PRODUCTION HANDBOOK & CERTIFICATION PREP

Terraform & OpenTofu Complete Production Course & Associate 003 Study Notes

From Beginner to Master: Hands-on HCL Architecture, State Encryption, Pulumi/Crossplane Migration Matrices, and HashiCorp Certified Associate 003 Exam Objectives.

Author: Naveed Ahmed

Role: Lead DevOps & Platform Architect

Web Ecosystem: naveedkumbhar.com

Release Edition: 2026 Production Handbook

Format: Free PDF Downloadable Course Notes

License: Open Practitioner Resource

1. Executive Infrastructure as Code Tool Overview

Infrastructure as Code (IaC) has evolved through distinct architectural eras. Starting from imperative shell scripts and configuration managers like Ansible, the industry standardized on declarative reconciliation loops represented by HashiCorp Terraform and its open-source Linux Foundation counterpart, OpenTofu. For teams building cloud-native platform APIs, programming language frameworks (Pulumi) and Kubernetes Custom Resource Definitions (Crossplane) offer alternative paradigms.

> **Key Architecture Takeaway:** Declarative IaC defines the desired end state (e.g. "ensure an S3 bucket exists with AES-256 encryption"), delegating dependency graphing, diff calculation, and CRUD API calls to the engine. Imperative tools describe execution steps, requiring manual state tracking.

Tool	Paradigm	License	State Model	Primary Sweet Spot
Terraform	Declarative HCL	BSL 1.1 / Commercial	JSON State (S3, GCS, Cloud)	Multi-cloud enterprise baselines
OpenTofu	Declarative HCL	MPL-2.0 (Open Source)	Encrypted State (S3, GCS)	Open-source enterprise infrastructure
Pulumi	Imperative Code (TS/Python/Go)	Apache-2.0	Pulumi Service / S3 Backend	Software engineering & dynamic apps
Crossplane	Kubernetes CRD Control Loops	Apache-2.0	etcd (Continuous reconciliation)	Internal Developer Platforms (IDP)
CloudFormation / CDK	Declarative JSON/YAML or CDK	AWS Proprietary	AWS Managed Engine	AWS-exclusive single-provider teams

2. HashiCorp Certified: Terraform Associate (003) Core Syllabus

The HashiCorp Certified: Terraform Associate (003) exam tests foundational concepts, CLI workflows, state management, module authoring, and cloud workspace configurations.

> **Exam Tip (Free Retake & Registration):** The 003 exam consists of 57 multiple-choice and multiple-select questions with a 60-minute duration. Check HashiCorp promo windows or training partner vouchers for free retake offers. Focus heavily on state locking, module inputs/outputs, and provisioner lifecycle rules.

Core Terraform CLI Commands & Lifecycle

```
# 1. Initialize working directory and download provider plugins
terraform init -upgrade

# 2. Format HCL according to canonical style conventions
terraform fmt -recursive

# 3. Validate syntax and internal variable consistency without contacting APIs
terraform validate

# 4. Generate an execution plan and inspect proposed infrastructure changes
terraform plan -out=tfplan.binary

# 5. Apply the execution plan directly to target cloud providers
terraform apply tfplan.binary

# 6. Target specific resources during emergency maintenance
terraform apply -target=aws_security_group.ingress_rules

# 7. Mark a damaged resource for recreation on next apply
terraform taint aws_instance.bastion  # (or terraform apply -replace=aws_instance.bastion)
```

Essential Terraform & OpenTofu Architecture Checklist

- **State Locking:** Always configure DynamoDB with a Partition Key named `LockID` (String) to prevent concurrent state corruption when teams run parallel CI/CD jobs.
- **Remote State Encryption:** OpenTofu introduces native client-side state encryption before bytes reach the S3 bucket, preventing plaintext secrets in state files from exposure.
- **Data Sources:** Use `data "aws_ami" "ubuntu"` or `data "aws_vpc" "default"` for read-only queries against existing cloud infrastructure without taking ownership.
- **Module Composition:** Structure code into Root Modules and Child Modules. Expose configurable attributes via `variables.tf` and emit computed attributes via `outputs.tf`.
- **Sensitive Variables:** Mark sensitive values with `sensitive = true` in variable declarations to suppress output in terminal logs and plan diffs.

3. OpenTofu vs Terraform: Migration & Production Setup

Following HashiCorp's transition of Terraform from the Mozilla Public License v2.0 (MPL-2.0) to the Business Source License (BSL 1.1) in August 2023, the Linux Foundation launched OpenTofu. OpenTofu maintains 100% drop-in compatibility with Terraform 1.5.x configurations while adding key open-source innovations.

Native Client-Side State Encryption in OpenTofu

```
terraform {
  encryption {
    key_provider "pbkdf2" "passphrase" {
      passphrase = var.tofu_encryption_passphrase
    }
    method "aes_gcm" "strong" {
      keys = key_provider.pbkdf2.passphrase
    }
    state {
      method = method.aes_gcm.strong
      enforced = true
    }
  }
}
```

Zero-Downtime Migration Steps (Terraform to OpenTofu)

1. **Pull and verify current state:** Execute `terraform state pull > state-backup.json` to secure an uncorrupted snapshot.
2. **Install OpenTofu binary:** Install via Homebrew (`brew install opentofu`) or Linux package repos.
3. **Reinitialize providers:** Run `tofu init`. OpenTofu will read your existing `backend.tf` and download verified providers from the OpenTofu registry.
4. **Run speculative plan:** Execute `tofu plan`. Verify that zero modifications, zero recreations, and zero destroys appear in the output.

4. CloudFormation, CDK, and Pulumi In-Depth Comparison

Engineering teams frequently evaluate whether to use cloud-native templating (CloudFormation, AWS CDK) or third-party engines (Terraform, Pulumi, Crossplane).

Dimension	AWS CloudFormation	AWS CDK	Pulumi
Authoring Language	JSON / YAML	TypeScript, Python, Go	TypeScript, Python, Go, C#
State Storage	Managed by AWS Engine	Synthesized to CFn Stack	Pulumi SaaS or S3/Blob storage
Multi-Cloud	No (AWS Only)	No (AWS Only)	Yes (AWS, GCP, Azure, K8s)
Loop & Logic Support	Limited intrinsic functions	Full Turing-complete logic	Full Turing-complete logic
Drift Detection	Built-in CloudFormation drift	Via CFn drift detection	`pulumi refresh` CLI detection
StackSets	Cross-account/region deployment	Supported via L2 constructs	Multi-stack orchestration

> **What is a CloudFormation StackSet?** A CloudFormation StackSet extends the functionality of single-region stacks by letting you create, update, or delete stacks across multiple AWS accounts and regions with a single operation. This is critical for enterprise security guardrails, VPC peering baselines, and GuardDuty multi-account configuration.

5. Production Kubernetes Runbooks & Probe Architecture

When orchestrating Kubernetes infrastructure via Terraform or OpenTofu, use the official Helm and Kubernetes providers alongside properly configured container health probes.

```
# Deploying microservices with OpenTofu and Helm provider
provider "helm" {
  kubernetes {
    config_path = "~/.kube/config"
  }
}

resource "helm_release" "ingress_controller" {
  name       = "ingress-nginx"
  repository = "https://kubernetes.github.io/ingress-nginx"
  chart      = "ingress-nginx"
  namespace  = "kube-system"

  set {
```

```
    name  = "controller.replicaCount"
    value = "3"
  }
}
```

Kubernetes Startup Probe Configuration

```
startupProbe:
  httpGet:
    path: /healthz
    port: 8080
  failureThreshold: 30
  periodSeconds: 10   # Gives JVM / Node.js up to 300s to initialize before liveness kills it
```

> **Key Takeaway:** Never increase `initialDelaySeconds` on a `livenessProbe` to compensate for slow startup times. Configure a dedicated `startupProbe` so the pod is granted a grace period to compile bytecode and run database migrations without sacrificing rapid crash loop detection during normal runtime.

6. Complete AWS Cutover Runbook Checklist

When executing a production cutover from legacy infrastructure to a newly provisioned Terraform/OpenTofu cloud estate, include these mandatory sections in your operational runbook:

1. **Pre-Cutover Verification & Dry Run:** Audit DNS TTL values 72 hours prior (lower TTL to 60 seconds), verify database replica lag, and execute automated smoke tests on staging.
2. **Access & Communication Channels:** Establish bridge conference line, designate Incident Commander, Communications Lead, and Database Custodian, and alert external payment gateway vendors.
3. **Data Synchronization & Freeze:** Put application into maintenance mode, execute final delta replication on transactional databases, and verify row count checksums between source and target.
4. **DNS Cutover & Routing Shift:** Update Amazon Route 53 weighted record sets or Alias targets to point to the new Application Load Balancers.
5. **Post-Cutover Smoke Tests & SLO Monitoring:** Run end-to-end user journeys (login, checkout, token exchange) and monitor latency, 5xx rates, and database CPU saturation for 60 minutes.
6. **Rollback Triggers & Emergency Fallback:** Pre-define non-negotiable rollback thresholds (e.g., error rate > 1% for 5 consecutive minutes or data inconsistency discovered). Maintain reverse replication to the original database until cutover window closes.

Naveed Ahmed — DevOps & Platform Architecture Hub

Explore 950+ production incident scenarios, free interactive tools, and architectural guides at naveedkumbhar.com