# Infrastructure as Code (IaC) CI/CD Pipeline with Jenkins and Terraform

## 1. Overview

This document provides a detailed explanation of a Jenkins-based CI/CD pipeline for deploying Terraform-managed AWS infrastructure.   
It integrates automated validation, security scanning, and compliance testing for improved DevSecOps practices.

## 2. Repository Structure

├── Jenkinsfile  
├── README.md  
├── environments  
│ ├── dev/  
│ ├── staging/  
│ └── prod/  
├── modules  
│ ├── eks/  
│ ├── vpc/  
│ └── s3-backend/

## 3. Tools & Integrations

|  |  |
| --- | --- |
| Tool | Purpose |
| Terraform | Infrastructure provisioning on AWS |
| TruffleHog | Secret scanning for sensitive credentials |
| Checkov | IaC (Terraform) misconfiguration detection |
| Conftest (OPA) | Policy enforcement and compliance testing |
| AWS CLI | Interaction with AWS services |
| Jenkins | CI/CD pipeline automation |

## 4. Jenkins Pipeline Flow

### Checkout

Fetches the source code from Git.

### TruffleHog Secret Scan

Scans repository for secrets.

### Checkov IaC Security Scan

Performs IaC misconfiguration scan.

### OPA Policy Compliance

Runs compliance checks using Conftest.

### Terraform Init

Initializes Terraform backend.

### Terraform Validate

Validates Terraform syntax.

### Terraform Plan

Prepares an execution plan for review.

### Manual Approval

Pauses for manual approval before applying changes.

### Terraform Apply

Applies Terraform configurations to AWS.

## 5. Generated Artifacts

After each pipeline run, the following artifacts are archived for auditing and review:

|  |  |
| --- | --- |
| Artifact | Description |
| trufflehog-report.json | Secret scanning report |
| checkov-report.json | IaC security report |
| opa-report.json | OPA policy compliance report |
| tfplan.txt | Terraform execution plan output |
| outputs.json | Deployed resource outputs |

## 6. Screenshots / Observations

Git Code Tree

A screenshot of a computer

AI-generated content may be incorrect.

Parameterization for Environment:

parameters {

choice(

name: 'DEPLOY\_ENV',

choices: ['dev', 'staging', 'prod'],

description: 'Select the environment to deploy to'

)

}

A screenshot of a computer

AI-generated content may be incorrect.

**TruffleHog – Secret Detection on the code base**

A screenshot of a computer

AI-generated content may be incorrect.

stage('TruffleHog - Secret Scan') {

steps {

echo " Running TruffleHog secret scan..."

sh '''

docker run --rm -v $(pwd):/repo ghcr.io/trufflesecurity/trufflehog:latest \

filesystem /repo --fail --json > trufflehog-report.json || echo "Secrets found — check report"

'''

}

Running Chkov

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Terraform Init,Plan and Apply

/\* -----------------------------

\* TERRAFORM INIT

\* ----------------------------- \*/

stage('Terraform Init') {

steps {

echo "🔧 Initializing Terraform in ${TF\_WORKDIR}..."

dir("${TF\_WORKDIR}") {

withCredentials([[$class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-creds']]) {

sh 'terraform init -input=false'

}

}

}

}

A screenshot of a computer

AI-generated content may be incorrect.

/\* -----------------------------

\* TERRAFORM VALIDATE

\* ----------------------------- \*/

stage('Terraform Validate') {

steps {

echo "Validating Terraform configuration..."

dir("${TF\_WORKDIR}") {

sh 'terraform validate'

}

}

}

/\* -----------------------------

\* TERRAFORM PLAN

\* ----------------------------- \*/

stage('Terraform Plan') {

steps {

echo " Running Terraform plan for ${DEPLOY\_ENV}..."

dir("${TF\_WORKDIR}") {

withCredentials([[$class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-creds']]) {

sh '''

terraform plan -out=tfplan -input=false

terraform show -no-color tfplan > tfplan.txt

'''

}

}

}

}

A screenshot of a computer

AI-generated content may be incorrect.

/\* -----------------------------

\* MANUAL APPROVAL

\* ----------------------------- \*/

stage('Manual Approval') {

steps {

input message: " Approve deployment to ${params.DEPLOY\_ENV} environment?"

}

}

A screenshot of a computer

AI-generated content may be incorrect.

/\* -----------------------------

\* TERRAFORM APPLY (TOLERANT)

\* ----------------------------- \*/

stage('Terraform Apply') {

steps {

echo " Applying Terraform changes to ${DEPLOY\_ENV}..."

dir("${TF\_WORKDIR}") {

withCredentials([[$class: 'AmazonWebServicesCredentialsBinding', credentialsId: 'aws-creds']]) {

// The key part: || true ensures Jenkins doesn’t fail even if apply errors

sh '''

set +e

terraform apply -auto-approve tfplan || echo " Terraform apply failed at node\_group creation due to the AWS free tiering, but continuing..."

terraform output -json > outputs.json || true

set -e

'''

}

}

}

}

}

**Artificats**

A screenshot of a computer

AI-generated content may be incorrect.

Note: Here , As I’m using aws free tier account, during the EKS-ClusterCreation->NodeGroup Creation, ive started to receive an error that “ free tier supported instance does not support by the instance that supported by EKS node group”

All the similar code that I attached will create the rest of all the stack in free tier AWS cloud Account.

# Resource created by the Terraform code

As I have created resource for Dev and Stage at the same time before destroying , the resource might be shown two times.

VPC:  
A computer screen shot of a computer screen

AI-generated content may be incorrect.

### Subnet

A screenshot of a computer

AI-generated content may be incorrect.

Route table

A screenshot of a computer

AI-generated content may be incorrect.

NAT Gateway

A screenshot of a computer

AI-generated content may be incorrect.

Internet Gateway

A screenshot of a computer

AI-generated content may be incorrect.

S3 bucket for Terraform Statfile lock

A computer screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Elastic Kubernetes Service  
Dev and Staging Cluster

A screenshot of a computer

AI-generated content may be incorrect.

### Cluster IAM role

A screenshot of a computer

AI-generated content may be incorrect.

Node Group Creation Fail because of Free tier Limitation

A screenshot of a computer

AI-generated content may be incorrect.

## 7. Summary

This pipeline ensures secure, policy-compliant, and automated Terraform deployments by integrating DevSecOps best practices with Jenkins.  
It covers secret detection, IaC security scanning, compliance validation, and controlled infrastructure deployment.