

**GCP – HOL -Session 16**

**Create GCP Resources using Terraform**

This HOL session will allow you to create and manage projects on Google Cloud Platform with Terraform. The state that Terraform generates is saved to Google Cloud Storage for persistence.

Objectives

* Create a Terraform Admin Project for the service account and remote state bucket.
* Grant Organization-level permissions to the service account.
* Configure the remote state in GCS.
* Use Terraform to provision a new project and an instance in that project.

export TF\_VAR\_org\_id=YOUR\_ORG\_ID  
export TF\_VAR\_billing\_account=YOUR\_BILLING\_ACCOUNT\_ID  
export TF\_ADMIN=${USER}-terraform-admin  
export TF\_CREDS=~/.config/gcloud/umathivaproject-terraform-admin.json

**Note**: The TF\_ADMIN variable will be used for the name of the Terraform Admin Project and must be unique.

You can find the values for YOUR\_ORG\_ID and YOUR\_BILLING\_ACCOUNT\_ID using the following commands:

gcloud organizations list  
gcloud beta billing accounts list

## Create the Terraform Admin Project

Using an Admin Project for your Terraform service account keeps the resources needed for managing your projects separate from the actual projects you create. While these resources could be created with Terraform using a service account from an existing project, or using Cloud Shell, in this tutorial you will create a separate project and service account exclusively for Terraform.

Create a new project and link it to your billing account:

gcloud projects create ${TF\_ADMIN} \  
  --organization ${TF\_VAR\_org\_id} \  
  --set-as-default  
  
gcloud beta billing projects link ${TF\_ADMIN} \  
  --billing-account ${TF\_VAR\_billing\_account}

## Create the Terraform service account

Create the service account in the Terraform admin project and download the JSON credentials:

gcloud iam service-accounts create terraform \  
  --display-name "Terraform admin account"  
  
gcloud iam service-accounts keys create ${TF\_CREDS} \  
  --iam-account terraform@${TF\_ADMIN}.iam.gserviceaccount.com

Grant the service account permission to view the Admin Project and manage Cloud Storage:

gcloud projects add-iam-policy-binding ${TF\_ADMIN} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/viewer  
  
gcloud projects add-iam-policy-binding ${TF\_ADMIN} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/storage.admin

Any actions that Terraform performs require that the API be enabled to do so. In this guide, Terraform requires the following:

gcloud services enable cloudresourcemanager.googleapis.com  
gcloud services enable cloudbilling.googleapis.com  
gcloud services enable iam.googleapis.com  
gcloud services enable compute.googleapis.com  
gcloud services enable serviceusage.googleapis.com

### **Add organization/folder-level permissions**

Grant the service account permission to create projects and assign billing accounts:

gcloud organizations add-iam-policy-binding ${TF\_VAR\_org\_id} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/resourcemanager.projectCreator  
  
gcloud organizations add-iam-policy-binding ${TF\_VAR\_org\_id} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/billing.user

## Set up remote state in Cloud Storage

Create the remote backend bucket in Cloud Storage and the backend.tf file for storage of the terraform.tfstate file:

gsutil mb -p ${TF\_ADMIN} gs://${TF\_ADMIN}  
  
cat > backend.tf << EOF  
terraform {  
 backend "gcs" {  
   bucket  = "testproject-terraform-admin”  
   prefix  = "terraform/state"  
 }  
}  
EOF

Enable versioning for said remote bucket:

gsutil versioning set on gs://${TF\_ADMIN}

Configure your environment for the Google Cloud Terraform provider:

export GOOGLE\_APPLICATION\_CREDENTIALS=${TF\_CREDS}  
export GOOGLE\_PROJECT=${TF\_ADMIN}

## Use Terraform to create a new project and Compute Engine instance

The project.tf file:

variable "project\_name" {}  
variable "billing\_account" {}  
variable "org\_id" {}  
variable "region" {}  
  
provider "google" {  
 region = "${var.region}"  
}  
  
resource "random\_id" "id" {  
 byte\_length = 4  
 prefix      = "${var.project\_name}-"  
}  
  
resource "google\_project" "project" {  
 name            = "${var.project\_name}"  
 project\_id      = "${random\_id.id.hex}"  
 billing\_account = "${var.billing\_account}"  
 org\_id          = "${var.org\_id}"  
}  
  
resource "google\_project\_services" "project" {  
 project = "${google\_project.project.project\_id}"  
 services = [  
   "compute.googleapis.com"  
 ]  
}  
  
output "project\_id" {  
 value = "${google\_project.project.project\_id}"  
}

The compute.tf file:

data "google\_compute\_zones" "available" {}  
  
resource "google\_compute\_instance" "default" {  
# project = "${google\_project\_services.project.project}"  
 zone = "${data.google\_compute\_zones.available.names[0]}"  
 name = "tf-compute-1"  
 machine\_type = "f1-micro"  
 boot\_disk {  
   initialize\_params {  
     image = "ubuntu-1604-xenial-v20170328"  
   }  
 }  
 network\_interface {  
   network = "default"  
   access\_config {  
   }  
 }  
}  
  
output "instance\_id" {  
 value = "${google\_compute\_instance.default.self\_link}"  
}

**Finally do this**

export TF\_VAR\_project\_name=${USER}-test-compute  
export **TF\_VAR\_**region=us-central1

Next, initialize the backend:

terraform init

Preview the Terraform changes:

terraform plan

Apply the Terraform changes:

terraform apply

SSH into the instance created:

export instance\_id=$(terraform output instance\_id)  
export project\_id=$(terraform output project\_id)  
  
gcloud compute ssh ${instance\_id} --project ${project\_id}

## Cleaning up

First, destroy the resources created by Terraform:

terraform destroy

Next, delete the Terraform Admin project and all of its resources:

gcloud projects delete ${TF\_ADMIN}

Finally, remove the organization level IAM permissions for the service account:

gcloud organizations remove-iam-policy-binding ${TF\_VAR\_org\_id} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/resourcemanager.projectCreator  
  
gcloud organizations remove-iam-policy-binding ${TF\_VAR\_org\_id} \  
  --member serviceAccount:terraform@${TF\_ADMIN}.iam.gserviceaccount.com \  
  --role roles/billing.user