Palo Alto Networks:

Terraform Demo Document

Terraform: Infrastructure Demo

Date: April 1st 2019

Author: Sumeet Chaurasia

Contents

**1. Terraform Introduction 3**

**2. Architecture overview** **4**

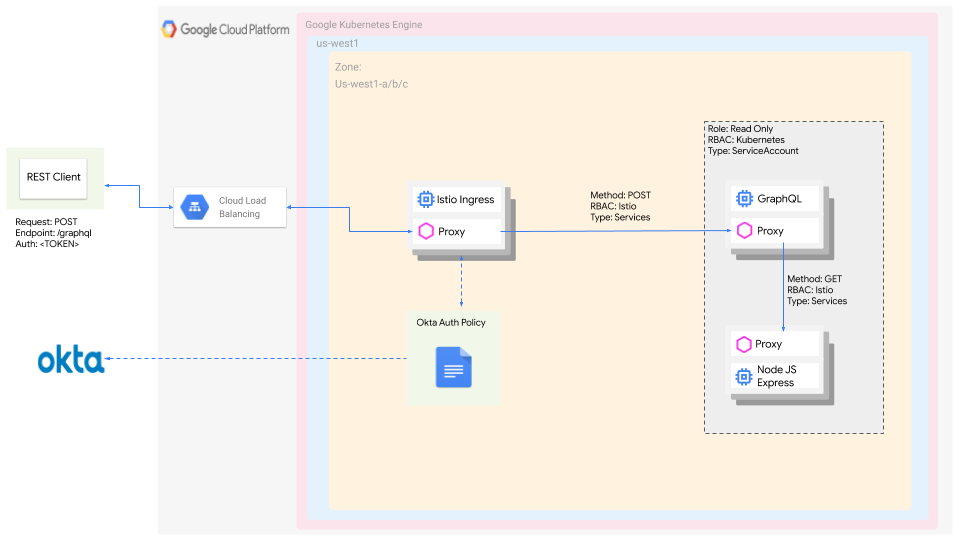
**3. Demo Description 5**

# 1. Terraform Introduction

Terraform is a tool for building, changing, and versioning infrastructure safely and efficiently. Configuration files describe to Terraform the components needed to run a single application or your entire datacenter. Terraform generates an execution plan describing what it will do to reach the desired state, and then executes it to build the described infrastructure. As the configuration changes, Terraform is able to determine what changed and create incremental execution plans which can be applied.

This document describes the output of terraform template and a high level architecture of what it will create, for more formal introduction of terraform please follow the terraform [documentation](https://www.terraform.io/intro/index.html).

# 2. Architecture overview

****

**REST Client:** Used by end user to make HTTPS POST request at endpoint ‘/graphql’ with valid jwt token to access the microservice deployed in GKE cluster via Istio’s ingress gateway.

**OKTA:** Demo OKTA account which is used for validating the jwt token, the policy in Istio currently validates the token with our OKTA instance.

# 3. Demo Description

**Objective**: Deploy microservices on GKE cluster with OKTA authentication using Terraform

The demonstration consists of two microservices(Graphql, and Express) with Terraform template to provision a GKE cluster on a separate subnet and deploy the microservices.

Follow the Readme file in terraform folder to setup the environment. The terraform template will create below resources:

1. A VPC network with a subnet “private google access” and “flow logs” enabled
2. A GKE cluster in the same VPC and subnet with Istio features enabled
3. Once the resources are created, terraform will dockerize the microservices, push it into the Google's Container Registry and deploys it in the freshly created GKE Cluster with proper Role Based Access Control on Kubernetes pods and Okta policy on Istio’s layer.

The two microservices consists of the GraphQL and Node JS Express, for our demonstration purpose the GraphQL microservice is responsible to communicate with the Node Express application via HTTP GET to publish a message as a response. The microservices communicate with each other via Istio’s sidecar proxy in GKE cluster. To access the application a suitable REST client could be used which has the capability to make a REST HTTP call with the GraphQL body.

As a part of OKTA authentication, every request to Istio’s ingress gateway will be authenticated by OKTA with the token provided by the end user. With the RBAC on Istio’s layer, it will only allow HTTPS POST as a request. The RBAC on Kubernetes restricts pods with only “Read Only” permission for other Pods and Services.