Community tutorials

CONTRIBUTE A TUTORIAL (/COMMUNITY/TUTORIALS/WRITE)

SEARCH TUTORIALS (/DOCS/OPEN-TUTORIALS)

EDIT ON GITHUB (HTTPS://GITHUB.COM/GOOGLECLOUDPLATFORM/COMMUNITY/EDIT/MASTER/TUTORIALS/MANAGING-GCP-PROJECTS-WITH-TERRAFORM/INDEX.MD)

REPORT ISSUE (HTTPS://GITHUB.COM/GOOGLECLOUDPLATFORM/COMMUNITY/ISSUES/NEW?TITLE=ISSUE%20WITH%20TUTORIALS/MANAGING-GCP-PROJECTS-WITH-TERRAFORM/INDEX.MD&BODY=ISSUE%20DESCRIPTION)

PAGE HISTORY (HTTPS://GITHUB.COM/GOOGLECLOUDPLATFORM/COMMUNITY/COMMITS/MASTER/TUTORIALS/MANAGING-GCP-PROJECTS-WITH-TERRAFORM/INDEX.MD)

Creating Google Cloud projects with Terraform

Author(s): @danisla (https://github.com/danisla), Published: 2017-06-19

Dan Isla | Solution Architect | Google

Contributed by Google employees.

This tutorial demonstrates how to create and manage projects on Google Cloud with Terraform (/docs/terraform). With Terraform, many of your resources such as projects, IAM policies, networks, Compute Engine instances, and Kubernetes Engine clusters can be managed, versioned, and easily recreated for your organization or teams. The state that Terraform generates is saved to Cloud Storage for persistence.

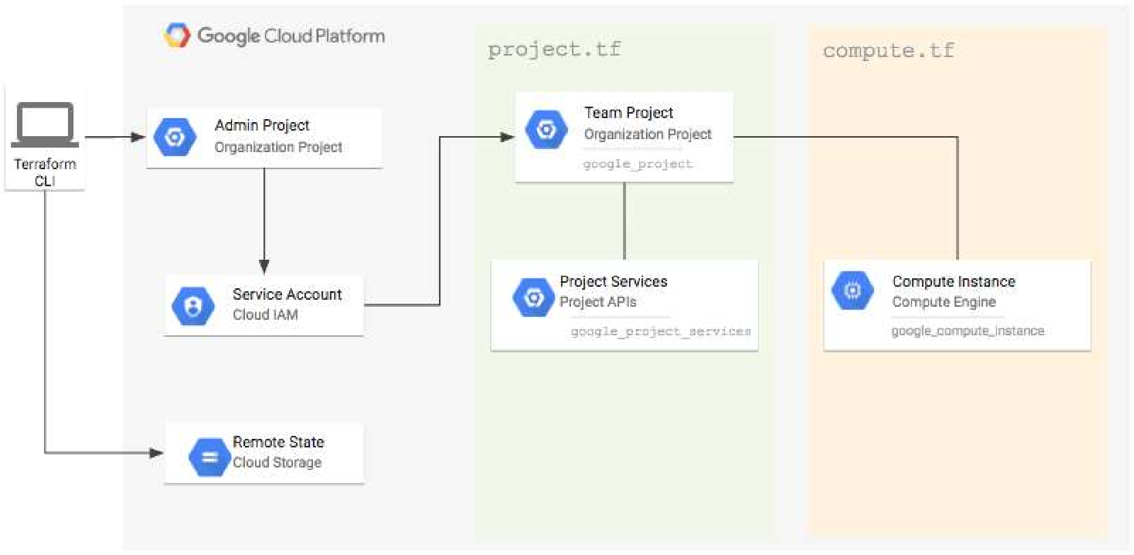
Note: This tutorial is focused on creating projects (/resource-manager/docs/creating-managing-projects). For a general introduction to Terraform on Google Cloud, see the provider documentation (https://www.terraform.io/docs/providers/google/guides/getting_started.html).

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 Cloud Storage.
- Use Terraform to provision a new project and an instance in that project.

Architecture diagram for tutorial components:

Figure 1. *Architecture diagram for tutorial components*



Before you begin

This tutorial assumes that you already have a Google Cloud account set up for your organization and that you are allowed to make organization-level changes in the account. See the documentation (/resource-manager/docs/creating-managing-organization#setting-up) for details on creating and managing organizations.

Commands in this tutorial outside of Terrafrom are run with the Google Cloud SDK (/sdk) `gcloud` command-line tool. This tutorial assumes that you have the `gcloud` tool installed and authorized to work with your account according to the documentation (/sdk/docs/authorizing).

This tutorial requires terraform v0.12.0+ and google_provider 3.0.0+. A previous of version of this tutorial using google_provider 2.x.x is here

(https://github.com/GoogleCloudPlatform/community/tree/af5148120947b493d9a19531a763dac4b02b3e00/tutorials/managing-gcp-projects-with-terraform)
. The current tutorial has been tested with the following:

```
Terraform v0.12.21
+ provider.google v3.11.0
+ provider.random v2.2.1
```

Costs

This tutorial uses billable components of Google Cloud, including the following:

- Compute Engine
- Cloud Storage

Use the Pricing Calculator (/products/calculator#id=cdaa96a1-84a6-468d-b5cc-493af9895149) to generate a cost estimate based on your projected usage.

Set up the environment

Export the following variables to your environment for use throughout the tutorial.

```
export TF_VAR_org_id=YOUR_ORG_ID
export TF_VAR_billing_account=YOUR_BILLING_ACCOUNT_ID
export TF_ADMIN=${USER}-terraform-admin
export TF_CREDS=~/.config/gcloud/${USER}-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.
  --role roles/viewer

gcloud projects add-iam-policy-binding ${TF_ADMIN} \
  --member serviceAccount:terraform@${TF_ADMIN}.iam.gserviceaccount.
  --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.
  --role roles/resourcemanager.projectCreator

gcloud organizations add-iam-policy-binding ${TF_VAR_org_id} \
  --member serviceAccount:terraform@${TF_ADMIN}.iam.gserviceaccount.
  --role roles/billing.user
```

If your billing account is owned by another organization, then make sure the service account email address has been added as a Billing Account User to the billing account permissions.

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  = "${TF_ADMIN}"
   prefix  = "terraform/state"
 }
}
EOF
```

Enable versioning for the 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_service" "service" {
  for_each = toset([
    "compute.googleapis.com"
  ])

  service = each.key

  project            = google_project.project.project_id
  disable_on_destroy = false
}

output "project_id" {
  value = google_project.project.project_id
}
```

View the code on GitHub (https://github.com/GoogleCloudPlatform/community/blob/master/tutorials/managing-gcp-projects-with-terraform/project.tf)
.

Terraform resources used:

- `provider "google"` (https://www.terraform.io/docs/providers/google/index.html): The Google cloud provider config. The credentials will be pulled using the `GOOGLE_APPLICATION_CREDENTIALS` environment variable, set earlier in this tutorial.
- `resource "random_id"` (https://www.terraform.io/docs/providers/random/r/id.html): Project IDs must be unique. Generate a random one prefixed by the desired project ID.
- `resource "google_project"` (https://www.terraform.io/docs/providers/google/r/google_project.html): The new

project to create, bound to the desired organization ID and billing account.

- `resource "google_project_services"` (https://www.terraform.io/docs/providers/google/r/google_project_services.html): Services and APIs enabled within the new project. Note that if you visit the web console after running Terraform, additional APIs may be implicitly enabled and Terraform would become out of sync. Re-running `terraform plan` will show you these changes before Terraform attempts to disable the APIs that were implicitly enabled. You can also set the full set of expected APIs beforehand to avoid the synchronization issue.
- `output "project_id"` (https://www.terraform.io/intro/getting-started/outputs.html): The project ID is randomly generated for uniqueness. Use an output variable to display it after Terraform runs for later reference. The length of the project ID should not exceed 30 characters.

The `compute.tf` file:

```
data "google_compute_zones" "available" {
  project = google_project.project.project_id
}

resource "google_compute_instance" "default" {
  project      = google_project.project.project_id
  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 {}
  }

  depends_on = [google_project_service.service]
}

output "instance_id" {
  value = google_compute_instance.default.self_link
}
```

View the code on GitHub (https://github.com/GoogleCloudPlatform/community/blob/master/tutorials/managing-gcp-projects-with-terraform/compute.tf)
.

Terraform resources used:

- `data "google_compute_zones"` (https://www.terraform.io/docs/providers/google/d/google_compute_zones.html): Data resource used to lookup available Compute Engine zones, bound to the desired region. Avoids hard-coding of zone names.
- `resource "google_compute_instance"` (https://www.terraform.io/docs/providers/google/r/compute_instance.html): The Compute Engine instance bound to the newly created project. Note that the resource depends on `google_project_service.service` resource explicitly. This is to tell Terraform to create it after the Compute Engine API has been enabled. Otherwise, Terraform would try to enable the Compute Engine API and create the instance at the same time, leading to an attempt to create the instance before the Compute Engine API is fully enabled.
- `output "instance_id"` (https://www.terraform.io/intro/getting-started/outputs.html): The `self_link` is output to make it easier to ssh into the instance after Terraform completes.

Set the name of the project you want to create and the region you want to create the resources in:

```
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}
```

Note that SSH may not work unless your organization user also has access to the newly created project resources.

Cleaning up

First, permanently delete 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.
  --role roles/resourcemanager.projectCreator
```

```
gcloud organizations remove-iam-policy-binding ${TF_VAR_org_id} \
  --member serviceAccount:terraform@${TF_ADMIN}.iam.gserviceaccount.
  --role roles/billing.user
```

Submit a tutorial

Share step-by-step guides

SUBMIT A TUTORIAL (/COMMUNITY/TUTORIALS/WRITE)

Request a tutorial

Ask for community help

SUBMIT A REQUEST (HTTPS://GITHUB.COM/GOOGLECLOUDPLATFORM/COMMUNITY/ISSUES?Q=IS%3AOPEN+IS%3AISSUE+LABEL%3A%22TUTORIAL+REQUEST%22)

View tutorials

Search Google Cloud tutorials

VIEW TUTORIALS (/DOCS/OPEN-TUTORIALS)

(/docs/open-tutorials) (/docs/tutorials)