HashiCorp Terraform

# Terraform Enterprise Onboarding Program

COBRA Team | HashiCorp Customer Success

February 2023

# Agenda

1. Welcome
2. Terraform Enterprise Onboarding Program
3. Terraform Enterprise Technical Overview
4. Next Steps

# Code of Conduct

**HashiCorp is dedicated to providing a harassment-free Terraform Enterprise Onboarding experience for everyone, regardless of gender, gender identity, sexual orientation, disability, physical appearance, body size, race, national origin, or religion. We value your attendance and do not wish anyone to feel uncomfortable or threatened at any time.**

The bottom line is that we do not tolerate harassment of conference participants in any form. Harassment includes but is not limited to offensive verbal comments related to gender, gender identity, sexual orientation, disability, physical appearance, body size, race, national origin, religion; sexual or inappropriate images in public spaces; deliberate intimidation; stalking; trolling; sustained disruption of talks or other events; and unwelcome sexual attention. Participants asked to stop any harassing behavior are expected to comply immediately. If you are being harassed, notice that someone else is being harassed, or have any other concerns, please let the HashiCorp event representative know immediately or email customer.success@hashicorp.com.

HashiCorp Terraform

# Terraform Enterprise Onboarding Program

# Pre-Onboarding Webinar

## Customer Success Overview and Support Model

Please watch the pre-recorded video included in your registration email that provides an overview of:

1. COBRA Onboarding Program
2. Support Model

HashiCorp

# Terraform Enterprise Onboarding Journey

A 7-week guided community program following a prescriptive path to successfully onboarding and adopting Terraform Enterprise

- Week 1 - Kickoff - Product & Architecture Overview
- Week 2 - Webinar - Architecture Deep Dive
- Week 3 - Webinar - Importing Resources & Migrating State
- Week 4 - Webinar - Terraform Workflows
- Week 5 - Office Hours
- Week 6 - Webinar - Terraform Governance & Integrations
- Week 7 - Webinar - Operating your Terraform Instance
- Exit Ramp and Operational Readiness Check

# Onboarding Goal

Our objective is to enable your team to successfully deploy the platform and see value within 90 days

**Terraform Enterprise Installed**

- Terraform Enterprise installed in your environment(s)
- Basic configuration completed
- Telemetry and monitoring in place
- Deployment and operational patterns established

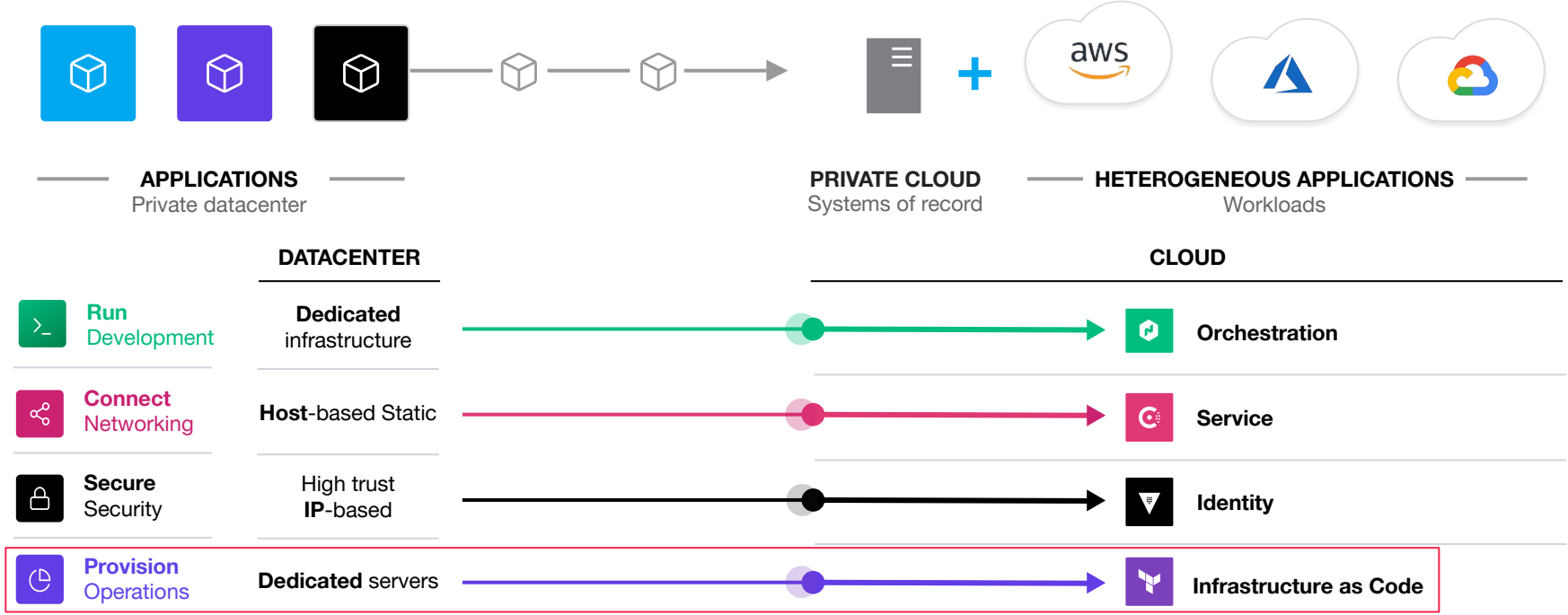
**Terraform Enterprise Operational**

- Organizations, Teams, and Users created & SAML integration in place (if being used)
- First team onboarded and consuming Terraform Enterprise
- A roadmap created for onboarding additional teams to the platform

**Completed within 90 days**

# Enabling a Common Operating Model

Standardised interfaces to cloud services, simplify and accelerate cloud adoption



# Terraform Enterprise Technical Overview

HashiCorp

# Terraform Enterprise

- Terraform Enterprise (TFE) is an Infrastructure as Code **(IaC)** system that enables users to create and manage resources on cloud platforms & other services via their APIs
- TFE uses the Hashicorp Configuration Language (HCL) & familiar languages via CDK for Terraform
- HCL should be stored in a Git repo, to be automated, versioned, and audited
- Providers enable Terraform to work with virtually any platform or service with an accessible API
- The Terraform Registry contains thousands of providers for use with Terraform

# Features

- Organizations
- SSO, Teams, Users
- API Tokens
- VCS Provider / Git Connections
- Private Module Registry
- SSH Keys
- Sentinel Policy Sets

- Workspaces
  - Tags
  - Terraform Code, Statefiles
  - Run History
  - Variables, Sensitive, ENV, Sets
  - Run Notifications, Tasks, Triggers
  - RBAC for selective Team Access
- Cloud Agents

# Organizations

- Security boundary and shared space for teams to collaborate on workspaces
- Users can belong to multiple organizations, the UI allows for easy switching between organizations

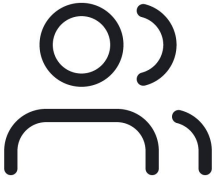
## Organization Components:

- Authentication
- Teams / Users / SSO
- Tags
- API Tokens (Org, Team, Users)
- VCS Provider / Git Connections
- Private Module Registry
- Workspaces
- Variables, ENV Variables, CLI Flags
- SSH Keys
- Sentinel Policy Sets
- Cloud Agents

# Single Sign On (SSO)

- Terraform Enterprise supports the SAML 2.0 standard
- Tested and supported IdPs include:
  - ADFS
  - Azure AD
  - Okta
  - OneLogin
- Prior to activating SAML always create a non-SSO admin account for recovery purposes
- SAML SSO Configuration Settings

# Teams & Users



- Teams are groups of users within an organization that can be assigned to workspaces within the organization
- Teams can be assigned to multiple workspaces and have different permissions in each workspace
- Teams can also be assigned organization-level permissions
- Users in Terraform Enterprise are members of Teams within Organizations
- Users do not belong to any organization or workspaces until an owner of them has added them to a team.

# API Tokens

Terraform Enterprise supports 3 types of API tokens:

- **User**: most flexible type inherit permissions from the user they are associated with
- **Team**: belong to specific team, allow access to the workspaces the team has permissions for
- **Organization**: not tied to a team or user, designed for creating and configuring workspaces and teams before delegation a workspace to that team

API tokens allow:

- Auth with TFE API
- Auth with TF remote backend for CLI runs
- Using private modules in command-line runs on local machine

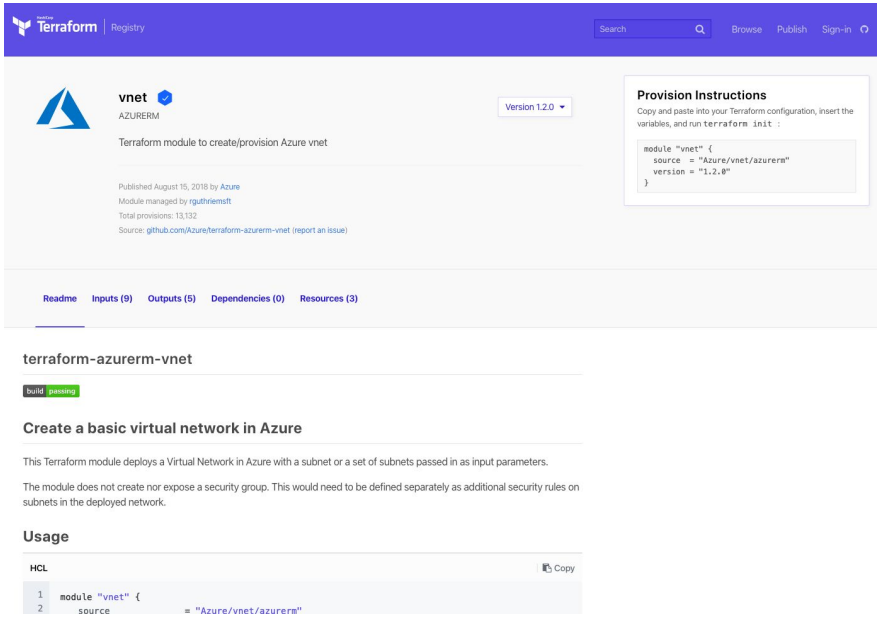
# VCS Integration

- TFE is most powerful when integrated with a VCS Provider
- TFE registers Git webhooks with Git repos to monitor for commits and pull requests
- TFE interacts with most Git providers using the API and OAuth token
- BitBucket Server & Azure DevOps server require an SSH key
- TFE supports integrating with multiple VCS providers within an Organization
- During workspace creation a configured Git provider is selected

| Supported VCS Providers |
| --- |
| GitHub |
| GitHub Enterprise |
| GitLab.com |
| GitLab EE and CE |
| BitBucket Cloud |
| BitBucket Server |
| Azure DevOps |

# Private Module Registry

- Terraform modules are a container for multiple cloud resources that are used together
- Modules can be used to create lightweight abstractions, to describe infrastructure in terms of its architecture, rather than directly in terms of specific cloud resources
- The [Private Module Registry]() (PMR) works similarly to the [public registry]() and includes support for versioning and a searchable list



# Sentinel Policy Sets

Sentinel is a framework for Policies as Code (PaC) similar to how Terraform implements Infrastructure as Code (Iac)

- Sandboxing
- Codification
- Version Control
- Automation
- Testing

CODE EDITOR

```
import "tfconfig"
import "strings"

# Require all modules directly under root module
# to come from Terraform
validate_modules_from_pmr  = func() {
 validated = true
 for tfconfig.modules as _, m {
   if not strings.has_prefix(m.source, "app.terraform.io/jrx") {
     print("Module with source", m.source, "is not in the PMR" )
     validated = false
   }
 }
 return validated
}
```

# Workspaces

**Workspaces Contain:**

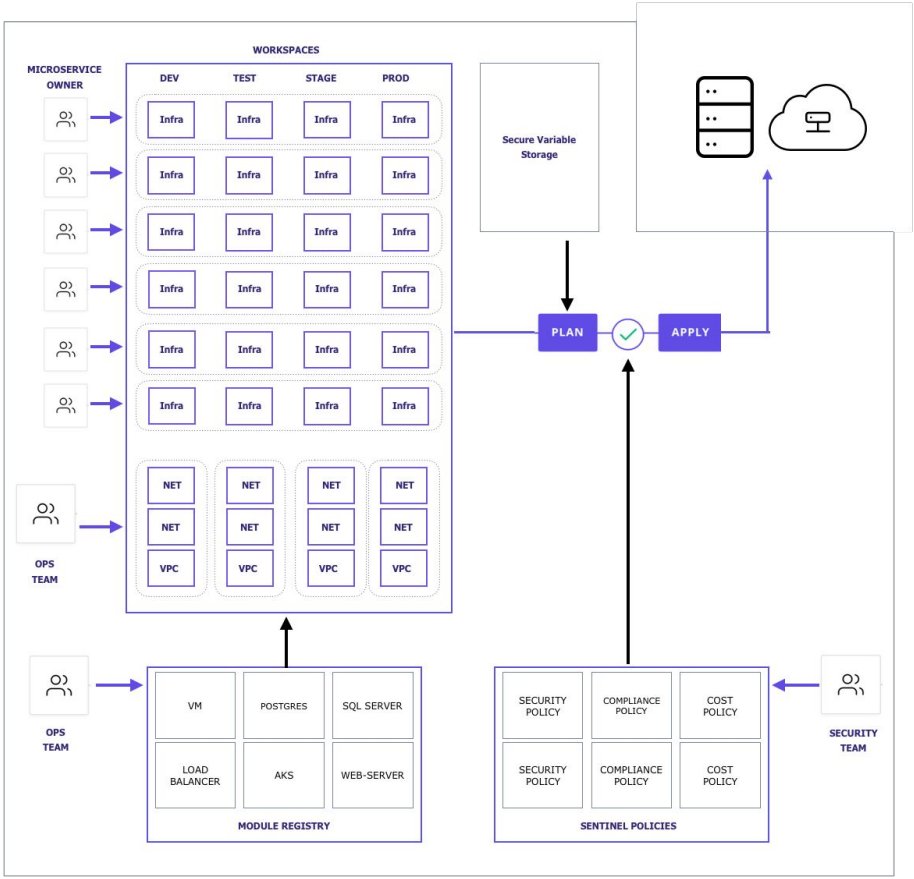
- Terraform Code, from a VCS Git Repo or uploaded as a .zip file to the API
- Variables (can be marked as Sensitive)
- Environment Variables
- Persistently stored TF statefiles for cloud resources that are managed
- Historical TF Statefiles and Run logs

**Workspaces can be run in the following ways:**

- Uploading a .zip file of TF code via the API
- Connected to a Git Repository from your VCS provider and will monitor for changes using Git Webhooks

# Workspaces

- Organize and decompose monolithic infrastructure into micro-infrastructures
- Match the organization of your application or teams with your infrastructure
- “Micro-infrastructures” are linked to create the complete infrastructure for the application

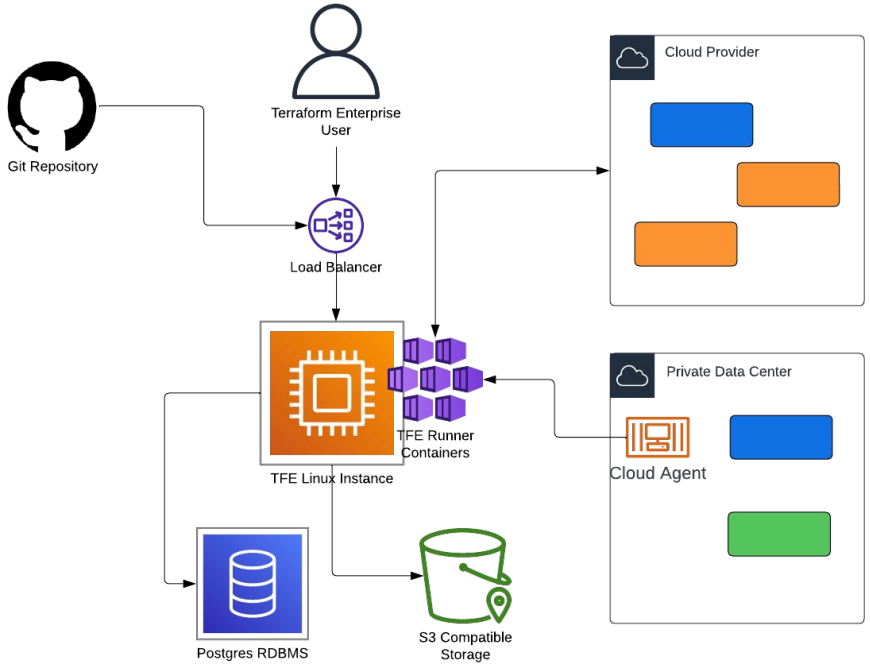


# Cloud Agents

- Terraform Cloud Agents allow TFE to communicate with isolated, private, or on-premises infrastructure
- Deployed as lightweight Docker-based agents within a specific network segment
- Useful for on-premises infrastructure types such as vSphere, Nutanix, OpenStack, enterprise networking providers, and **anything in a protected enclave**
- **The agent architecture is pull-based, so no inbound public internet connectivity is required**
- Agents poll Terraform Enterprise for work and carry out execution of that work locally

# Terraform Enterprise Architecture

- TFE is a self-managed service composed of microservices running within Docker
- TFE uses S3-compatible storage, Postgres RDBMS, Redis, and Replicated (license management)
- Remote runners called Cloud Agents are available for deployment where desired



# TFE Installation

What do we need to decide?

1

## Network Access

Network connectivity type?

- **Online**
- **Air-Gapped**

2

## Installation Location

Where will TFE be installed?

- **On-Premise Data Center**
- **Cloud Provider**

3

## Operational Mode

Which supported operational mode?

- **External Services**
- **Mounted Disk**

# 1 Network Access

How will installation be performed?

## Online

- Requires public internet access for the TFE server
- Admin executes the installer directly in a terminal session
- Installer manages all required software and outputs the dashboard URL

## Air-Gapped

- Does not utilize or require public internet access for the TFE server
- Admin installs a supported version of Docker
- Admin downloads, transports, and executes the air-gap file & installer bootstrapper

# ❷ Installation Location

Where will Terraform Enterprise be installed?

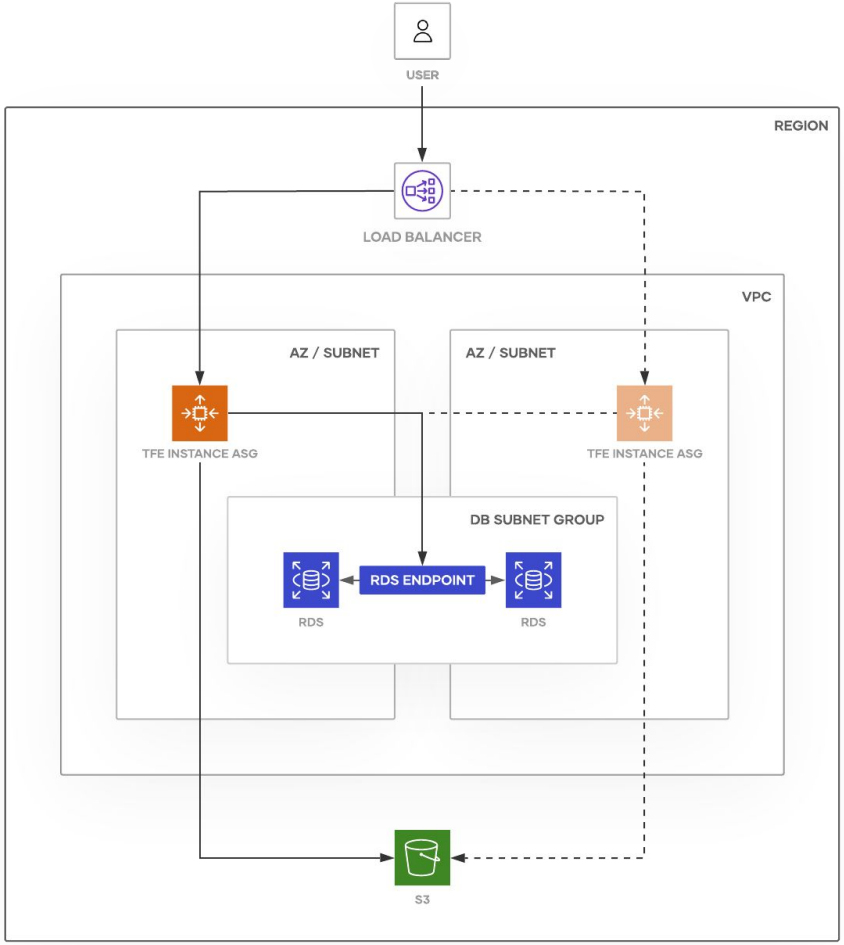
## Cloud Provider

- HashiCorp provides reference architectures for deploying onto a cloud platform
- AWS Reference Architecture
- Azure Reference Architecture
- GCP Reference Architecture

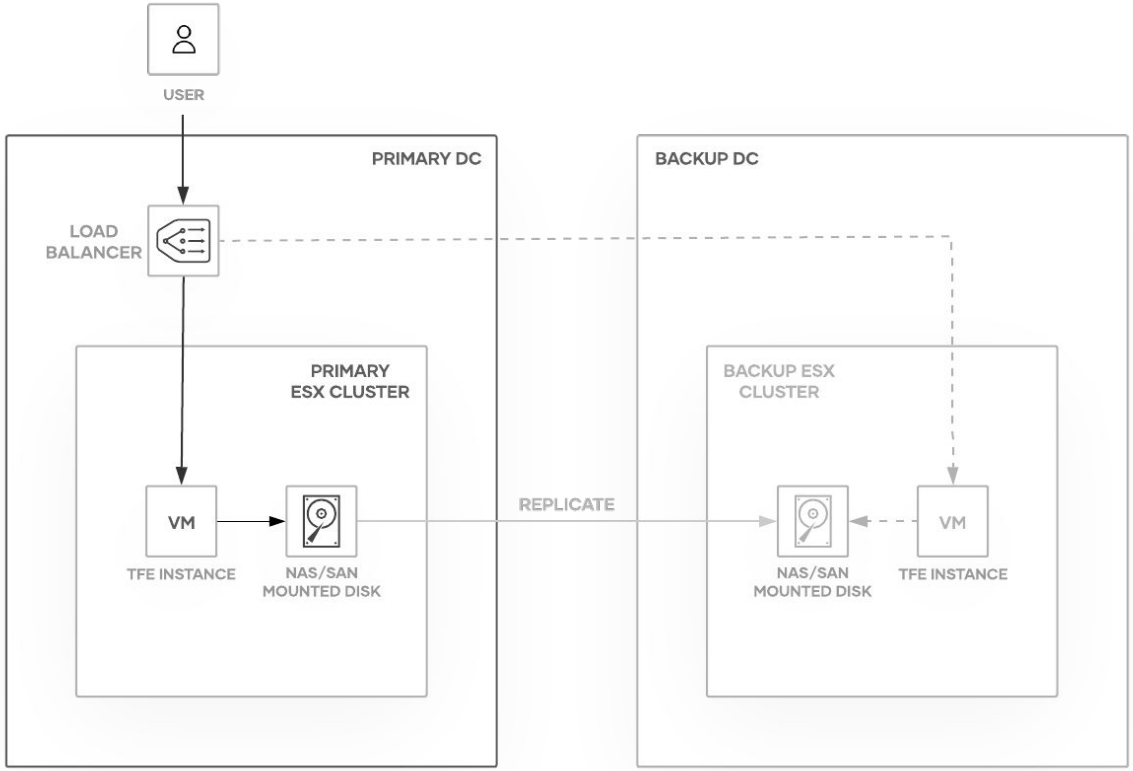
## Data-Center Deployment

- HashiCorp provides a reference architecture for deploying to VMWare
- VMware Reference Architecture

# Cloud Provider Standalone Architecture (Recommended)



# VMware Standalone Architecture (Recommended)



# 3 Operational Mode

## External Services

- High Capacity
- Needs automation to set up quickly
- Good for Production Workloads
- Uses externally running Postgres, S3 Storage, and Redis (in Active/Active)
- Required to move to Active/Active
- **Preferred** mode for Cloud installation

## Mounted Disk

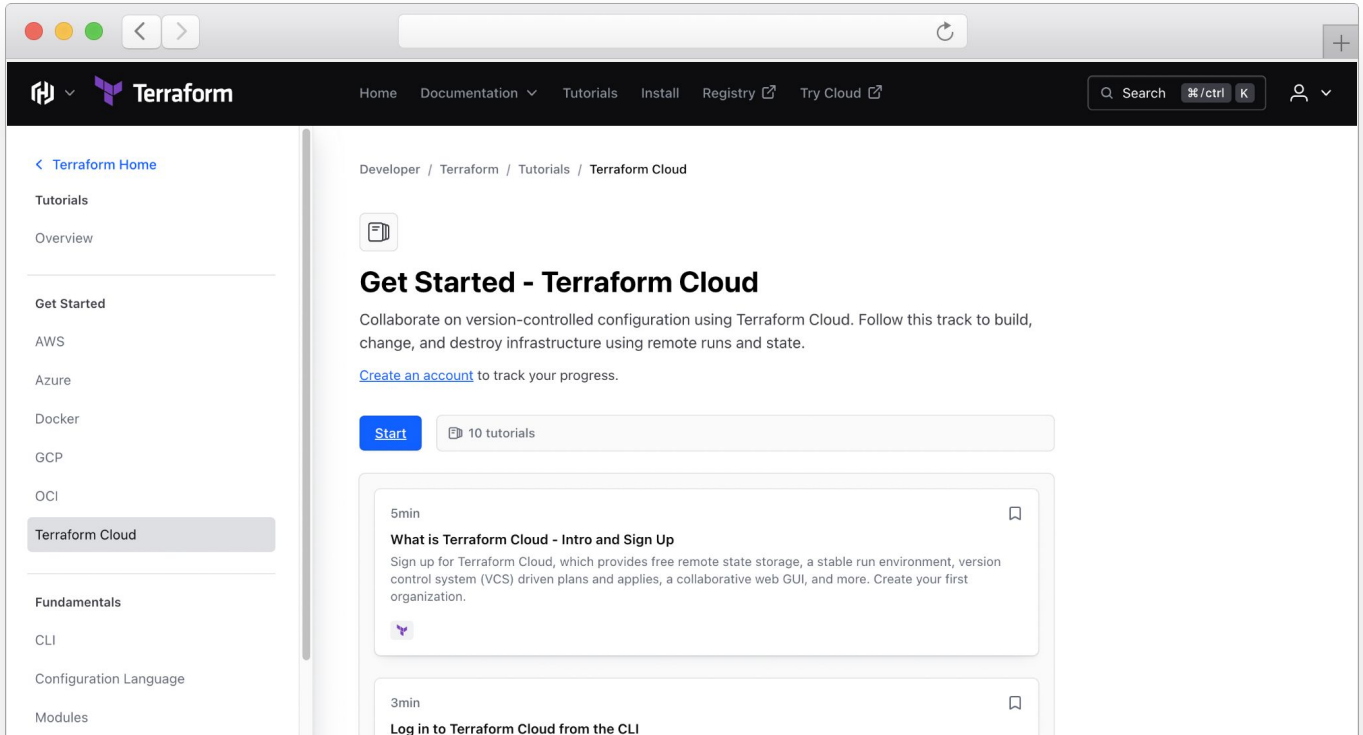
- Low Capacity
- Self-contained
- Easy to set up manually
- Good for Non-Production Workloads and Testing
- Single Docker instance for Postgres, S3 Storage, Redis

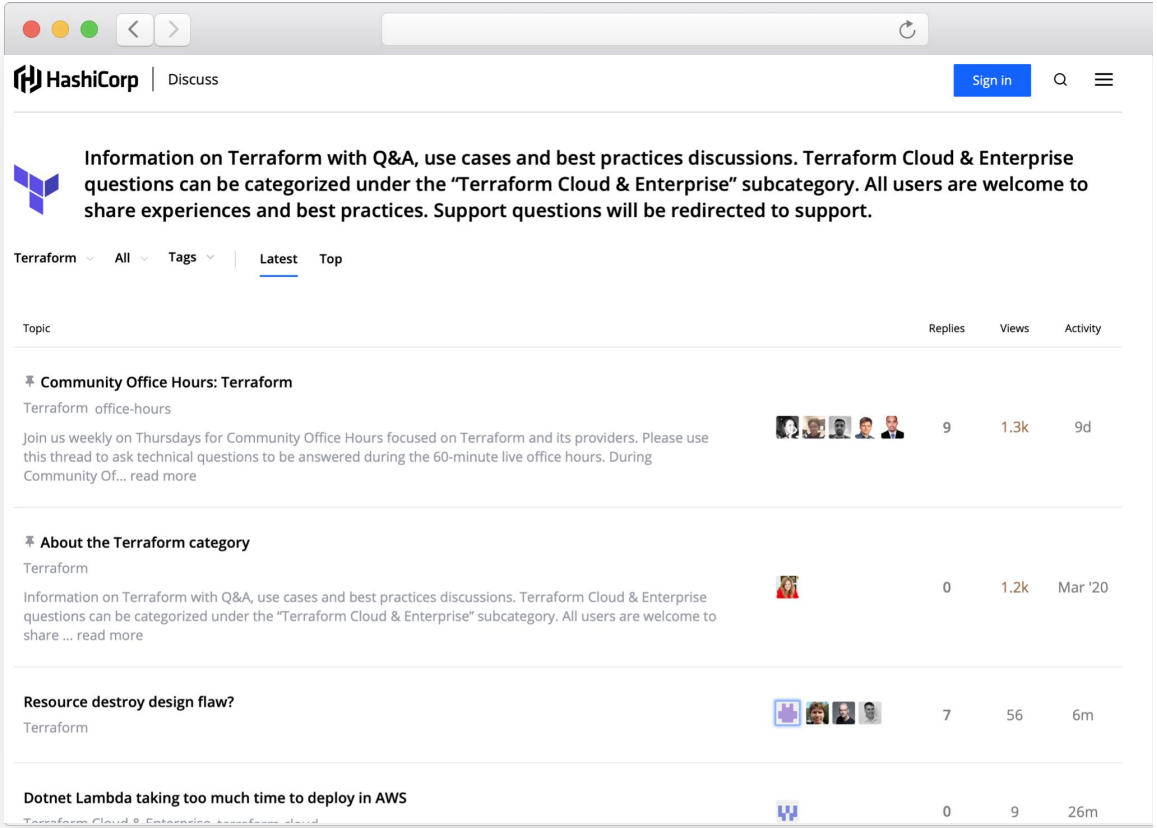
# Next Steps

# Developer

https://developer.hashicorp.com/terraform

Our new documentation platform makes it easy to learn from dozens of interactive lab environments, hundreds of tutorials, and thousands of reference docs.

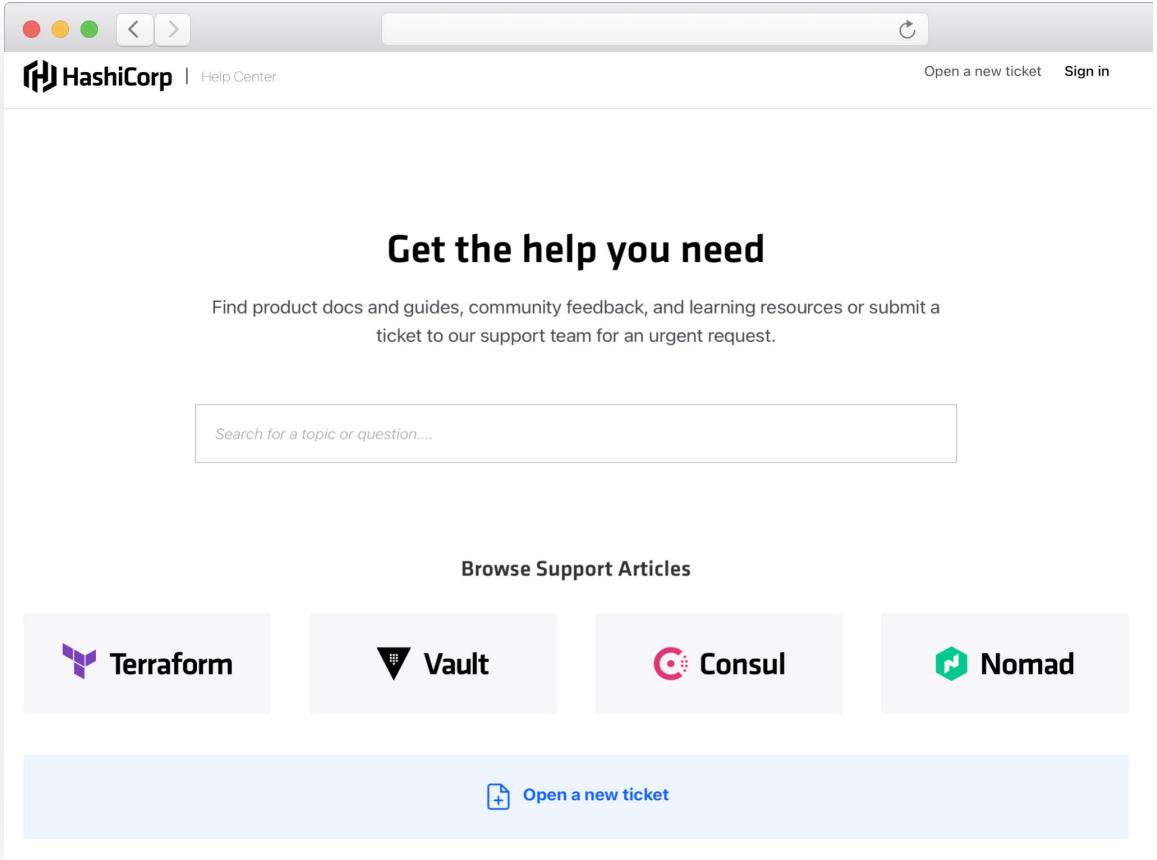





# Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers.

discuss.hashicorp.com




# Support

https://support.hashicorp.com

# Upcoming Webinars

Terraform

## Architecture Deep Dive

This webinar covers best practices for architecting and installing Terraform Enterprise

## Importing Resources & Migrating State

Topics include Terraform Basics, Importing existing infrastructure into a TFE instance, and migrating from TF OSS to TFE

## Terraform Workflow Management

Deep dive into best practices around run workflows, workspaces, variables, modules, and Git repo structure

# Action Items

- Identify your use case(s) and define your goals and project milestones with Terraform Enterprise
- Share to customer.success@hashicorp.com
  - Authorized technical contacts for support
  - Stakeholders contact information (name and email addresses)
- Gather requirements and complete 3 critical decisions:
  - Network connectivity type
  - Installation location
  - Installation mode

# Q & A

# Thank You

customer.success@hashicorp.com
www.hashicorp.com/customer-success