GCP Deployments

Authors:

- Adam Edwards

Overview

This document outlines how Project Echo may deploy and manage resources within Google cloud Platform (GCP).

Problem Statement

Project Echo has a requirement to deploy and manage resources within GCP projects, progressing Project Echo from being only a sandbox running on developer machines.

Considerations

- Quick to set up. Week 5 of Trimester 2 2024 is about to start and work ideally begins as soon as possible.

Requirements

- Infrastructure as code
- IaC tools used that can deploy resources to GCP
- Low or no cost solutions
- Simple enough to quickly be implemented whilst maintaining good practices

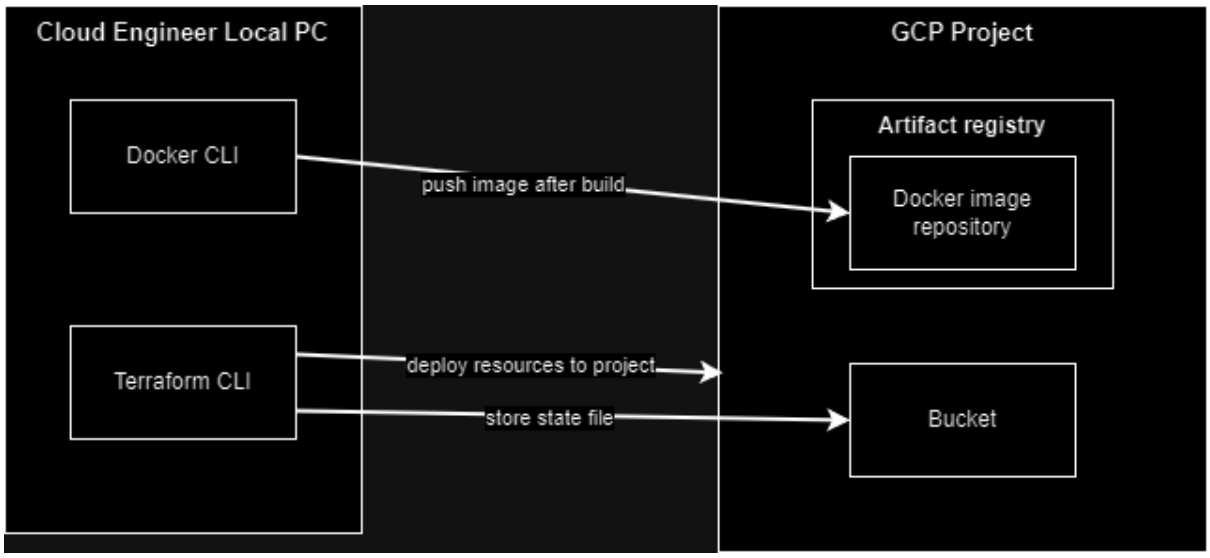
Proposed Method

The proposed method to deploy cloud infrastructure is Infrastructure as Code (IAC) using Terraform CLI and docker from cloud engineer local PCs. Ideally a CICD automation tool would be used but to get started deploying as soon as possible the solution is suggested to remain simple. Further below are suggestions for ideal future states on this exact subject.

On Infrastructure as Code

Manually managing infrastructure is not recommended as there is great risk of error when redeploying resources (especially throughout Change Management processes) and manual deployments require more effort. Using the appropriate IAC tooling is recommended, which reduces the risk of manual error and ensures deployments are easily reproduced.

Flow



Reference Architecture

Google Cloud Platform

Google Cloud Platform (GCP) Is the public cloud provider of choice for the program. Environments will be set up as projects, the resources for Project Echo will be deployed here.

Google Cloud Platform – Artifact Registry

Artifact Registry is used to store built artifacts such as docker images (these go within an Artifact Registry docker repository) that will be required at runtime for Project Echo's services.

Google Cloud Platform – Cloud Storage – Buckets

GCP buckets provide object storage and for the shared usage of Terraform state we require a centralized location to store the state files, these can be stored and shared using a bucket.

Terraform

Terraform, specifically Terraform CLI (there is also Terraform Cloud), is the most used Infrastructure as Code (IaC) tool when it comes to cloud deployments.

Terraform keeps a record of resources that are deployed within state files. These are required to be shared, to prevent conflicting deployments if engineers were to try to deploy the same resources. This can be solved by using a GCP Cloud Storage bucket.

Docker

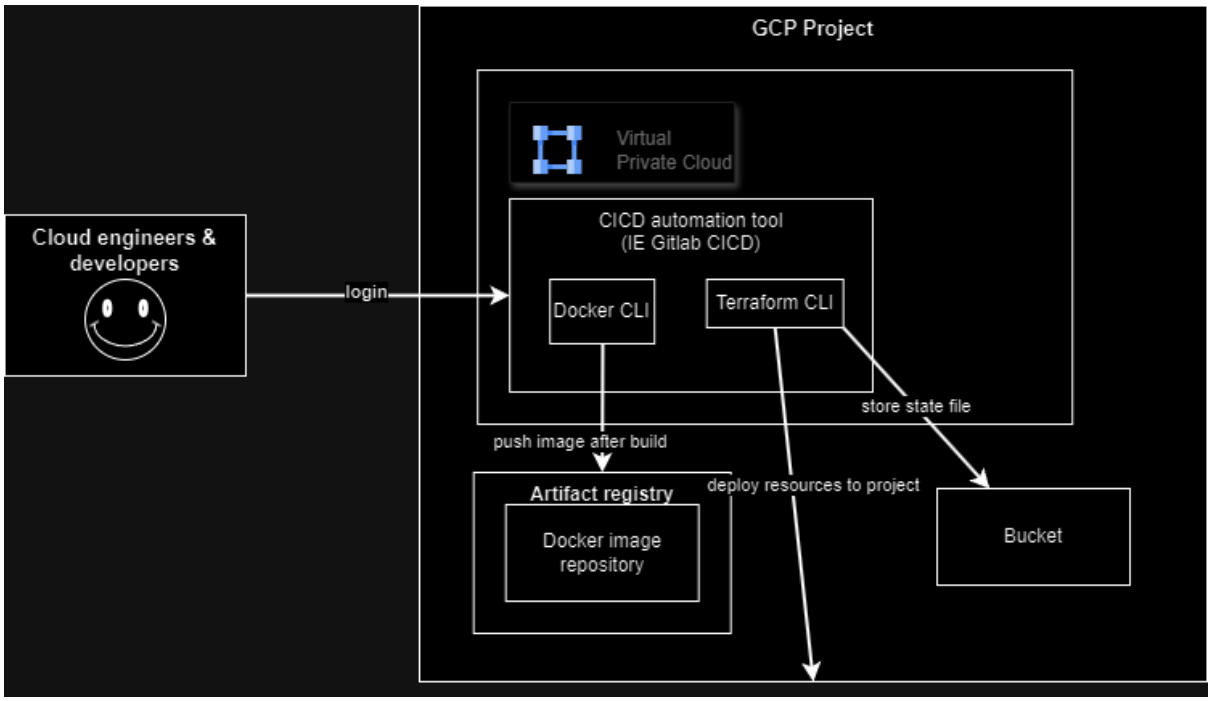
Docker images are produced by Project Echo and they are used to run containers. To run these as containers within GCP cloud repositories are required within the GCP project, which can be provided by Google Artifact Registry.

Future State Proposals

Future State Flow 1

Within the project Virtual Private Cloud (VPC) a CICD tool can be deployed (probably upon a Compute Engine Instance). Cloud engineers and developers can log into it to perform deployments to the project.

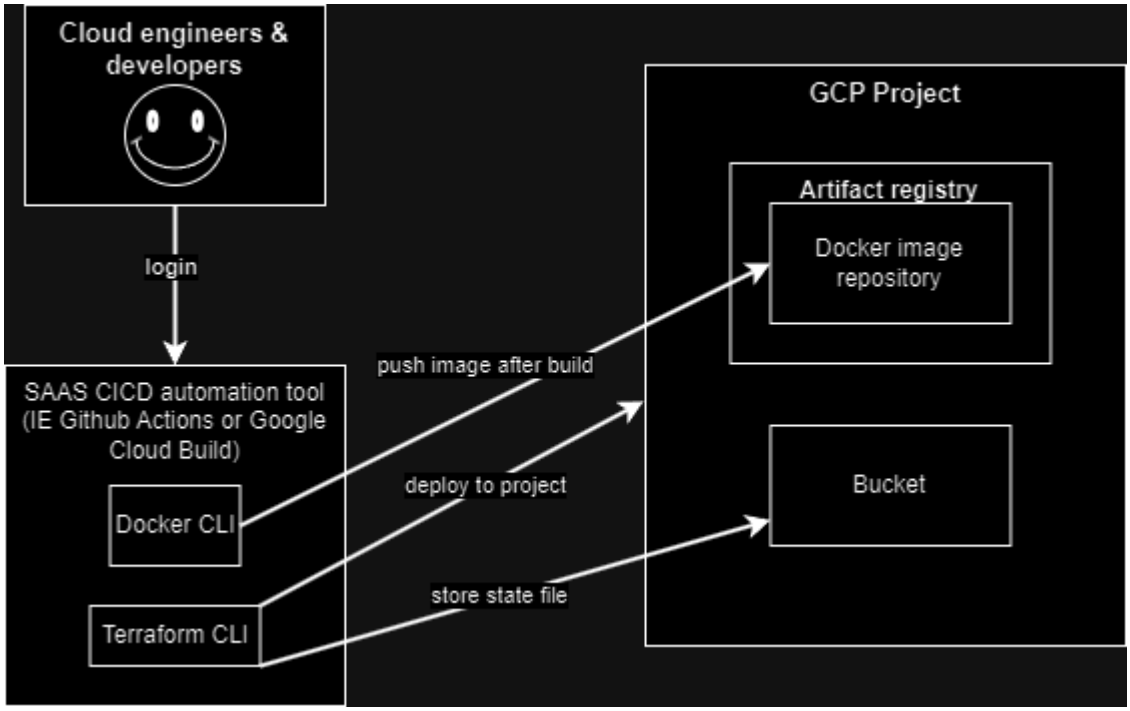
As the CICD tool is deployed within the project there would be one instance per environment, which is very good for ensuring there is no cross deployments between environments.



Future State Flow 2

This is like the above where cloud engineers and developers log into a centralized tool but the running the application and its related infrastructure is not required due to a Software as a Service (SaaS) CICD tool being used to deploy to the GCP project. It would be my preference to do this as it means managing less infrastructure.

A primary difference is that there is only one tool used to deploy to all environments so care must be taken to have pipelines configured correctly, as not to deploy to the wrong environment.



CICD Tool Suggestions

Gitlab CICD

This is the tool I would recommend using if deploying within the VPC. Pipelines are defined using yaml and it is a tool that follows more modern paradigms than older tools such as Jenkins.

Github Actions

Github Actions is the CICD component of Github and a CI pipeline is already in use for Project Echo's one repository. Pipelines are defined within yaml files and it is a recommended option if using a SAAS CICD tool.

Google Cloud Build

This is Google's CICD automation tool. It is modern and driven by pipelines defined as yaml files. I have no experience with it but reading the documentation and the fact that the project is using GCP this is a very viable option and worth exploring.

Jenkins

Jenkins is what would be considered the most classical CICD build tool. Pipelines are code driven (not config driven through yaml files) which adds to complexity creating a steeper learning curve. Despite using this daily for many years I recommend using more modern tooling.

Reference Links

https://developer.hashicorp.com/terraform/cli

https://docs.docker.com/reference/cli/docker/image/push/

https://cloud.google.com/storage/docs/creating-buckets

https://docs.gitlab.com/ee/install/

https://cloud.google.com/blog/topics/developers-practitioners/devops-and-cicd-google-cloud-explained