

# Terraform Enterprise Architecture Deep Dive

*Stand Alone*

February 2023

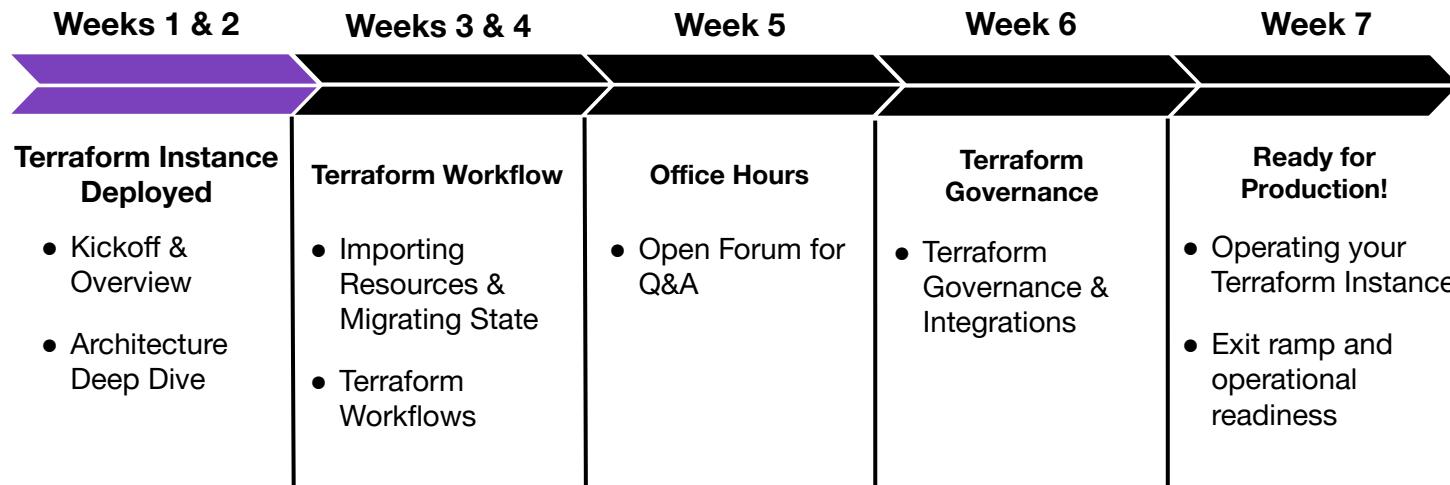


---

# Agenda

1. Deployment
2. Operational Modes
3. Reference Architecture
4. Dependencies
5. Installation Patterns
6. Configuration

# Terraform Enterprise Path to Production



— 01

# Deployment





# Pre-Install Checklist

Critical tasks than need completed prior to starting Terraform Enterprise installation include:

- Establish the [Operational Mode](#) - cannot be changed once TFE is operating\*
- Obtain [Credentials](#)
- Prepare [data storage](#)
- Configure [Linux](#) instance(s)

\*Customers planning to implement Active/Active are advised to initially operate in External Service mode and then migrate



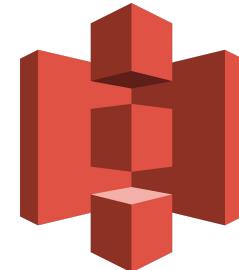
# Credentials

- License File
- TLS Certificate & Private Key
  - Required for the instance to operate
  - Certificate must match Terraform Enterprise's hostname
  - Certificate **must** be trusted by all services TFE interfaces with
  - Key and X.509 must be PEM (base64) encoded
- Correct permissions on object storage



# Data Storage

- S3-compatible object storage is used to store state files (S3, GCP Cloud Storage, Azure blob)
- Bucket must be created prior to installation and is specified during the installation process
- Terraform Enterprise expects to manage all data in the object storage service
- Disable any lifecycle rules that would delete, archive, or transition objects in this bucket

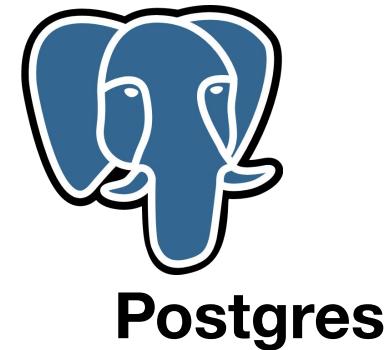


**S3 Storage**



# Data Storage

- A PostgreSQL server (Amazon RDS) or a PostgreSQL-compatible server is required
- TFE uses Postgres to store:
  - SSO Configuration
  - Terraform Organizations
  - Workspace Configuration
- Supported PostgreSQL server versions:
  - 12.x, 13.x or 14.x
  - 11.x will be deprecated in v202303-1 release
- [Postgres Requirements](#)



# Replicated



REPLICATED

- Manages licensing of Terraform Enterprise
- Used as a scheduler to manage Docker containers
- Customers should not interact with Replicated directly unless directed by support

# Terraform - Services



- **tfe\_nginx** - Nginx reverse proxy, facilitates access to TFE services
- **tfe\_atlas** - The API and Web UI, TFE used to be known as Atlas
- **tfe\_build\_manager** - Manages the queue of Terraform runs
- **tfe\_build\_worker** - Creates workers on-demand as required by the queue, injects variables, secrets, and Terraform configuration to a temporary container
- **tfe\_worker** - The ephemeral container which executes a Terraform `plan` or `apply`, can be replaced with a custom image, may be created with a randomly generated name
- [List of Terraform Enterprise Containers](#)

— 02

# Operational Modes



---

# Operational Modes

- Mounted Disk
- External Services
- Active/Active



---

# Mounted Disk

Production supported  
(Recommended for VMWare)

- Good for Production workloads
- Long-Lived
- Single Region
- Self-contained
- Easy to set up manually
- High Availability (HA) / Disaster Recovery (DR) with cold standby
- Single Docker instance for Postgres, S3 Storage



---

# Standalone External Services

Production supported  
*(Recommended for Cloud Deployments)*

- Good for high Capacity Production Workloads
- Single Region
- Best done via automation for quick set up
- Uses externally running Postgres & S3 Storage
- High Availability (HA) / Disaster Recovery (DR) with cold standby



---

# Active/Active

- High Availability (HA)
- Disaster Recovery (DR)
- Horizontal Scaling
- Single Region
- Requires automation for setup
- Increased operational complexity
- Uses 2X - 5X the resources of a Standalone installation
- Uses externally running Postgres, S3 Storage, and Redis

— 03

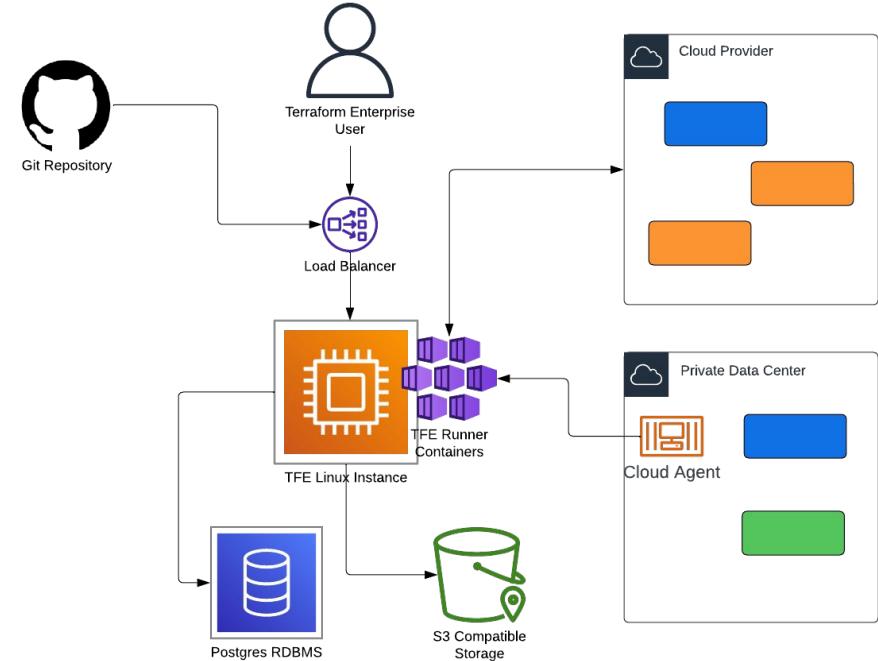
# Reference Architecture



# Terraform Enterprise Architecture



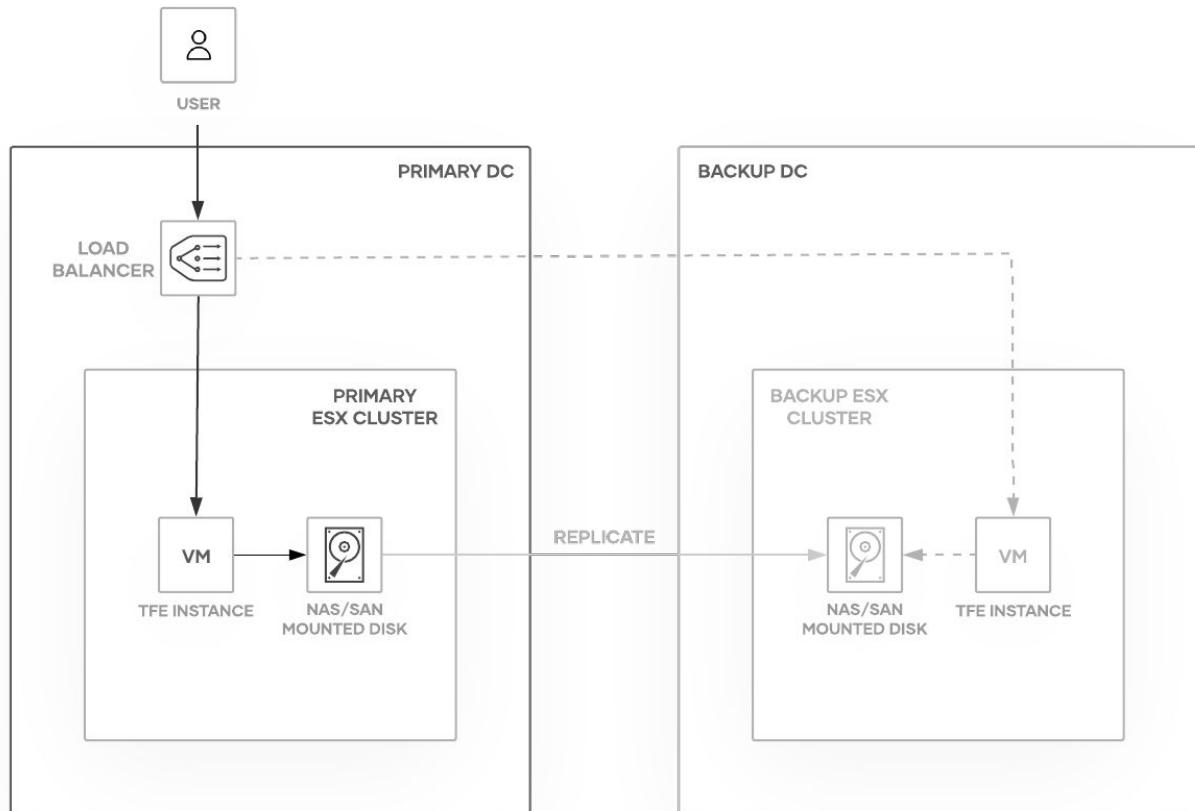
- TFE is a self-managed service composed of microservices running within [Docker](#)
- TFE uses S3-compatible storage, Postgres RDBMS, Redis, and Replicated (license management)
- The preferred installation into a **cloud provider** is a Standalone installation in **External Services** mode
- Common installation into **VMWare** is a Standalone installation in **Mounted Disk** mode





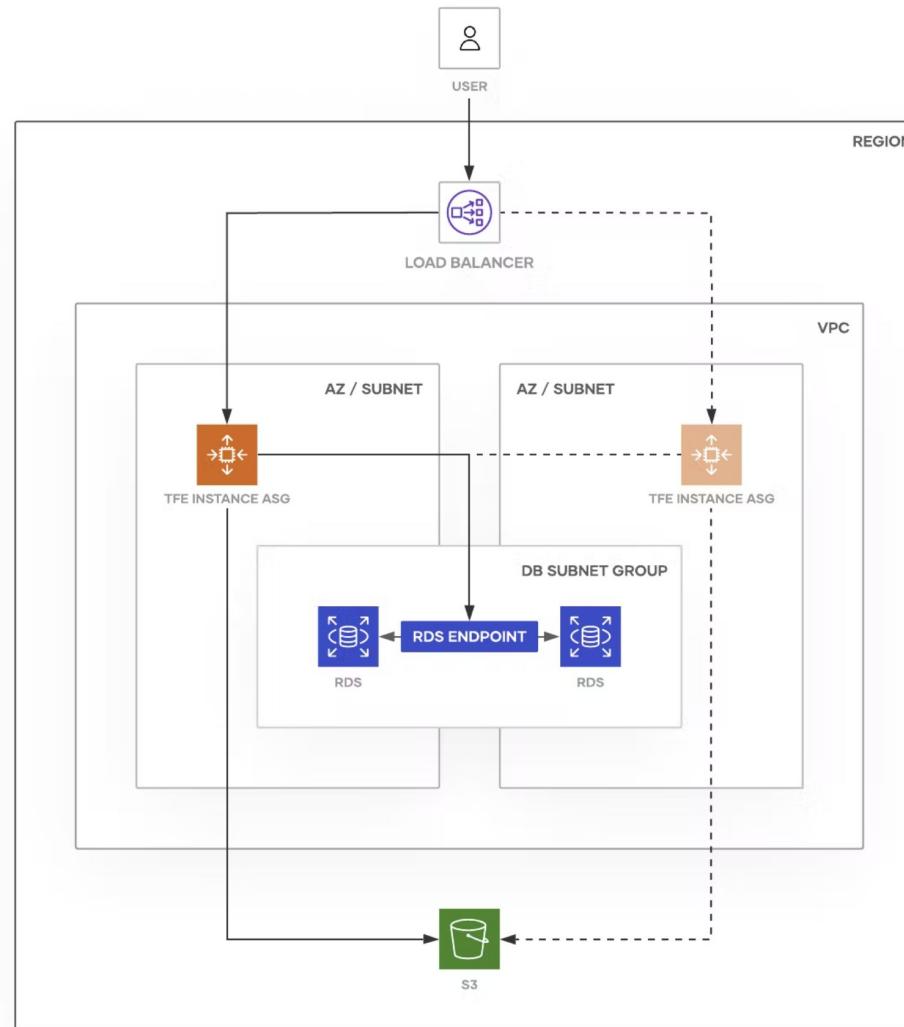
# VMware Standalone

## Reference Architecture



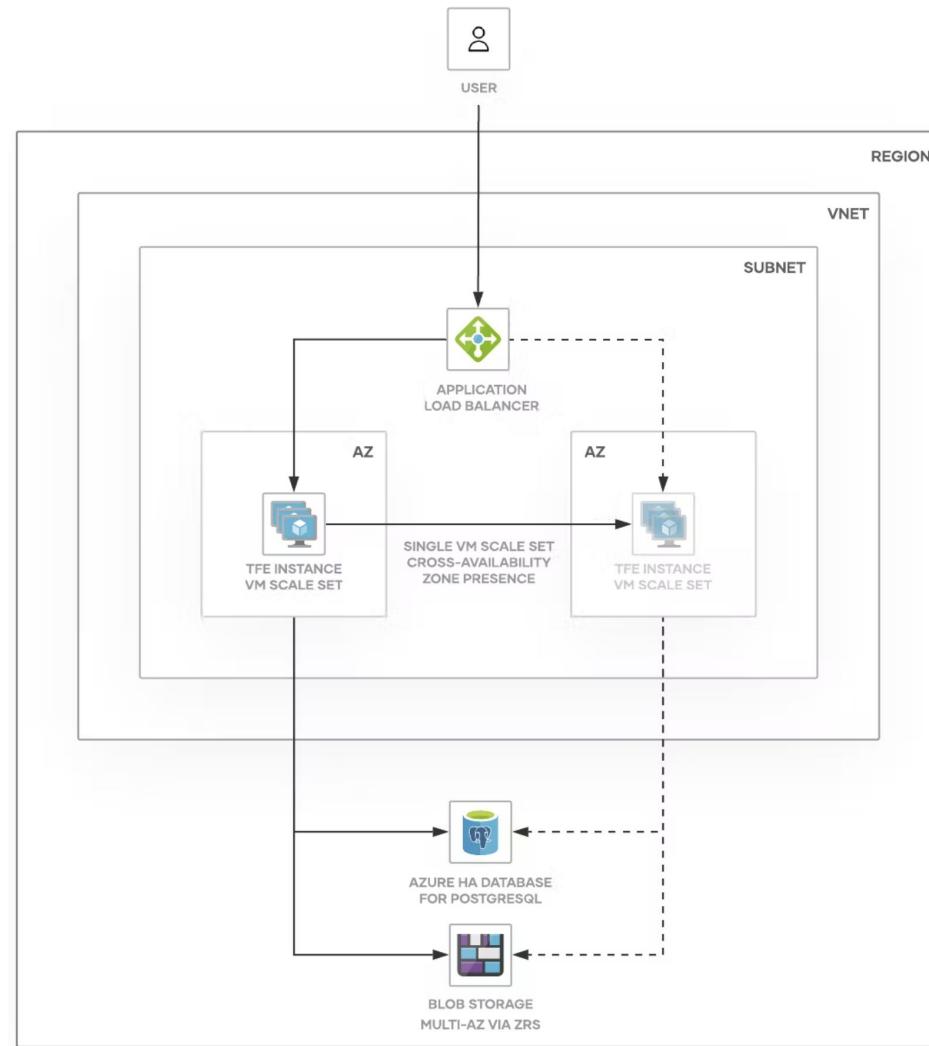


# AWS Standalone Reference Architecture





# Azure Standalone Reference Architecture

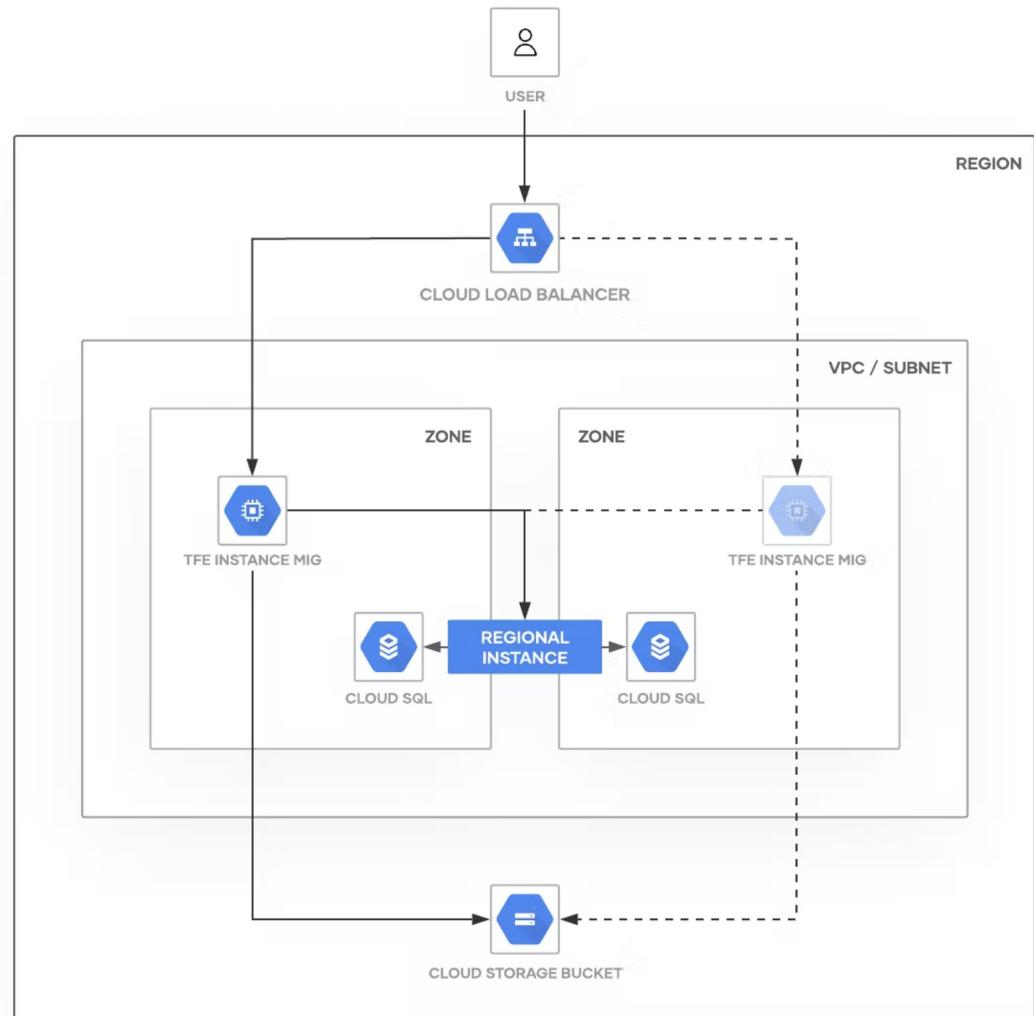




# GCP

## Standalone

### Reference Architecture



— 04

# Dependencies

# OS & Software Requirements



Operating System	
Distro	Version
Ubuntu	20.04, 18.04
Debian	11, 10
<a href="#"><u>CentOS</u></a>	8.4, 7.4 - 7.9
Oracle Linux	8.4, 7.4 - 7.9
<a href="#"><u>Red Hat Enterprise Linux</u></a>	8.x, 7.x
Amazon Linux	2.0

Software	
Name	Version
Docker CE*	20.10.x 19.03.x
containerd.io	containerd 1.4.9 containerd 1.5.5 or newer
runc	runc v1.0.0-rc93 or newer
PostgreSQL	12.x, 13.x, 14.x

Note: Docker CE is not vendor supported on Red Hat Enterprise Linux and Oracle Linux

# VMware Recommended Sizing



## TFE Server (vSphere and vRealize)

Type	CPU Sockets	CPU Cores	Memory	Storage	Storage (Mounted Disk Volume)
Minimum	2	4	16GB RAM	40GB	200GB
Scaled	2	8	32GB RAM	40GB	200GB

## PostgreSQL Database (External Services)

Type	CPU Sockets	CPU Cores	Memory	Storage	
Minimum	2	2 cores	8GB RAM	50GB	
Scaled	2	4-8 cores	16-32GB RAM	50GB	

# Cloud Recommended Sizing



## TFE Server

Type	CPU	Memory	Storage	AWS	Azure	GCP
Minimum	4 core	16GB RAM	50GB	m5.xlarge	Standard_D4_v4	N1-standard-4 (200GB Storage)
Scaled	8 core	32GB RAM	50GB	m5.2xlarge	Standard_D8_v4	N1-standard-8 (200GB Storage)

## PostgreSQL Database

Type	CPU	Memory	Storage	AWS	Azure	GCP
Minimum	4 core	16GB RAM	50GB	db.m4.xlarge	GP_Gen5_4	Custom PostgreSQL Production
Scaled	8 core	32GB RAM	50GB	db.m4.2xlarge	GP_Gen5_8	Custom PostgreSQL Production

# Network Requirements - Egress



## Online Installations

- \*.replicated.com
- \*.quay.io
- quay-registry.s3.amazonaws.com
- \*.cloudfront.net
- \*.docker.com
- \*.docker.io
- \*.terraform.io
- releases.hashicorp.com

## TFE also needs Egress access to

- Any VCS servers/services that will be utilized
- Login/authentication servers if SAML will be configured (ADFS, Okta, etc)
- The various cloud API endpoints that will be managed with Terraform
- Any other third party services that will either be integrated with the TFE server or managed with it



# Network Requirements - Egress

## Cost Estimation APIs

When Cost Estimation is enabled, it uses the respective cloud provider's APIs to get up-to-date pricing info

- [api.pricing.us-east-1.amazonaws.com](https://api.pricing.us-east-1.amazonaws.com)
- [cloudbilling.googleapis.com](https://cloudbilling.googleapis.com)
- [prices.azure.com](https://prices.azure.com)

# Network Requirements - Ingress



## Source - User/Client/VCS

- **80:** Terraform Enterprise application access (HTTP; redirects to HTTPS)
- **443:** Terraform Enterprise application access (HTTPS)
- VCS Web Hooks are inbound, if you intend to use VCS integration with a cloud hosted VCS, you'll need to expose TFE on the public internet

## Source - Administrators

- **22:** SSH access (administration and debugging)
- **8800:** Replicated (TFE setup dashboard, HTTPS)

# Network Requirements - TLS Certificates



## Trusted

- Prior to installation determine the hostname and generate a certificate that will be used during installation
- Requires a FQDN to access the TFE instance, using an IP is not supported
- If using on-prem VCS, the cert could be signed by your own root authority
- If using SaaS VCS, then you'll need a publicly trusted cert

## Self-signed

Don't do it! It will be more trouble in the long run

## Load Balancer / WAF: TLS Offload / Inspection

Ensure you re-encrypt the traffic and still use a trusted cert on the origin servers

— 05

# Installation Patterns



---

# Installation Patterns

- Online Installation



- Airgapped / Offline





TERMINAL

```
$ echo "Installing with internet access"
$ curl -o install-TFE.sh
https://install.terraform.io/TFE/stable
$ sudo bash install-TFE.sh
[... time passes ...]

To continue the installation, visit the following URL
in your browser:

https://<this\_server\_address>:8800
```



---

# Online Installation



# Air Gapped Installation

Running Terraform Enterprise within an Airgapped Environment includes a variety of manual configuration and management processes

TERMINAL

```
$ echo "Installing with Airgap file"
$ apt install docker -y
$ wget
https://install.terraform.io/airgap/latest.tar.gz
$ echo "SCP the .airgap file onto the machine"
$ tar xzf replicated.tar.gz
$ sudo ./install.sh airgap
[... time passes ...]

To continue the installation, visit the following URL
in your browser:

https://<this\_server\_address>:8800
```



# Automated Installation

## Replicated Settings

- Initial replicated configuration can be defined in /etc/replicated.conf
- Settings include:
  - dashboard password
  - TLS certificates
  - license file
  - application settings
- Replicated automatically checks this location during execution of the installer

## Application Settings

- Configuration of TFE can be completed by passing a JSON formatted settings file during the installation
- Online & air gapped installations are supported by this method

## Initial User Creation

- An admin user must be created after installation to use and manage TFE
- This is typically created in the UI, the API can be leveraged for this to automate the process



# Automated Installation

## Replicated Settings

```
{  
    "DaemonAuthenticationType": "password",  
    "DaemonAuthenticationPassword": "your-password-here",  
    "TlsBootstrapType": "server-path",  
    "TlsBootstrapHostname": "server.company.com",  
    "TlsBootstrapCert": "/etc/server.crt",  
    "TlsBootstrapKey": "/etc/server.key",  
    "BypassPreflightChecks": true,  
    "ImportSettingsFrom":  
        "/path/to/application-settings.json",  
    "LicenseFileLocation": "/path/to/license.rli"  
}
```

TERMINAL



A screenshot of a dark-themed code editor window titled "CODE EDITOR". The editor displays a block of Terraform configuration code. The code defines several variables and their values:

```
{  
  "hostname": {  
    "value": "terraform.example.com"  
  },  
  "installation_type": {  
    "value": "poc"  
  },  
  "capacity_concurrency": {  
    "value": "5"  
  }  
}
```



---

# Automated Installation

[Application Settings](#)



# Automated Installation

## Initial admin user creation

```
$ replicated admin --no-tty retrieve-iact > iact.txt
$ curl payload.json
{
    "username": "admin",
    "email": "it@mycompany.com",
    "password": "thisisabadpassword"
}
$ curl --header "Content-Type: application/json" \
    --request POST --data @payload.json \
    https://TFE.company.com/admin/initial-admin-user?token=$(cat
iact.txt)
{
    "status": "created",
    "token":
    "aabbcddd.v1.atlas.ddeeffgghhijjjkkllmmmnnooppqqrrssttuuvvxxxxyz
z"
```

TERMINAL

— 06

# Configuration



The screenshot shows a web browser window with a title bar featuring red, yellow, and green buttons, a back/forward button, and a refresh icon. The main content area has a header 'HTTPS for admin console'. Below it is a paragraph explaining the use of a self-signed TLS certificate. A section titled 'Provide Custom SSL Certificate' contains fields for 'Hostname', 'Private Key', and 'Certificate', each with a 'Choose file' button. A note below states that files will be uploaded directly to the management server. At the bottom are two buttons: 'Use Self-Signed Cert' (orange) and 'Upload & Continue' (green).

HTTPS for admin console

We're currently using a self-signed TLS certificate to secure the communication between your browser & the management console. If you don't upload your own TLS cert, you'll see a warning about this in your browser every time you access the management console.

### Provide Custom SSL Certificate

**Hostname** (Ensure this domain name resolves to this server & is routable on your network)

app.yourdomain.com

**Private Key**

**Certificate**

Files will be uploaded directly to the management server & will never leave.  
[If your private key and cert are already on this server, click here.](#)



# HTTPS Config

Enter hostname and upload private key and certificate



# TFE Licensing

Upload .rli file provided in your welcome email

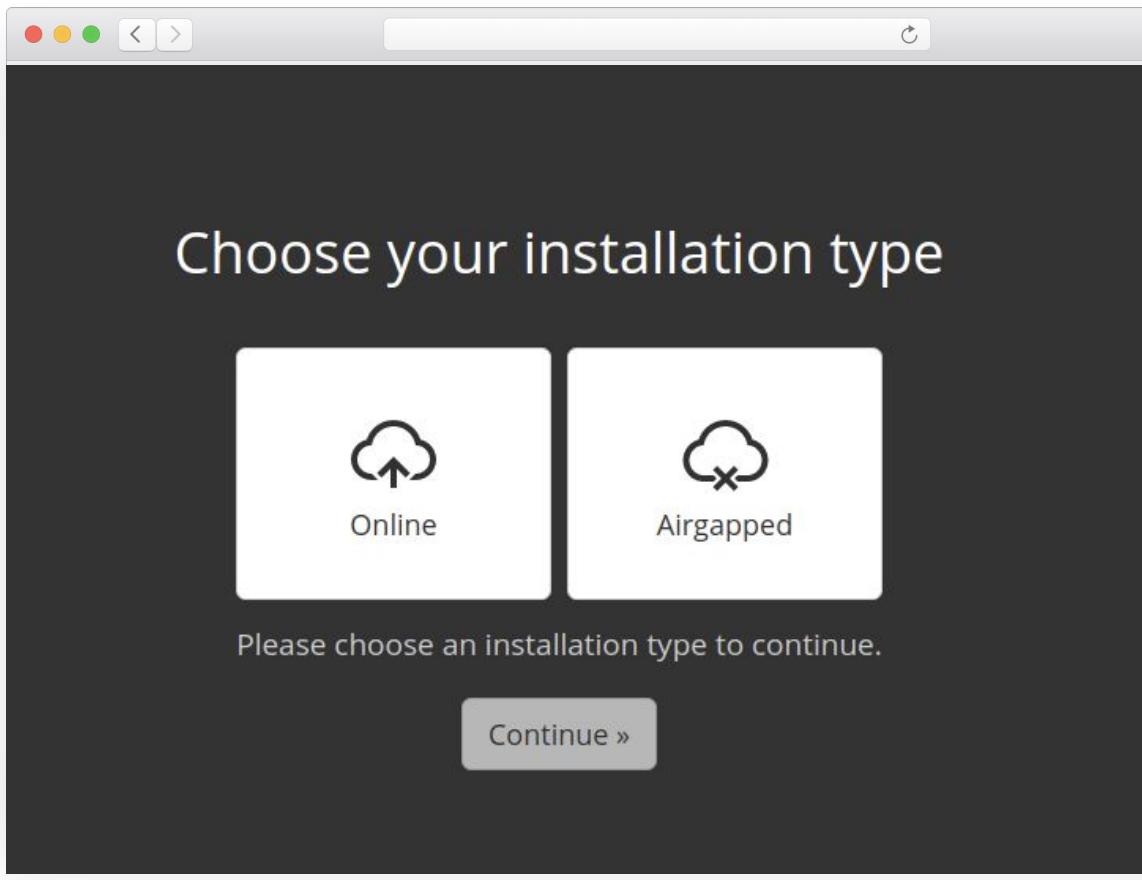
The screenshot shows a dark-themed web page with a header containing window control buttons (red, yellow, green) and back/forward navigation arrows. To the right of the header is a refresh/circular arrow icon.

**Upload your license**

Click the button below to find and upload your license file.  
The file will have a **.rli** extension.

**Choose license**

Restore from a snapshot



---

# Select Installation Type

Select online or air gapped and the preflight check will begin



# Admin Console Auth

- Create password or configure LDAP
- Anonymous is not recommended for use in production deployment

The screenshot shows a window titled "Secure the Admin Console". The window has a standard OS X title bar with red, yellow, and green buttons, and a refresh icon. The main content area contains the following text:

Keeping this admin console secure is important.

You can create a shared password that will be required to access the settings, or you can connect it to your existing directory based authentication system.

Below this text are three radio buttons:

- Anonymous
- Password
- LDAP

There are two input fields labeled "Password" and "Confirm Password". The "Password" field has a small key icon with a dropdown arrow to its right. Below these fields is a large green "Continue" button.

The screenshot shows a Mac OS X-style window titled "Settings". On the left is a sidebar with the following options:

- Hostname
- Encryption Password
- Installation Type
- SSL/TLS Configuration
- Capacity
- Externally Managed Vault
- Terraform Build Worker image
- Backup API Token
- Log Forwarding
- Advanced Configuration

The main content area has three sections:

### Hostname

Ensure this domain name is routable on your network.

Hostname:

[Check DNS](#)

### Encryption Password

The password used to encrypt and decrypt the internally managed Vault unseal key and root token. Required only when using internally managed Vault.

**NOTE:** Please be sure to retain the value as it will be needed in the event of a re-installation.

Encryption Password (Required):

### Installation Type

What kind of installation is this?

**NOTE:** You must not change the Installation Type after initial configuration. Doing so will result in data loss: data is not migrated between Installation Types!



# Installer Settings

Specify installation type and operation mode



# Installer Settings

Specify installation type and operation mode

The screenshot shows a Mac OS X-style window titled "How will you be storing the data generated by the install?". It includes standard window controls (red, yellow, green buttons, close, minimize, maximize) and a refresh icon. Below the title, a note states: "NOTE: You must not change the Production Type after initial configuration. Doing so will result in data loss: data is not migrated between Production Types!" Two radio button options are shown: "External Services" (unchecked) and "Mounted Disk" (checked). A section titled "Mounted Disk Configuration" is expanded, containing the sub-note: "Terraform Enterprise will store all the critical state in a path on the host system. The expectation is that this path is backed by a persistent disk (EBS, SAN, etc.)." A required input field labeled "Path on Host" is present. Another section titled "SSL/TLS Configuration" is also expanded, with the sub-note: "This allows you to add custom SSL/TLS data to the install. The most common usage is to add custom certificates to the system, to allow an internal certificate authority (CA) to be trusted. This is done when you use certificates on your VCS provider and/or Terraform Enterprise installation and thus need Terraform Enterprise to trust services." An input field labeled "Custom Certificate Authority (CA) Bundle" is shown.

How will you be storing the data generated by the install?

**NOTE:** You must not change the Production Type after initial configuration. Doing so will result in data loss: data is not migrated between Production Types!

External Services  Mounted Disk

### Mounted Disk Configuration

Terraform Enterprise will store all the critical state in a path on the host system.  
The expectation is that this path is backed by a persistent disk (EBS, SAN, etc.).

**Path on Host** (Required)

**SSL/TLS Configuration**

This allows you to add custom SSL/TLS data to the install. The most common usage is to add custom certificates to the system, to allow an internal certificate authority (CA) to be trusted.  
This is done when you use certificates on your VCS provider and/or Terraform Enterprise installation and thus need Terraform Enterprise to trust services.

**Custom Certificate Authority (CA) Bundle**

The screenshot shows the Terraform Enterprise configuration interface. At the top, there's a header bar with a refresh icon and a navigation bar. Below it, the main content area has several sections:

- Capacity**: A section for controlling instance performance. It includes a note about setting concurrency too high causing instability, a field for "Total concurrent Terraform plans and applies" set to 10 (with a "Valid number" button), and a field for "The maximum amount of memory" set to 256 (with a "Valid number" button).
- Externally Managed Vault**: A section for configuring an external Vault cluster. It contains a note about the feature, a link to documentation, and a checkbox labeled "Enable Externally Managed Vault".
- Terraform Build Worker image**: A section for configuring the Docker image used for build workers. It notes that it can be the standard image or a custom one. It includes two radio buttons: "Use TFE's standard image" (selected) and "Provide the location of a custom image".



# Installer Settings

Specify installation type and operation mode



# Installer Settings

Click save and the installer will configure, deploy the TFE docker containers and start the app

The screenshot shows a web-based configuration interface for a 'Terraform Build Worker image'. At the top, there are standard OS X window controls (red, yellow, green buttons, close, minimize, maximize, and refresh). Below the title 'Terraform Build Worker image' is a descriptive text: 'Configure which docker image will be used when running terraform plans and applies. This can either be the standard image that ships with PTFE or a custom image that includes extra tools not present in the default one.' Two radio button options are shown: 'Use TFE's standard image' (selected) and 'Provide the location of a custom image'.  
  
A section titled 'Advanced Configuration' follows, with a note: 'These are advanced configuration options that should not be changed without the direction of HashiCorp support personnel.' A checked checkbox labeled 'Enable Metrics Collection' has a tooltip explaining: 'Collected metrics are used by HashiCorp support to diagnose performance issues and help customers tune behavior.'  
  
The 'Initial Admin Creation Token Subnets' section contains a text input field with the value '10.0.1.0/24,172.16.4.0/24'. A tooltip for this field states: 'To automate the creation of the initial admin user, customers can retrieve a special token (called the IACT) that can be exchanged with another API to create the admin user. By default no subnet list is defined and thusly this feature is disabled, but can be configured to allow access from a list of subnets (cidr masks separated by commas). For example: 10.0.1.0/24,172.16.4.0/24.'  
  
The 'Initial Admin Creation Token Time Limit' section contains a text input field with the value '60'. A tooltip for this field states: 'To prevent an unconfigured instance from being discovered and hijacked by a rogue operator, ips from the above subnet list are only allowed to access the retrieval API for a certain initial period of time. This setting defines that time period in minutes. Setting this to *unlimited* will disable the time limit.'  
  
At the bottom right is a blue 'Save' button.

# Next Steps

# Tutorials

<https://developer.hashicorp.com/terraform/tutorials>



**Step-by-step guides to accelerate deployment of Terraform Enterprise**

The screenshot shows a web browser window displaying the Terraform Tutorials page. The URL in the address bar is <https://developer.hashicorp.com/terraform/tutorials>. The page has a dark header with the Terraform logo and navigation links for Home, Documentation, Tutorials, Install, Registry, Try Cloud, Search, and User profile.

The main content area is titled "Get Started - Terraform Cloud". It features a sub-header "Developer / Terraform / Tutorials / Terraform Cloud". Below this is a large button labeled "Start" and a link "Create an account to track your progress." A callout box indicates there are "10 tutorials".

The first tutorial card is titled "What is Terraform Cloud - Intro and Sign Up" with a duration of "5min". Its description states: "Sign up for Terraform Cloud, which provides free remote state storage, a stable run environment, version control system (VCS) driven plans and applies, a collaborative web GUI, and more. Create your first organization." Below this is another card titled "Log in to Terraform Cloud from the CLI" with a duration of "3min".

The left sidebar contains a navigation tree with categories like Tutorials, Overview, Get Started, AWS, Azure, Docker, GCP, OCI, Fundamentals, CLI, Configuration Language, and Modules. The "Terraform Cloud" category is currently selected and highlighted in grey.



---

# Resources

- Reference Architecture
  - [AWS / Azure / GCP](#)
  - [VMware](#)
- [Operational Modes](#)
- [Pre-Install Checklist](#)
- Dependencies
  - [Supported Operating Systems](#)
  - [Docker](#)
  - [Postgres](#)
  - [Network Requirements](#)
  - [Certificates](#)
- [Services & Data Flow Diagram](#)
- [Terraform Enterprise Containers](#)
- [Airgapped Deployment Walkthrough](#)
- [Interactive Installation Walkthrough](#)

# Need Additional Help?



## Customer Success

Contact our Customer Success Management team with any questions. We will help coordinate the right resources for you to get your questions answered.

[customer.success@hashicorp.com](mailto:customer.success@hashicorp.com)

## Technical Support

Something not working quite right? Engage with HashiCorp Technical Support by opening a ticket for your issue at [support.hashicorp.com](https://support.hashicorp.com).

## Discuss

Engage with the HashiCorp Cloud community including HashiCorp Architects and Engineers

[discuss.hashicorp.com](https://discuss.hashicorp.com)

# Upcoming Webinars



## Importing Resources & Migrating State

We breakdown how to migrate your existing TF State from a local state file, remote s3 bucket, into TFE

## Terraform Workflow Management

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

## Office Hours

Bring your questions to Office Hours!

# Action Items



- Share to [customer.success@hashicorp.com](mailto:customer.success@hashicorp.com)
  - Authorized technical contacts for support
  - Stakeholders contact information (name and email addresses)
- Finish requirements gathering and prepare for installation
- Begin deployment of your TFE Instance

# Q & A



# Thank You

[customer.success@hashicorp.com](mailto:customer.success@hashicorp.com)  
[www.hashicorp.com/customer-success](http://www.hashicorp.com/customer-success)