

Index

A

Access Key IDs (in AWS), [Setting Up Your AWS Account](#), [Installing Terraform](#), [Providers-Human users](#), CircleCI as a CI server, with stored secrets-CircleCI as a CI server, with stored secrets

accessing secrets, interfaces for, [The Interface You Use to Access Secrets](#)
(see also secrets management)

accidental complexity, [Why It Takes So Long to Build Production-Grade Infrastructure](#)

accounts (in AWS)

configuring multiple, [Working with Multiple AWS Accounts](#)-[Working with Multiple AWS Accounts](#)

setup, [Setting Up Your AWS Account](#)-[Setting Up Your AWS Account](#)
ad hoc scripts, [Ad Hoc Scripts](#)-[Ad Hoc Scripts](#)

agent versus agentless software, [Agent Versus Agentless](#)-[Agent Versus Agentless](#)

ALB (Application Load Balancer), [Deploying a Load Balancer](#)-[Deploying a Load Balancer](#), [Small Modules](#)-[Small Modules](#)

aliases, [Working with Multiple AWS Regions](#)

configuration aliases, [Creating Modules That Can Work with Multiple Providers](#)-[Creating Modules That Can Work with Multiple Providers](#)

for Gmail, [Working with Multiple AWS Accounts](#)

KMS CMK aliases, [Encrypted files](#)

when to use, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#), [Working with Multiple AWS Accounts](#)

Amazon S3 (Simple Storage Service), as remote backend, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)

Amazon Web Services (see AWS)

AMI (Amazon Machine Image)

ami parameter, [Deploying a Single Server](#)

Packer template, [Server Templating Tools](#)-[Server Templating Tools](#)

zero-downtime deployment, [Zero-Downtime Deployment](#)-[Zero-Downtime Deployment](#)

AMI IDs, managing, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)

ami parameter, [Provisioning Tools](#), [Deploying a Single Server](#)

Ansible, comparison with other IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-[Provisioning plus server templating plus orchestration](#)

application code workflow

automated tests, [Run Automated Tests](#)

changing code, [Make Code Changes](#)-[Make Code Changes](#)

comparison with infrastructure code workflow, [Putting It All Together](#)

deployment, [Deploy](#)-[Promotion across environments](#)

locally run code, [Run the Code Locally](#)

merge and release, [Merge and Release](#)

reviewing code changes, [Submit Changes for Review](#)

steps in, [A Workflow for Deploying Application Code](#)

version control, [Use Version Control](#)

Application Load Balancer (ALB), [Deploying a Load Balancer](#)-[Deploying a Load Balancer](#), [Small Modules](#)-[Small Modules](#)

arguments for resources, [Deploying a Single Server](#)

arrays

lookup syntax, [Loops with the count Parameter](#)

of resources, [Loops with the count Parameter](#)

ASGs (Auto Scaling Groups), [Deploying a Cluster of Web Servers](#)-[Deploying a Cluster of Web Servers](#)

instance refresh, [Zero-Downtime Deployment Has Limitations](#)-[Zero-Downtime Deployment Has Limitations](#)

scheduled actions, [Module Outputs](#)-[Module Outputs](#)

zero-downtime deployment, [Zero-Downtime Deployment](#)-[Zero-Downtime Deployment](#)

assume role policies, [EC2 Instance running Jenkins as a CI server, with IAM roles](#)

Atlantis, [Run Automated Tests](#)-[Deployment tooling](#)

auditing multiple AWS accounts, [Working with Multiple AWS Accounts](#)

authentication (see secrets management)

auto healing, [Orchestration Tools](#)

auto scaling, [Orchestration Tools](#), [Module Outputs](#)-[Module Outputs](#), [If-statements with the count parameter](#)-[If-statements with the count parameter](#)

automated tests, [Automated tests](#)

in application code workflow, [Run Automated Tests](#)
end-to-end tests, [End-to-End Tests](#)-[End-to-End Tests](#)

in infrastructure code workflow, [Run Automated Tests](#)-[Run Automated Tests](#)

integration tests

- with multiple modules, [Integration Tests](#)-[Integration Tests](#)
- retries in, [Retries](#)-[Retries](#)
- Ruby comparison, [Integration Tests](#)-[Integration Tests](#)
- stages of, [Test stages](#)-[Test stages](#)

plan testing, [Plan testing](#)-[Plan testing](#)

purpose of, [Automated Tests](#)

server testing, [Server testing](#)-[Server testing](#)

static analysis, [Static analysis](#)-[Static analysis](#)

types of, [Automated Tests](#)-[Automated Tests](#)

unit tests

- dependency injection, [Dependency injection](#)-[Dependency injection](#)
- Ruby comparison, [Unit Tests](#)-[Unit Tests](#)
- running in parallel, [Running tests in parallel](#)-[Running tests in parallel](#)
- with Terratest, [Unit testing Terraform code](#)-[Unit testing Terraform code](#)

when to use, [When to use validations, preconditions, and postconditions](#)

AZs (Availability Zones), Deploying a Single Server, count and for_each Have Limitations-count and for_each Have Limitations

AWS (Amazon Web Services)

accounts

configuring multiple, Working with Multiple AWS Accounts-Working with Multiple AWS Accounts

setup, Setting Up Your AWS Account-Setting Up Your AWS Account

AZs (Availability Zones), Deploying a Single Server, count and for_each Have Limitations-count and for_each Have Limitations

benefits of, Getting Started with Terraform

configuring as provider, Deploying a Single Server

regions, Deploying a Single Server, Working with Multiple AWS Regions-Working with Multiple AWS Regions

aws eks update-kubeconfig command, Deploying Docker Containers in AWS Using Elastic Kubernetes Service

AWS Provider, Open Source Code Examples, What Is a Provider?

AZs (Availability Zones), Deploying a Single Server, count and for_each Have Limitations-count and for_each Have Limitations

B

backends

configuring with Terragrunt, Promote artifacts across environments-Promote artifacts across environments

local backends, Shared Storage for State Files

remote backends

limitations of, [Limitations with Terraform's Backends-Limitations with Terraform's Backends](#)

shared storage with, [Shared Storage for State Files-Shared Storage for State Files](#)

Bash scripts, [Beyond Terraform Modules](#)

as ad hoc scripts, [Ad Hoc Scripts-Ad Hoc Scripts](#)

externalizing, [The terraform_remote_state Data Source-The terraform_remote_state Data Source](#)

best practices for testing, [Conclusion](#)

blocking public access to S3 buckets, [Shared Storage for State Files](#)

blue-green deployment, [Deployment strategies](#)

branches, reasons to avoid, [The trouble with branches-The trouble with branches](#)

bucket parameter, [Shared Storage for State Files, Shared Storage for State Files](#)

built-in functions, explained, [The terraform_remote_state Data Source-The terraform_remote_state Data Source](#)

bus factor, [What Are the Benefits of Infrastructure as Code?](#)

C

canary deployment, [Deployment strategies](#)

centralized secret stores, [The Way You Store Secrets](#)

changing code

in application code workflow, [Make Code Changes](#)-[Make Code Changes](#)

in infrastructure code workflow, [Make Code Changes](#)

checklist for production-grade infrastructure, [The Production-Grade Infrastructure Checklist](#)-[The Production-Grade Infrastructure Checklist](#)

Chef, comparison with other IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-Provisioning plus server templating plus orchestration

child accounts, creating, [Working with Multiple AWS Accounts](#)-[Working with Multiple AWS Accounts](#)

CI servers

automated tests, [Run Automated Tests](#)

as deployment servers, [Deployment server](#), [Deployment server](#)-[Deployment server](#)

machine user authentication, [Machine users](#)-[GitHub Actions as a CI server](#), [with OIDC](#), [Conclusion](#)

CI/CD workflow

for application code

automated tests, [Run Automated Tests](#)

changing code, [Make Code Changes](#)-[Make Code Changes](#)

comparison with infrastructure code workflow, [Putting It All Together](#)

deployment, [Deploy](#)-Promotion across environments

locally run code, [Run the Code Locally](#)

merge and release, [Merge and Release](#)

reviewing code changes, [Submit Changes for Review](#)
steps in, [A Workflow for Deploying Application Code](#)
version control, [Use Version Control](#)

CircleCI, CircleCI as a CI server, with stored secrets-CircleCI as a CI server, with stored secrets

EC2 Instances with IAM roles, EC2 Instance running Jenkins as a CI server, with IAM roles-EC2 Instance running Jenkins as a CI server, with IAM roles

GitHub Actions, GitHub Actions as a CI server, with OIDC-GitHub Actions as a CI server, with OIDC

for infrastructure code

automated tests, [Run Automated Tests](#)-Run Automated Tests

changing code, [Make Code Changes](#)

comparison with application code workflow, [Putting It All Together](#)

deployment, [Deploy](#)-Promote artifacts across environments

locally run code, [Run the Code Locally](#)

merge and release, [Merge and Release](#)-Merge and Release

reviewing code changes, [Submit Changes for Review](#)-Style guide

steps in, [A Workflow for Deploying Infrastructure Code](#)

version control, [Use Version Control](#)-The trouble with branches

CIDR blocks, [Deploying a Single Web Server](#), [Static analysis](#)

CircleCI, CircleCI as a CI server, with stored secrets-CircleCI as a CI server, with stored secrets, [Run Automated Tests](#)

CircleCI Context, CircleCI as a CI server, with stored secrets
CLB (Classic Load Balancer), Deploying a Load Balancer
cleaning up manual tests, Cleaning Up After Tests-Cleaning Up After Tests
CLI (command-line interface), The Interface You Use to Access Secrets
cloud providers (see providers)

CloudFormation, comparison with other IaC tools, How Does Terraform Compare to Other IaC Tools?-Provisioning plus server templating plus orchestration

cluster of web servers
deploying, Deploying a Cluster of Web Servers-Deploying a Cluster of Web Servers
refactoring, Small Modules-Small Modules

clusters (Kubernetes), Orchestration Tools
deploying Docker containers, Deploying Docker Containers in AWS Using Elastic Kubernetes Service-Deploying Docker Containers in AWS Using Elastic Kubernetes Service

inspecting, A Crash Course on Kubernetes-A Crash Course on Kubernetes

CMK (Customer Managed Key), Encrypted files
code examples
fair use permissions, Using the Code Examples-Using the Code Examples

location of, Open Source Code Examples-Open Source Code Examples
versions used for, Open Source Code Examples

coding guidelines, [Submit Changes for Review](#)-Style guide
collaboration using Git repositories, [Deploying a Single Server](#)
combining IaC tools, [Use of Multiple Tools Together](#)-Provisioning plus
server templating plus orchestration
command history, avoiding writing to, [Human users](#)
command-line interface (CLI), [The Interface You Use to Access Secrets](#)
comments, as documentation, [Documentation](#)
community size in IaC tool comparison, [Large Community Versus Small Community](#)-[Large Community Versus Small Community](#)
comparison of IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-Provisioning plus server templating plus orchestration
agent versus agentless, [Agent Versus Agentless](#)-Agent Versus
Agentless
combining tools, [Use of Multiple Tools Together](#)-Provisioning plus
server templating plus orchestration
configuration management versus provisioning, [Configuration Management Versus Provisioning](#)
general-purpose versus domain-specific language, [General-Purpose Language Versus Domain-Specific Language](#)-General-Purpose
Language Versus Domain-Specific Language
large versus small community, [Large Community Versus Small Community](#)-[Large Community Versus Small Community](#)
master versus masterless, [Master Versus Masterless](#)-Master Versus
Masterless
mature versus cutting edge, [Mature Versus Cutting Edge](#)

mutable versus immutable infrastructure, **Mutable Infrastructure Versus Immutable Infrastructure**-**Mutable Infrastructure Versus Immutable Infrastructure**

paid versus free, **Paid Versus Free Offering**-**Paid Versus Free Offering**
procedural versus declarative language, **Procedural Language Versus Declarative Language**-**Procedural Language Versus Declarative Language**

composable modules for production-grade infrastructure, **Composable Modules**-**Composable Modules**

concat built-in function, **If-else-statements with the count parameter**

conditionals

count parameter

if-else-statements with, **If-else-statements with the count parameter**-**If-else-statements with the count parameter**

if-statements with, **If-statements with the count parameter**-**If-statements with the count parameter**

for_each and **for** expressions, **Conditionals with for_each and for Expressions**-**Conditionals with for_each and for Expressions**

if string directive, **Conditionals with the if String Directive**-**Conditionals with the if String Directive**

ternary syntax, **Isolation via Workspaces**, **If-statements with the count parameter**, **If-else-statements with the count parameter**

types and purpose of, **Conditionals**

configurable web servers, deploying, **Deploying a Configurable Web Server**-**Deploying a Configurable Web Server**

configuration aliases, [Creating Modules That Can Work with Multiple Providers](#)-[Creating Modules That Can Work with Multiple Providers](#)

configuration drift, [What Is DevOps?](#)

configuration files

explained, [How Does Terraform Work?](#)

naming conventions, [Isolation via File Layout](#)-[Isolation via File Layout](#)

configuration management tools, [Configuration Management Tools](#)-[Configuration Management Tools](#)

combining with provisioning, [Provisioning plus configuration management](#)

provisioning tools versus, [Configuration Management Versus Provisioning](#)

container engines, [Server Templating Tools](#)

containers, [Server Templating Tools](#)

deploying with EKS (Elastic Kubernetes Service), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

explained, [A Crash Course on Docker](#)-[A Crash Course on Docker](#)

Pods, [Orchestration Tools](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

control plane (Kubernetes), [A Crash Course on Kubernetes](#), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

cost in IaC tool comparison, [Paid Versus Free Offering](#)-[Paid Versus Free Offering](#)

count parameter

conditionals with

if-else-statements, If-else-statements with the count parameter-If-else-statements with the count parameter

if-statements, If-statements with the count parameter-If-statements with the count parameter

limitations, Loops with the count Parameter-Loops with the count Parameter, count and for_each Have Limitations-count and for_each Have Limitations

loops with, Loops with the count Parameter-Loops with the count Parameter

create_before_destroy lifecycle setting, Deploying a Cluster of Web Servers, Terraform Tips and Tricks: Loops, If-Statements, Deployment, and Gotchas, Zero-Downtime Deployment, Zero-Downtime Deployment Has Limitations, Refactoring Can Be Tricky

creation-time provisioners, Provisioners

credentials (see secrets)

Customer Managed Key (CMK), Encrypted files

customer secrets, The Types of Secrets You Store

D

data sources

external, External data source-External data source

replicating in multiple AWS regions, Working with Multiple AWS Regions-Working with Multiple AWS Regions

secrets management, Resources and Data Sources

with encrypted files, [Encrypted files](#)-[Encrypted files](#)

with environment variables, [Environment variables](#)-[Environment variables](#)

with secret stores, [Secret stores](#)-[Secret stores](#)

for subnets, [Deploying a Cluster of Web Servers](#)-[Deploying a Cluster of Web Servers](#)

`terraform_remote_state`, [The `terraform_remote_state` Data Source](#)-[The `terraform_remote_state` Data Source](#)

declarative language

procedural versus, [Procedural Language Versus Declarative Language](#)-[Procedural Language Versus Declarative Language](#)

Terraform as, [Terraform Tips and Tricks: Loops, If-Statements, Deployment, and Gotchas](#)

declaring variables, [Deploying a Configurable Web Server](#)-[Deploying a Configurable Web Server](#)

default parameter, [Deploying a Configurable Web Server](#)

Default VPC (in AWS), [Setting Up Your AWS Account](#), [Deploying a Single Web Server](#), [Deploying a Cluster of Web Servers](#)

`default_tags` block, [Loops with `for_each` Expressions](#)

defining listeners, [Deploying a Load Balancer](#)

dependencies

`implicit`, [Deploying a Single Web Server](#)

in Terragrunt, [Promote artifacts across environments](#)

types of, [Versioned Modules](#)

versioning pinning, [Versioned Modules](#)-[Versioned Modules](#)
dependency injection, [Dependency injection](#)-[Dependency injection](#)
depends_on parameter, [Deploying a Configurable Web Server](#)
deploying
application code, [Deploy](#)-[Promotion across environments](#)
Docker with EKS (Elastic Kubernetes Service), [Deploying Docker](#)
[Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying](#)
[Docker Containers in AWS Using Elastic Kubernetes Service](#)
infrastructure code, [Deploy](#)-[Promote artifacts across environments](#)
load balancers, [Deploying a Load Balancer](#)-[Deploying a Load](#)
[Balancer](#)
modules in multiple AWS regions, [Working with Multiple AWS](#)
[Regions](#)-[Working with Multiple AWS Regions](#)
resources in multiple AWS regions, [Working with Multiple AWS](#)
[Regions](#)-[Working with Multiple AWS Regions](#)
servers
cluster of web servers, [Deploying a Cluster of Web Servers](#)-
[Deploying a Cluster of Web Servers](#)
configurable web servers, [Deploying a Configurable Web Server](#)-
[Deploying a Configurable Web Server](#)
single servers, [Deploying a Single Server](#)-[Deploying a Single](#)
[Server](#)
web servers, [Deploying a Single Web Server](#)-[Deploying a Single](#)
[Web Server](#)

with zero-downtime deployment, [Zero-Downtime Deployment-Zero-Downtime Deployment](#)

deployment servers, [Deployment server](#), [Deployment server-Deployment server](#)

deployment strategies, [Deployment strategies](#)-[Deployment strategies](#), [Deployment strategies](#)-[Deployment strategies](#)

deployment tools, [Deployment tooling](#), [Deployment tooling](#)

Deployments (Kubernetes), [Orchestration Tools](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

description parameter, [Deploying a Configurable Web Server](#), [Deploying a Configurable Web Server](#)

destroy-time provisioners, [Provisioners](#)

DevOps, [What Is DevOps?-What Is DevOps?](#)

project time estimates, [Why It Takes So Long to Build Production-Grade Infrastructure](#)-[Why It Takes So Long to Build Production-Grade Infrastructure](#)

resources for information, [Recommended Reading](#)-[Online Forums](#)

Docker

containers (see containers)

deploying with EKS (Elastic Kubernetes Service), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

explained, [A Crash Course on Docker](#)-[A Crash Course on Docker](#)

installing, [A Crash Course on Docker](#)

purpose of, [Server Templating Tools](#)

Docker Hub, [A Crash Course on Docker](#)

docker kill command, [A Crash Course on Kubernetes](#)

docker ps command, [A Crash Course on Kubernetes](#)

Docker Registry, [A Crash Course on Docker](#)

docker rm command, [A Crash Course on Docker](#)

docker run command, [A Crash Course on Docker](#)

docker start command, [A Crash Course on Docker](#)

documentation

IaC as, [What Are the Benefits of Infrastructure as Code?](#)

for Terraform, [What You Won't Find in This Book, Deploying a Single Server](#)

types of, [Documentation-Documentation](#)

DRY (Don't Repeat Yourself) principle, [Deploying a Configurable Web Server](#), [Limitations with Terraform's Backends](#), Promote artifacts across environments

DSL (domain-specific language) versus GPL (general-purpose programming language), [General-Purpose Language Versus Domain-Specific Language](#)-[General-Purpose Language Versus Domain-Specific Language](#)

DynamoDB tables

limitations of remote backends, [Limitations with Terraform's Backends](#)-[Limitations with Terraform's Backends](#)

locking with, [Shared Storage for State Files](#)

dynamodb_table parameter, [Shared Storage for State Files](#)

E

EC2 Instances, Deploying a Single Server, Deploying a Single Server
(see also servers)

deploying in multiple AWS regions, Working with Multiple AWS Regions-Working with Multiple AWS Regions

IAM roles for, EC2 Instance running Jenkins as a CI server, with IAM roles-EC2 Instance running Jenkins as a CI server, with IAM roles

EKS (Elastic Kubernetes Service), Deploying Docker Containers in AWS Using Elastic Kubernetes Service-Deploying Docker Containers in AWS Using Elastic Kubernetes Service

ELB (Elastic Load Balancer) service, Deploying a Load Balancer

enabling

server-side encryption, Shared Storage for State Files

versioning in Amazon S3, Shared Storage for State Files

encrypt parameter, Shared Storage for State Files

encryption

secrets management with, Encrypted files-Encrypted files

server-side, enabling, Shared Storage for State Files

end-to-end tests, Automated Tests, End-to-End Tests-End-to-End Tests

enforcing tagging standards, Loops with for_each Expressions

environment variables, secrets management with, The terraform_remote_state Data Source, Human users-Human users, Environment variables-Environment variables

environments

folders for, **Isolation via File Layout**-**Isolation via File Layout**
promotion across, **Promotion across environments**, **Promote artifacts across environments**-**Promote artifacts across environments**
sandbox, **Manual Testing Basics**-**Cleaning Up After Tests**, **Unit testing Terraform code**, **Run the Code Locally**
errors in deployment, **Deployment strategies**-**Deployment strategies**
essential complexity, **Why It Takes So Long to Build Production-Grade Infrastructure**
example code, as documentation, **Documentation examples**
fair use permissions, **Using the Code Examples**-**Using the Code Examples**
location of, **Open Source Code Examples**-**Open Source Code Examples**
versions used for, **Open Source Code Examples**
executing scripts
for external data source, **External data source**-**External data source**
with provisioners, **Provisioners**-**Provisioners**
expressions, **Deploying a Single Web Server**
external data source, **External data source**-**External data source**
externalizing Bash scripts, **The terraform_remote_state Data Source**-**The terraform_remote_state Data Source**

F

fair use permissions for code examples, [Using the Code Examples](#)-[Using the Code Examples](#)

false incrementalism, [Work Incrementally](#)

feature toggles, [Deployment strategies](#)

file layout

conventions for, [File layout](#)

isolating state files, [Isolation via File Layout](#)-[Isolation via File Layout](#)

file paths

local, [Module Versioning](#)

path references for, [File Paths](#)-[File Paths](#)

file-based secret stores, [The Way You Store Secrets](#)

folders

for environments, [Isolation via File Layout](#)-[Isolation via File Layout](#)

running commands across multiple, [Isolation via File Layout](#)

for expressions

conditionals with, [Conditionals with for_each and for Expressions](#)-[Conditionals with for_each and for Expressions](#)

loops with, [Loops with for Expressions](#)-[Loops with for Expressions](#)

for string directive, loops with, [Loops with the for String Directive](#)-[Loops with the for String Directive](#)

format built-in function, [The terraform_remote_state Data Source](#)

for_each expressions

conditionals with, [Conditionals with for_each and for Expressions](#)-[Conditionals with for_each and for Expressions](#)

limitations, count and for_each Have Limitations-count and for_each Have Limitations

loops with, Loops with for_each Expressions-Loops with for_each Expressions

function composition, Composable Modules-Composable Modules

G

get deployments command, A Crash Course on Kubernetes, Deploying Docker Containers in AWS Using Elastic Kubernetes Service

get nodes command, A Crash Course on Kubernetes, Deploying Docker Containers in AWS Using Elastic Kubernetes Service

get pods command, A Crash Course on Kubernetes, Deploying Docker Containers in AWS Using Elastic Kubernetes Service

get services command, A Crash Course on Kubernetes, Deploying Docker Containers in AWS Using Elastic Kubernetes Service

git commit command, Deploying a Single Server

git push command, Deploying a Single Server

Git repositories

collaborating with teammates, Deploying a Single Server

SSH authentication for, Module Versioning

GitHub Actions, GitHub Actions as a CI server, with OIDC-GitHub Actions as a CI server, with OIDC

GitHub, pull requests, Submit Changes for Review

.gitignore file, Deploying a Single Server

Gmail aliases, Working with Multiple AWS Accounts

Go, installing, [Unit testing Terraform code](#)

Golden Rule of Terraform, [The Golden Rule of Terraform-The Golden Rule of Terraform](#)

GPL (general-purpose programming language) versus DSL (domain-specific language), [General-Purpose Language Versus Domain-Specific Language-General-Purpose Language Versus Domain-Specific Language](#)

H

HCL (HashiCorp Configuration Language), [Deploying a Single Server health checks](#), [Deploying a Load Balancer](#)

heredoc syntax, [Deploying a Single Web Server](#), Conditionals with the if String Directive

history files, avoiding writing to, [Human users](#)

Hofstadter's Law, [Why It Takes So Long to Build Production-Grade Infrastructure](#)

human users, secrets management and, [Human users-Human users hypervisors](#), [Server Templating Tools](#)

I

IaC (infrastructure as code), [Production-Grade Terraform Code](#)

(see also production-grade infrastructure)

ad hoc scripts, [Ad Hoc Scripts-Ad Hoc Scripts](#)

benefits of, [What Are the Benefits of Infrastructure as Code?-What Are the Benefits of Infrastructure as Code?](#)

configuration management tools, [Configuration Management Tools-Configuration Management Tools](#)

importing into state files, [Valid Plans Can Fail](#)
orchestration tools, [Orchestration Tools](#)-[Orchestration Tools](#)
out of band resources, [Valid Plans Can Fail](#)
provisioning tools, [Provisioning Tools](#)
purpose of, [What Is Infrastructure as Code?](#)
refactoring, [Refactoring Can Be Tricky](#)-[Refactoring Can Be Tricky](#)
resources for information, [Recommended Reading](#)-[Online Forums](#)
server templating tools, [Server Templating Tools](#)-[Server Templating Tools](#)
team adoption of
incrementalism in, [Work Incrementally](#)-[Work Incrementally](#)
learning curve, [Give Your Team the Time to Learn](#)-[Give Your Team the Time to Learn](#)
return on investment, [Convince Your Boss](#)-[Convince Your Boss](#)
tool comparison, [How Does Terraform Compare to Other IaC Tools?](#)-
[Provisioning plus server templating plus orchestration](#)
agent versus agentless, [Agent Versus Agentless](#)-[Agent Versus Agentless](#)
combining tools, [Use of Multiple Tools Together](#)-[Provisioning plus server templating plus orchestration](#)
configuration management versus provisioning, [Configuration Management Versus Provisioning](#)
general-purpose versus domain-specific language, [General-Purpose Language Versus Domain-Specific Language](#)-[General-Purpose Language Versus Domain-Specific Language](#)

large versus small community, [Large Community Versus Small Community](#)-[Large Community Versus Small Community](#)

master versus masterless, [Master Versus Masterless](#)-[Master Versus Masterless](#)

mature versus cutting edge, [Mature Versus Cutting Edge](#)

mutable versus immutable infrastructure, [Mutable Infrastructure Versus Immutable Infrastructure](#)-[Mutable Infrastructure Versus Immutable Infrastructure](#)

paid versus free, [Paid Versus Free Offering](#)-[Paid Versus Free Offering](#)

procedural versus declarative language, [Procedural Language Versus Declarative Language](#)-[Procedural Language Versus Declarative Language](#)

validating, [Manual Testing Basics](#)

IAM (Identity and Access Management) service, [Setting Up Your AWS Account](#)

IAM OIDC identity providers, [GitHub Actions as a CI server, with OIDC IAM Policies](#), [Setting Up Your AWS Account](#)

IAM roles

child account authentication, [Working with Multiple AWS Accounts](#)

cross-account authentication, [Working with Multiple AWS Accounts](#)

for EC2 Instances, [EC2 Instance running Jenkins as a CI server, with IAM roles](#)-[EC2 Instance running Jenkins as a CI server, with IAM roles](#)

IAM users

changing policies, If-else-statements with the count parameter-If-else-statements with the count parameter

creating, Setting Up Your AWS Account, Loops with the count Parameter-Loops with for_each Expressions

errors, Valid Plans Can Fail-Valid Plans Can Fail

idempotence, Configuration Management Tools, Deploying a Single Server, Shared Storage for State Files

if string directive, conditionals with, Conditionals with the if String Directive-Conditionals with the if String Directive

if-else-statements with count parameter, If-else-statements with the count parameter-If-else-statements with the count parameter

if-statements with count parameter, If-statements with the count parameter-If-statements with the count parameter

images (of servers), Server Templating Tools-Server Templating Tools

immutable infrastructure

explained, Server Templating Tools

mutable versus, Mutable Infrastructure Versus Immutable Infrastructure-Mutable Infrastructure Versus Immutable Infrastructure

promotion across environments, Promotion across environments, Promote artifacts across environments

releasing application code, Merge and Release

workflows, Putting It All Together-Putting It All Together

implicit dependencies, Deploying a Single Web Server

importing infrastructure, Valid Plans Can Fail

incrementalism in IaC (infrastructure as code) adoption, [Work Incrementally](#)-[Work Incrementally](#)

infrastructure as code (see IaC)

infrastructure code workflow

automated tests, [Run Automated Tests](#)-[Run Automated Tests](#)

changing code, [Make Code Changes](#)

comparison with application code workflow, [Putting It All Together](#)

deployment, [Deploy](#)-Promote artifacts across environments

locally run code, [Run the Code Locally](#)

merge and release, [Merge and Release](#)-[Merge and Release](#)

reviewing code changes, [Submit Changes for Review](#)-Style guide

steps in, [A Workflow for Deploying Infrastructure Code](#)

version control, [Use Version Control](#)-The trouble with branches

infrastructure secrets, [The Types of Secrets You Store](#)

inline blocks

creating multiple with for_each expressions, [Loops with for_each Expressions](#)-[Loops with for_each Expressions](#)

separate resources versus, [Inline Blocks](#)-[Inline Blocks](#)

input variables, [Deploying a Configurable Web Server](#)-[Deploying a Configurable Web Server](#)

in composable modules, [Composable Modules](#)-[Composable Modules](#)

in modules, [Module Inputs](#)-[Module Inputs](#)

inspecting Kubernetes clusters, [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

installing

Docker, [A Crash Course on Docker](#)

Go, [Unit testing Terraform code](#)

kubectl, [A Crash Course on Kubernetes](#)

providers, [How Do You Install Providers?](#)-[How Do You Install Providers?](#)

Terraform, [Installing Terraform](#)-[Installing Terraform](#), [Versioned Modules](#)

Terragrunt, [Promote artifacts across environments](#)

instance metadata endpoints, [EC2 Instance running Jenkins as a CI server, with IAM roles](#)

instance profiles, [EC2 Instance running Jenkins as a CI server, with IAM roles](#)

instance refresh, [Zero-Downtime Deployment Has Limitations](#)-[Zero-Downtime Deployment Has Limitations](#)

instance_type parameter, [Deploying a Single Server](#)

integration tests

purpose of, [Automated Tests](#)

retries in, [Retries](#)-[Retries](#)

Ruby comparison, [Integration Tests](#)-[Integration Tests](#)

stages of, [Test stages](#)-[Test stages](#)

with multiple modules, [Integration Tests](#)-[Integration Tests](#)

interfaces, accessing secrets, [The Interface You Use to Access Secrets](#)
interpolation, [Deploying a Configurable Web Server](#)
isolating

multiple AWS accounts, [Working with Multiple AWS Accounts](#)

at network level, [Inline Blocks](#)

state files, [What Is Terraform State?](#)

purpose of, [State File Isolation-State File Isolation](#)

via file layout, [Isolation via File Layout-Isolation via File Layout](#)

via workspaces, [Isolation via Workspaces-Isolation via Workspaces](#)

K

kernel, [Server Templating Tools](#)

kernel space, [Server Templating Tools](#)

key parameter, [Shared Storage for State Files](#)

key policies, [Encrypted files](#)

KMS (key management system), [The Way You Store Secrets](#)

kubectl, [A Crash Course on Kubernetes](#), [A Crash Course on Kubernetes-A Crash Course on Kubernetes](#)

Kubernetes

clusters, [Orchestration Tools](#)

deploying Docker containers, [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

inspecting, [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

control plane, [A Crash Course on Kubernetes](#), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

[EKS \(Elastic Kubernetes Service\)](#), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

explained, [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

[kubectl](#), [A Crash Course on Kubernetes](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

objects in, [A Crash Course on Kubernetes](#)

[Pods](#), [Orchestration Tools](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

running, [A Crash Course on Kubernetes](#)

worker nodes, [A Crash Course on Kubernetes](#), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

[Kubernetes Deployment](#), [Orchestration Tools](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

[Kubernetes Service](#), [A Crash Course on Kubernetes](#), [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

L

large modules, limitations of, [Small Modules](#)

launch configurations, [Deploying a Cluster of Web Servers](#)

launch templates, [Deploying a Cluster of Web Servers](#)

learning curve in IaC (infrastructure as code) adoption, [Give Your Team the Time to Learn](#)-[Give Your Team the Time to Learn](#)

length built-in function, [Loops with the count Parameter](#)

lifecycle settings, [Deploying a Cluster of Web Servers](#), [Shared Storage for State Files](#), [Zero-Downtime Deployment](#)

list comprehensions (Python), [Loops with for Expressions](#)

listener rules, creating, [Deploying a Load Balancer](#)

listeners, defining, [Deploying a Load Balancer](#)

literals, [Deploying a Single Web Server](#)

live repository, [Live repo and modules repo](#)-[The Golden Rule of Terraform](#), [Promote artifacts across environments](#)-[Promote artifacts across environments](#)

load balancing, [Orchestration Tools](#), [Deploying a Load Balancer](#)-[Deploying a Load Balancer](#)

local backends, [Shared Storage for State Files](#)

local file paths, [Module Versioning](#)

local names, [How Do You Install Providers?](#)

local references, [Module Locals](#)

local values in modules, [Module Locals](#)-[Module Locals](#)

locally run code

in application code workflow, [Run the Code Locally](#)

in infrastructure code workflow, [Run the Code Locally](#)

lock files, [Versioned Modules](#)

locking

with DynamoDB tables, Shared Storage for State Files
with remote backends, Shared Storage for State Files
state files, What Is Terraform State?
version control disadvantages of, Shared Storage for State Files
lookup syntax for arrays, Loops with the count Parameter
loops
count parameter, Loops with the count Parameter-Loops with the
count Parameter
for expressions, Loops with for Expressions-Loops with for
Expressions
for string directive, Loops with the for String Directive-Loops with the
for String Directive
for_each expressions, Loops with for_each Expressions-Loops with
for_each Expressions
types and purpose of, Loops

M

machine users, secrets management and, Machine users-GitHub Actions as
a CI server, with OIDC, Conclusion
managed node groups, Deploying Docker Containers in AWS Using Elastic
Kubernetes Service-Deploying Docker Containers in AWS Using Elastic
Kubernetes Service
Managed Policies (in AWS), Setting Up Your AWS Account
manual tests
cleanup, Cleaning Up After Tests-Cleaning Up After Tests

explained, [Manual Testing Basics](#)-[Manual Testing Basics](#)

Ruby comparison, [Manual Tests](#)-[Manual Tests](#)

maps of resources, [Loops with for_each Expressions](#)

master versus masterless servers, [Master Versus Masterless](#)-[Master Versus Masterless](#)

maturity in IaC tool comparison, [Mature Versus Cutting Edge](#)

merging code

in application code workflow, [Merge and Release](#)

in infrastructure code workflow, [Merge and Release](#)-[Merge and Release](#)

mocks, [Automated Tests](#)

modules

authenticating multiple AWS accounts, [Working with Multiple AWS Accounts](#)-[Working with Multiple AWS Accounts](#)

configuring for Kubernetes, [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

count parameter, [Loops with the count Parameter](#)-[Loops with the count Parameter](#)

creating for Kubernetes, [A Crash Course on Kubernetes](#)-[A Crash Course on Kubernetes](#)

creating reusable, [Module Basics](#)-[Module Basics](#)

deploying in multiple AWS regions, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)

for_each expressions, [Loops with for_each Expressions](#)

inline blocks versus separate resources, [Inline Blocks-Inline Blocks](#)
input variables, [Module Inputs-Module Inputs](#)
integration tests, [Integration Tests-Integration Tests](#)
local values, [Module Locals-Module Locals](#)
for multiple providers, [Creating Modules That Can Work with Multiple Providers-Creating Modules That Can Work with Multiple Providers](#)
output variables, [Module Outputs-Module Outputs](#)
path references, [File Paths-File Paths](#)
for production-grade infrastructure
composable modules, [Composable Modules-Composable Modules](#)
non-Terraform code in, [Beyond Terraform Modules-External data source](#)
small modules, [Small Modules-Small Modules](#)
testable modules, [Testable Modules-When to use validations, preconditions, and postconditions](#)
versioned modules, [Versioned Modules-Versioned Modules](#)
publishing, [Versioned Modules-Versioned Modules](#)
purpose of, [Limitations with Terraform's Backends, How to Create Reusable Infrastructure with Terraform Modules](#)
repository for, [Live repo and modules repo-Live repo and modules repo](#)
self-validating, [Testable Modules](#)

precondition/postcondition blocks in, [Preconditions and postconditions](#)-When to use validations, preconditions, and postconditions

validation blocks in, [Validations](#)-[Validations](#)

tagging standards, [Loops with for_each Expressions](#)

types of, [Module Basics](#), [Creating Modules That Can Work with Multiple Providers](#)

unit tests for, [Unit testing Terraform code](#)-[Unit testing Terraform code](#)

versioning, [Module Versioning](#)-[Module Versioning](#)

versioning pinning, [Versioned Modules](#)-[Versioned Modules](#)

multiple AWS accounts, configuring, [Working with Multiple AWS Accounts](#)-[Working with Multiple AWS Accounts](#)

multiple AWS regions, configuring, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)

multiple folders, running commands across, [Isolation via File Layout](#)

multiple inline blocks, creating with for_each expressions, [Loops with for_each Expressions](#)-[Loops with for_each Expressions](#)

multiple modules, integration tests with, [Integration Tests](#)-[Integration Tests](#)

multiple providers

Docker, explained, [A Crash Course on Docker](#)-[A Crash Course on Docker](#)

EKS (Elastic Kubernetes Service), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)-[Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

Kubernetes, explained, **A Crash Course on Kubernetes**-**A Crash Course on Kubernetes**

multiple AWS accounts, **Working with Multiple AWS Accounts**-**Working with Multiple AWS Accounts**

multiple AWS regions, **Working with Multiple AWS Regions**-**Working with Multiple AWS Regions**

multiple copies of different providers, **Working with Multiple Different Providers**-**Deploying Docker Containers in AWS Using Elastic Kubernetes Service**

multiple copies of same provider, **Working with Multiple Copies of the Same Provider**-**Creating Modules That Can Work with Multiple Providers**

reusable modules for, **Creating Modules That Can Work with Multiple Providers**-**Creating Modules That Can Work with Multiple Providers**

mutable versus immutable infrastructure, **Mutable Infrastructure Versus Immutable Infrastructure**-**Mutable Infrastructure Versus Immutable Infrastructure**

N

naming

configuration files, **Isolation via File Layout**-**Isolation via File Layout**
resources, **Deploying a Single Server**

network isolation, **Inline Blocks**

network security

CIDR blocks, **Deploying a Single Web Server**, **Static analysis**

data sources for subnets, **Deploying a Cluster of Web Servers**

Default VPC (in AWS), Setting Up Your AWS Account
network isolation, **Inline Blocks**
port numbers and, **Deploying a Single Web Server**
security groups, creating, **Deploying a Single Web Server**, **Deploying a Load Balancer**
VPC subnets, **Deploying a Single Web Server**
NLB (Network Load Balancer), **Deploying a Load Balancer**
non-Terraform code, running in modules, **Beyond Terraform Modules-External data source**
null_resource, **Provisioners with null_resource-Provisioners with null_resource**

O

objects (Kubernetes), **A Crash Course on Kubernetes**
OIDC (Open ID Connect), **GitHub Actions as a CI server, with OIDC-GitHub Actions as a CI server, with OIDC**
one built-in function, **If-else-statements with the count parameter**
OPA (Open Policy Agent), **Plan testing-Plan testing**
OpenStack Heat, comparison with other IaC tools, **How Does Terraform Compare to Other IaC Tools?-Provisioning plus server templating plus orchestration**
orchestration tools, **Orchestration Tools-Orchestration Tools**
combining with provisioning and server templating, **Provisioning plus server templating plus orchestration-Provisioning plus server templating plus orchestration**

for deployment, [Deployment tooling](#)
purpose of, [A Crash Course on Kubernetes](#)
out of band resources, [Valid Plans Can Fail](#)
output variables, [Deploying a Configurable Web Server](#)-[Deploying a Configurable Web Server](#)
in composable modules, [Composable Modules](#)-[Composable Modules](#)
in modules, [Module Outputs](#)-[Module Outputs](#)

P

Packer, [Server Templating Tools](#)-[Server Templating Tools](#)
parallel unit tests, running, [Running tests in parallel](#)-[Running tests in parallel](#)
partial configurations, [Limitations with Terraform's Backends](#), [Integration Tests](#)-[Integration Tests](#)
passing secrets via environment variables, [The terraform_remote_state Data Source](#), [Human users](#)-[Human users](#)
path references in modules, [File Paths](#)-[File Paths](#)
permissions for code examples, [Using the Code Examples](#)-[Using the Code Examples](#)
personal secrets, [The Types of Secrets You Store](#)
plain text, avoiding secrets in, [Secret Management Basics](#)-[Secret Management Basics](#)
plan files, secrets management, [Plan files](#)-[Plan files](#)
plan testing, [Plan testing](#)-[Plan testing](#), [Run Automated Tests](#)
plugins, providers as, [What Is a Provider?](#)-[What Is a Provider?](#)

Pod Template, A Crash Course on Kubernetes-A Crash Course on Kubernetes

Pods, Orchestration Tools, A Crash Course on Kubernetes-A Crash Course on Kubernetes

policy-as-code tools, Plan testing-Plan testing

port numbers, Deploying a Single Web Server

postcondition blocks, Preconditions and postconditions-When to use validations, preconditions, and postconditions

precondition blocks, Preconditions and postconditions-When to use validations, preconditions, and postconditions

preferred local names, How Do You Install Providers?

prevent_destroy parameter, Shared Storage for State Files

private API, state files as, What Is Terraform State?

private Git repositories, Module Versioning

private keys, The Way You Store Secrets

private subnets, Deploying a Single Web Server

Private Terraform Registry, Versioned Modules

procedural versus declarative language, Procedural Language Versus Declarative Language-Procedural Language Versus Declarative Language

production-grade infrastructure

checklist for, The Production-Grade Infrastructure Checklist-The Production-Grade Infrastructure Checklist

definition of, Production-Grade Terraform Code

modules for

composable modules, [Composable Modules](#)-[Composable Modules](#)

non-Terraform code in, [Beyond Terraform Modules](#)-[External data source](#)

small modules, [Small Modules](#)-[Small Modules](#)

testable modules, [Testable Modules](#)-[When to use validations, preconditions, and postconditions](#)

versioned modules, [Versioned Modules](#)-[Versioned Modules](#)

time estimates for, [Production-Grade Terraform Code](#)-[Why It Takes So Long to Build Production-Grade Infrastructure](#)

promotion across environments, [Promotion across environments](#), [Promote artifacts across environments](#)-[Promote artifacts across environments](#)

providers, [A Crash Course on Docker](#)

(see also multiple providers)

configuring, [Deploying a Single Server](#), [How Do You Use Providers?](#)-[How Do You Use Providers?](#)

multiple AWS accounts, [Working with Multiple AWS Accounts](#)-[Working with Multiple AWS Accounts](#)

multiple AWS regions, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)

in reusable modules, [Creating Modules That Can Work with Multiple Providers](#)-[Creating Modules That Can Work with Multiple Providers](#)

installing, [How Do You Install Providers?](#)-[How Do You Install Providers?](#)

as plugins, [What Is a Provider?](#)-[What Is a Provider?](#)

secrets management, **Providers**-**Providers**
human users and, **Human users**-**Human users**
machine users and, **Machine users**-GitHub Actions as a CI server,
with OIDC, Conclusion
transparent portability, **How Does Terraform Work?**
versioning pinning, **Versioned Modules**-**Versioned Modules**
provisioners
executing scripts, **Provisioners**-**Provisioners**
with null_resource, **Provisioners with null_resource**-**Provisioners with null_resource**
User Data scripts versus, **Provisioners**
provisioning tools, **Provisioning Tools**
combining with configuration management, **Provisioning plus configuration management**
combining with orchestration and server templating, **Provisioning plus server templating plus orchestration**-**Provisioning plus server templating plus orchestration**
combining with server templating, **Provisioning plus server templating**-**Provisioning plus server templating**
configuration management tools versus, **Configuration Management Versus Provisioning**
public access to S3 buckets, blocking, **Shared Storage for State Files**
public keys, **The Way You Store Secrets**
public subnets, **Deploying a Single Web Server**

Public Terraform Registry, [Versioned Modules](#)-[Versioned Modules](#)
publishing modules, [Versioned Modules](#)-[Versioned Modules](#)
pull requests, [Submit Changes for Review](#)
Pulumi, comparison with other IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-Provisioning plus server templating plus orchestration
Puppet, comparison with other IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-Provisioning plus server templating plus orchestration

R

random integers, generating, [count](#) and [for_each](#) Have Limitations
RDS (Relational Database Service), [The terraform_remote_state Data Source](#)
README files, [Documentation](#)
reconciliation loops, [A Crash Course on Kubernetes](#)
refactoring
Terraform code, [Refactoring Can Be Tricky](#)-[Refactoring Can Be Tricky](#)
web server clusters, [Small Modules](#)-[Small Modules](#)
references, [Deploying a Single Web Server](#)
region parameter, [Shared Storage for State Files](#)
regions (in AWS), [Deploying a Single Server](#), [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)
Rego, [Plan testing](#)-[Plan testing](#)
Relational Database Service (RDS), [The terraform_remote_state Data Source](#)

relative filepaths, [File Paths](#)

releasing code

- in application code workflow, [Merge and Release](#)
- in infrastructure code workflow, [Merge and Release-Merge and Release](#)

remote backends

- limitations of, [Limitations with Terraform's Backends-Limitations with Terraform's Backends](#)
- shared storage with, [Shared Storage for State Files-Shared Storage for State Files](#)

remote procedure calls (RPCs), [What Is a Provider?](#)

removing resources, [Cleanup-Cleanup](#)

replicas, [Orchestration Tools](#)

replicating data sources in multiple AWS regions, [Working with Multiple AWS Regions-Working with Multiple AWS Regions](#)

repositories

- sharing, [Deploying a Single Server](#)
- SSH authentication for, [Module Versioning](#)
- for version control, [Live repo and modules repo-Live repo and modules repo](#)

required_providers block, [How Do You Install Providers?-How Do You Install Providers?](#)

required_version parameter, [Versioned Modules](#)

resource attribute references, [Deploying a Single Web Server](#)

resource dependencies, [Isolation via File Layout](#)

resources

arrays of, [Loops with the count Parameter](#)

creating, [Deploying a Single Server](#)

creating out of band, [Valid Plans Can Fail](#)

deploying in multiple AWS regions, [Working with Multiple AWS Regions](#)-[Working with Multiple AWS Regions](#)

lifecycle settings, [Deploying a Cluster of Web Servers](#)

maps of, [Loops with for_each Expressions](#)

naming, [Deploying a Single Server](#)

null_resource, [Provisioners with null_resource](#)-[Provisioners with null_resource](#)

removing, [Cleanup](#)-[Cleanup](#)

secrets management, [Resources and Data Sources](#)

 with encrypted files, [Encrypted files](#)-[Encrypted files](#)

 with environment variables, [Environment variables](#)-[Environment variables](#)

 with secret stores, [Secret stores](#)-[Secret stores](#)

separate resources versus inline blocks, [Inline Blocks](#)-[Inline Blocks](#)

tagging standards, [Loops with for_each Expressions](#)

resources for information, [Recommended Reading](#)-[Online Forums](#)

retries

 in integration tests, [Retries](#)-[Retries](#)

with known errors, [Deployment strategies](#)

return on investment for IaC (infrastructure as code), [Convince Your Boss-Convince Your Boss](#)

reusability of code, [What Are the Benefits of Infrastructure as Code?](#)

reusable modules, [Creating Modules That Can Work with Multiple Providers](#)

creating, [Module Basics-Module Basics](#)

for multiple providers, [Creating Modules That Can Work with Multiple Providers-Creating Modules That Can Work with Multiple Providers](#)

reviewing code changes

in application code workflow, [Submit Changes for Review](#)

for infrastructure code, [Submit Changes for Review-Style guide](#)

rolling deployments, [Configuration Management Tools, Deployment strategies-Deployment strategies](#)

root accounts, [Working with Multiple AWS Accounts](#)

root modules, [Module Basics, Creating Modules That Can Work with Multiple Providers](#)

root users, [Setting Up Your AWS Account](#)

port numbers and, [Deploying a Single Web Server](#)

RPCs (remote procedure calls), [What Is a Provider?](#)

running

Kubernetes, [A Crash Course on Kubernetes](#)

parallel unit tests, [Running tests in parallel-Running tests in parallel](#)

S

S3 buckets

creating, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)
limitations of remote backends, [Limitations with Terraform's Backends](#)-[Limitations with Terraform's Backends](#)

public access, blocking, [Shared Storage for State Files](#)
server-side encryption, enabling, [Shared Storage for State Files](#)
versioning, enabling, [Shared Storage for State Files](#)

sandbox environments, [Manual Testing Basics](#)-[Cleaning Up After Tests](#),
[Unit testing Terraform code](#), [Run the Code Locally](#)

scheduled actions, [Module Outputs](#), If-statements with the count parameter
scripts

for deployment, [Deployment tooling](#), [Deployment tooling](#)
executing

for external data source, [External data source](#)-[External data source](#)

with provisioners, [Provisioners](#)-[Provisioners](#)

Secret Access Keys (in AWS), [Setting Up Your AWS Account](#), [Installing Terraform](#)

secret stores, secrets management with, [Secret stores](#)-[Secret stores](#)
secrets

access interface for, [The Interface You Use to Access Secrets](#)
avoiding in plain text, [Secret Management Basics](#)-[Secret Management Basics](#)

passing via environment variables, [The terraform_remote_state Data Source](#)

with remote backends, [Shared Storage for State Files](#)

storage methods, [The Way You Store Secrets-The Way You Store Secrets](#)

types of, [The Types of Secrets You Store-The Types of Secrets You Store](#)

version control disadvantages, [Shared Storage for State Files](#)

secrets management

CI/CD workflow

CircleCI, [CircleCI as a CI server, with stored secrets-CircleCI as a CI server, with stored secrets](#)

EC2 Instances with IAM roles, [EC2 Instance running Jenkins as a CI server, with IAM roles-EC2 Instance running Jenkins as a CI server, with IAM roles](#)

GitHub Actions, [GitHub Actions as a CI server, with OIDC-GitHub Actions as a CI server, with OIDC](#)

multiple AWS accounts, [Working with Multiple AWS Accounts, Working with Multiple AWS Accounts-Working with Multiple AWS Accounts](#)

for plan files, [Plan files-Plan files](#)

for providers, [Providers-Providers](#)

human users and, [Human users-Human users](#)

machine users and, [Machine users-GitHub Actions as a CI server, with OIDC, Conclusion](#)

for resources and data sources, [Resources and Data Sources](#)
with encrypted files, [Encrypted files-Encrypted files](#)
with environment variables, [Environment variables-Environment variables](#)
with secret stores, [Secret stores-Secret stores](#)
rules of, [Secret Management Basics-Secret Management Basics](#)
for state files, [State files-State files](#)
Terraform authentication options, [Installing Terraform](#)
tool comparison, [A Comparison of Secret Management Tools-A Comparison of Secret Management Tools](#)
security (see network security; secrets management)
security groups, creating, [Deploying a Single Web Server](#), [Deploying a Load Balancer](#)
selector block, [A Crash Course on Kubernetes](#)
self-validating modules, [Testable Modules](#)
precondition/postcondition blocks in, [Preconditions and postconditions-When to use validations, preconditions, and postconditions](#)
validation blocks in, [Validations-Validations](#)
semantic versioning, [Module Versioning](#)
sensitive parameter, [Deploying a Configurable Web Server](#), [Deploying a Configurable Web Server](#)
separate resources versus inline blocks, [Inline Blocks-Inline Blocks](#)
server templating tools, [Server Templating Tools-Server Templating Tools](#)

combining with orchestration and provisioning, **Provisioning plus server templating plus orchestration**-**Provisioning plus server templating plus orchestration**

combining with provisioning, **Provisioning plus server templating**
server testing, **Server testing**-**Server testing**
server-side encryption, enabling, **Shared Storage for State Files**
servers, deploying

cluster of web servers, **Deploying a Cluster of Web Servers**-**Deploying a Cluster of Web Servers**

configurable web servers, **Deploying a Configurable Web Server**-
Deploying a Configurable Web Server

single servers, **Deploying a Single Server**-**Deploying a Single Server**
web servers, **Deploying a Single Web Server**-**Deploying a Single Web Server**

service discovery, **Orchestration Tools**

shared storage for state files, **Shared Storage for State Files**-**Shared Storage for State Files**

with remote backends, **Shared Storage for State Files**-**Shared Storage for State Files**

version control disadvantages, **Shared Storage for State Files**-**Shared Storage for State Files**

sharing Git repositories with teammates, **Deploying a Single Server**

small modules for production-grade infrastructure, **Small Modules**-**Small Modules**

snowflake servers, **What Is DevOps?**

software delivery, [Why Terraform](#)

sops, [Encrypted files](#)

splat expression, [Loops with the count Parameter](#)

Spolsky, Joel, [Use Version Control](#)

SSH authentication for Git repositories, [Module Versioning](#)

stages of integration tests, [Test stages](#)-[Test stages](#)

state files

explained, [What Is Terraform State?](#)-[What Is Terraform State?](#)

importing infrastructure, [Valid Plans Can Fail](#)

isolating, [What Is Terraform State?](#)

purpose of, [State File Isolation](#)-[State File Isolation](#)

via file layout, [Isolation via File Layout](#)-[Isolation via File Layout](#)

via workspaces, [Isolation via Workspaces](#)-[Isolation via Workspaces](#)

locking, [What Is Terraform State?](#)

as private API, [What Is Terraform State?](#)

secrets management, [State files](#)-[State files](#)

shared storage for, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)

with remote backends, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)

version control disadvantages, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)

`terraform_remote_state` data source, [The `terraform_remote_state` Data Source](#)-[The `terraform_remote_state` Data Source](#)

static analysis, [Static analysis](#)-[Static analysis](#)

storing secrets, [The Way You Store Secrets](#)-[The Way You Store Secrets](#)

string directives, [Loops with the for String Directive](#)

strip markers, [Conditionals with the if String Directive](#)

structural types, [Deploying a Configurable Web Server](#)

style guidelines, [Style guide](#)

subnets

data sources for, [Deploying a Cluster of Web Servers](#)-[Deploying a Cluster of Web Servers](#)

of VPCs, [Deploying a Single Web Server](#)

subnets parameter, [Deploying a Load Balancer](#)

subnet_ids parameter, [Deploying a Cluster of Web Servers](#)

T

tagging standards, enforcing, [Loops with for_each Expressions](#)

target groups, creating, [Deploying a Load Balancer](#)

TDD (Test-Driven Development), [Testable Modules](#)

team adoption of IaC

incrementalism in, [Work Incrementally](#)-[Work Incrementally](#)

learning curve, [Give Your Team the Time to Learn](#)-[Give Your Team the Time to Learn](#)

return on investment, [Convince Your Boss](#)-[Convince Your Boss](#)

templatefile built-in function, [The terraform_remote_state Data Source](#)-[The terraform_remote_state Data Source](#), [File Paths](#)

ternary syntax, [Isolation via Workspaces](#), If-statements with the count parameter, If-else-statements with the count parameter

Terraform

authentication options, [Installing Terraform](#)

changes, 2017-2019, [Changes from the First Edition to the Second Edition](#)-[Changes from the First Edition to the Second Edition](#)

comparison with other IaC tools, [How Does Terraform Compare to Other IaC Tools?](#)-Provisioning plus server templating plus orchestration

agent versus agentless, [Agent Versus Agentless](#)-[Agent Versus Agentless](#)

combining tools, [Use of Multiple Tools Together](#)-Provisioning plus server templating plus orchestration

configuration management versus provisioning, [Configuration Management Versus Provisioning](#)

general-purpose versus domain-specific language, [General-Purpose Language Versus Domain-Specific Language](#)-[General-Purpose Language Versus Domain-Specific Language](#)

large versus small community, [Large Community Versus Small Community](#)-[Large Community Versus Small Community](#)

master versus masterless, [Master Versus Masterless](#)-[Master Versus Masterless](#)

mature versus cutting edge, [Mature Versus Cutting Edge](#)

mutable versus immutable infrastructure, [Mutable Infrastructure Versus Immutable Infrastructure](#)-[Mutable Infrastructure Versus Immutable Infrastructure](#)

paid versus free, [Paid Versus Free Offering](#)-[Paid Versus Free Offering](#)

procedural versus declarative language, [Procedural Language Versus Declarative Language](#)-[Procedural Language Versus Declarative Language](#)

core, [What Is a Provider?](#), [Versioned Modules](#)-[Versioned Modules](#)

as declarative language, [Terraform Tips and Tricks: Loops, If-Statements, Deployment, and Gotchas](#)

as deployment tool, [Deployment tooling](#)

documentation, [What You Won't Find in This Book](#), [Deploying a Single Server](#)

[Golden Rule of Terraform](#), [The Golden Rule of Terraform](#)-[The Golden Rule of Terraform](#)

installing, [Installing Terraform](#)-[Installing Terraform](#), [Versioned Modules](#)

new features, [Changes from the Second Edition to the Third Edition](#)-[Changes from the Second Edition to the Third Edition](#)

operational overview, [How Does Terraform Work?](#)-[How Does Terraform Work?](#)

purpose of, [Preface](#)

refactoring, [Refactoring Can Be Tricky](#)-[Refactoring Can Be Tricky](#)

transparent portability, [How Does Terraform Work?](#)

version used, [Open Source Code Examples](#)

terraform apply command, [Deploying a Single Server](#)

Terraform Cloud, [Deployment tooling](#)

Terraform configurations, [How Does Terraform Work?](#)

terraform console command, [The terraform_remote_state Data Source](#)

terraform destroy command, [Cleanup](#)

Terraform Enterprise, [Deployment tooling](#)

terraform fmt command, [Style guide](#)

terraform graph command, [Deploying a Single Web Server](#)

terraform import command, [Valid Plans Can Fail](#)

terraform init command, [Deploying a Single Server, Shared Storage for State Files](#)

terraform output command, [Deploying a Configurable Web Server](#)

terraform plan command, [Deploying a Single Server, Valid Plans Can Fail-Valid Plans Can Fail, Plan testing-Plan testing, Run Automated Tests](#)

Terraform state files (see state files)

terraform validate command, [Static analysis](#)

terraform version command, [Versioned Modules](#)

terraform workspace list command, [Isolation via Workspaces](#)

terraform workspace new command, [Isolation via Workspaces](#)

terraform workspace select command, [Isolation via Workspaces](#)

terraform workspace show command, [Isolation via Workspaces](#)

terraform_remote_state data source, [The terraform_remote_state Data Source-The terraform_remote_state Data Source](#)

Terragrunt, Limitations with Terraform's Backends, Deployment tooling

installing, Promote artifacts across environments

promotion across environments, Promote artifacts across environments-Promote artifacts across environments

tgswitch, Versioned Modules

Terratest, Working with Multiple Different Providers

integration tests with, Test stages-Test stages

plan tests with, Plan testing-Plan testing

retries in integration tests, Retries-Retries

unit tests with, Unit testing Terraform code-Unit testing Terraform code

versions of, Unit testing Terraform code

test doubles, Automated Tests

test pyramid, End-to-End Tests

Test-Driven Development (TDD), Testable Modules

testable modules for production-grade infrastructure, Testable Modules-When to use validations, preconditions, and postconditions

testing

automated tests, Automated tests

in application code workflow, Run Automated Tests

end-to-end tests, End-to-End Tests-End-to-End Tests

in infrastructure code workflow, Run Automated Tests-Run Automated Tests

integration tests, Integration Tests-Retries

plan testing, Plan testing-Plan testing

purpose of, Automated Tests

server testing, Server testing-Server testing

static analysis, Static analysis-Static analysis

types of, Automated Tests-Automated Tests

unit tests, Unit Tests-Running tests in parallel

when to use, When to use validations, preconditions, and postconditions

best practices, Conclusion

comparison of approaches, Conclusion

manual tests

cleanup, Cleaning Up After Tests-Cleaning Up After Tests

explained, Manual Testing Basics-Manual Testing Basics

Ruby comparison, Manual Tests-Manual Tests

tfenv, Versioned Modules-Versioned Modules

tgswitch, Versioned Modules

time estimates for production-grade infrastructure, Production-Grade Terraform Code-Why It Takes So Long to Build Production-Grade Infrastructure

transparent portability, How Does Terraform Work?

type constraints, Deploying a Configurable Web Server

type parameter, Deploying a Configurable Web Server

Ubuntu, A Crash Course on Docker

UI (user interface), The Interface You Use to Access Secrets

unit tests

dependency injection, Dependency injection-Dependency injection purpose of, Automated Tests

Ruby comparison, Unit Tests-Unit Tests

running in parallel, Running tests in parallel-Running tests in parallel with Terratest, Unit testing Terraform code-Unit testing Terraform code

units, defined, Automated Tests

User Data scripts versus provisioners, Provisioners

user interface (UI), The Interface You Use to Access Secrets

user space, Server Templating Tools

user_data parameter, Provisioning Tools, Deploying a Single Web Server

user_data_replace_on_change parameter, Deploying a Single Web Server

V

Vagrant, purpose of, Server Templating Tools

validating

code, What Are the Benefits of Infrastructure as Code?

infrastructure, Manual Testing Basics

validation blocks, Validations-Validations

validation parameter, Deploying a Configurable Web Server

values built-in function, [Loops with for_each Expressions](#)
variable references, [Deploying a Configurable Web Server](#)
variables, [Module Inputs](#)

(see also input variables; output variables)

declaring, [Deploying a Configurable Web Server](#)-[Deploying a Configurable Web Server](#)

version control, [Deploying a Single Server](#)

in application code workflow, [Use Version Control](#)

disadvantages of, [Shared Storage for State Files](#)-[Shared Storage for State Files](#)

importance of, [What Are the Benefits of Infrastructure as Code?](#)

in infrastructure code workflow, [Use Version Control](#)-[The trouble with branches](#)

secrets in, [Secret Management Basics](#)

versioning

in Amazon S3, [Shared Storage for State Files](#), [Shared Storage for State Files](#)

modules, [Module Versioning](#)-[Module Versioning](#)

for production-grade infrastructure, [Versioned Modules](#)-[Versioned Modules](#)

versioning pinning, [Versioned Modules](#)-[Versioned Modules](#)

VM images, [Server Templating Tools](#)

VMs (virtual machines), [Server Templating Tools](#)

VPCs (virtual private clouds)

Default VPC (in AWS), Setting Up Your AWS Account
network isolation, [Inline Blocks](#)
subnets, [Deploying a Single Web Server](#)

W

web servers

deploying, [Deploying a Single Web Server](#)-[Deploying a Single Web Server](#)

cluster of web servers, [Deploying a Cluster of Web Servers](#)-[Deploying a Cluster of Web Servers](#)

configurable web servers, [Deploying a Configurable Web Server](#)-[Deploying a Configurable Web Server](#)

refactoring, [Small Modules](#)-[Small Modules](#)

worker nodes (Kubernetes), [A Crash Course on Kubernetes](#), [Deploying Docker Containers in AWS Using Elastic Kubernetes Service](#)

workflows (see CI/CD workflow)

workspaces

isolating state files, [Isolation via Workspaces](#)-[Isolation via Workspaces](#)

limitations of, [Isolation via Workspaces](#)

Y

yak shaving, [Why It Takes So Long to Build Production-Grade Infrastructure](#)-[Why It Takes So Long to Build Production-Grade Infrastructure](#)

Z

zero-downtime deployment, **Zero-Downtime Deployment-Zero-Downtime Deployment**

limitations, **Zero-Downtime Deployment Has Limitations-Zero-Downtime Deployment Has Limitations**

About the Author

Yevgeniy (Jim) Brikman loves programming, writing, speaking, traveling, and lifting heavy things. He is the cofounder of Gruntwork, a company that provides DevOps as a Service. He's also the author of another book published by O'Reilly Media called *Hello, Startup: A Programmer's Guide to Building Products, Technologies, and Teams*. Previously, he worked as a software engineer at LinkedIn, TripAdvisor, Cisco Systems, and Thomson Financial and got his BS and master's degrees at Cornell University. For more info, check out ybrikman.com.

Colophon

The animal on the cover of *Terraform: Up and Running* is a flying dragon lizard (*Draco volans*), a small reptile so named for its ability to glide using winglike flaps of skin known as *patagia*. The patagia are brightly colored and allow the animal to glide for up to eight meters. The flying dragon lizard is commonly found in many Southeast Asian countries, including Indonesia, Vietnam, Thailand, the Philippines, and Singapore.

Flying dragon lizards feed on insects and can grow to more than 20 centimeters in length. They live primarily in forested regions, gliding from tree to tree to find prey and avoid predators. Females descend from the trees only to lay their eggs in hidden holes in the ground. The males are highly territorial and will chase rivals from tree to tree.

Although once thought to be poisonous, flying dragon lizards pose no threat to humans and are sometimes kept as pets. They are not currently threatened or endangered. Many of the animals on O'Reilly covers are endangered; all of them are important to the world.

The cover illustration is by Karen Montgomery, based on an antique line engraving from *Johnson's Natural History*. The cover fonts are Gilroy Semibold and Guardian Sans. The text font is Adobe Minion Pro; the heading font is Adobe Myriad Condensed; and the code font is Dalton Maag's Ubuntu Mono.