for the exam

Prepare for the exam

Certification

Overview Certifications

HashiCorp Certified: Terraform Associate

The Terraform Associate certification is for Cloud Engineers specializing in operations, IT, or development who know the basic concepts and skills associated with open source HashiCorp Terraform. Candidates will be best prepared for this exam if they have professional experience using Terraform in production, but performing the exam objectives in a personal demo environment may also be sufficient. This person understands which enterprise

features exist and what can and cannot be done using the open source offering. Visit our exam partner to schedule and take the exam.

Prerequisites

- Basic terminal skills
- Basic understanding of on premises and cloud architecture

Product Version Tested

Terraform 0.14 and higher.

Preparing for the Exam

The Terraform Associate exam has both a study guide and a review guide. While much of the information in these two guides are the same, they are presented differently for different uses. Use the study guide if you want to study all the exam objectives. Use the review guide if you already have Terraform experience and want to choose which objectives to review

before taking the exam. We provide sample questions so you know what to expect when taking the exam.

Exam Details

Assessment Type	Multiple choice
Format	Online proctored
Duration	1 hour
Price	$70.50 USD plus locally applicable taxes and fees
Language	English
Expiration	2 years

Exam Objectives

1	**Understand infrastructure as code (IaC) concepts**
1a	Explain what IaC is
1b	Describe advantages of IaC patterns

2	**Understand Terraform's purpose (vs other IaC)**
2a	Explain multi-cloud and provider-agnostic benefits
2b	Explain the benefits of state

3	**Understand Terraform basics**
3a	Handle Terraform and provider installation and versioning
3b	Describe plugin based architecture
3c	Demonstrate using multiple providers
3d	Describe how Terraform finds and fetches providers
3e	Explain when to use and not use provisioners and when to use `local-exec` or `remote-exec`

4	**Use the Terraform CLI (outside of core workflow)**
4a	Given a scenario: choose when to use `terraform fmt` to format code
4b	Given a scenario: choose when to use `terraform taint` to taint Terraform resources
4c	Given a scenario: choose when to use `terraform import` to import existing infrastructure into your Terraform state
4d	Given a scenario: choose when to use `terraform workspace` to create workspaces
4e	Given a scenario: choose when to use `terraform state` to view Terraform state
4f	Given a scenario: choose when to enable verbose logging and what the outcome/value is

5	**Interact with Terraform modules**
5a	Contrast module source options
5b	Interact with module inputs and outputs
5c	Describe variable scope within modules/child modules
5d	Discover modules from the public Terraform Module Registry
5e	Defining module version

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

6b	Initialize a Terraform working directory (`terraform init`)
6c	Validate a Terraform configuration (`terraform validate`)

6d	Generate and review an execution plan for Terraform (`terraform plan`)
6	**Navigate Terraform workflow**
6e	Execute changes to infrastructure with Terraform (`terraform apply`)
6f	Destroy Terraform managed infrastructure (`terraform destroy`)

7	**Implement and maintain state**
7a	Describe default `local` backend
7b	Outline state locking
7c	Handle backend authentication methods
7d	Describe remote state storage mechanisms and supported standard backends
7e	Describe effect of Terraform refresh on state
7f	Describe `backend` block in configuration and best practices for partial configurations
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 practice
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 together
8f	Use Terraform built-in functions to write configuration

8g	Configure resource using a `dynamic` block
8h	Describe built-in dependency management (order of execution based)

9	**Understand Terraform Cloud and Enterprise capabilities**
9a	Describe the benefits of Sentinel, registry, and workspaces
9b	Differentiate OSS and TFE workspaces
9c	Summarize features of Terraform Cloud

CONNECT

Multi-Cloud Networking

RUN

Multi-Cloud Orchestration

PRODUCTS

Terraform

Vault

Consul

Nomad

Vagrant

Packer

Boundary

Waypoint

Sentinel

RESOURCES

Blog

Tutorials

Community

Events

Integrations

Library

Partners

Podcast

Support

Training

COMPANY

About Us

Jobs

Press Center

Brand

Contact Us

System Status Cookie Manager Terms of Use Security Privacy

Trademark Policy

stdin: is not a tty