**GCP project creation**

To automate the project creation with a self service web portal, we should develop a web application(front-end) and deploy it into a GCP resource which should scale based on the usage and it should not block the users until the resources(GCP project with default resources and Github repo) are provisioned.

This application should be divided into two parts.

1. Frontend portal app.
2. Application which provisions the resources.

**Why are we isolating the provisioning logic into a different application? And why not in the same frontend portal app?**

By isolating the provisioning logic into a dedicated application we can avoid blocking the users in the frontend application till the resources are provisioned and it helps scaling the individual applications.

**I would recommend GKE and cloud pubsub for this self service portal application.**

**How does it work?**

Once users submit the form with required inputs for project creation, the inputs will be validated in the frontend portal app. If the validation is successful, the frontend portal app creates a servicenow ticket(for the records) and publishes a message to the pubsub topic with the required information(refer the fields specified).

Note: As part of the validation step, If the GCP project requested is for production environment, the frontend portal app will check for the security ISO approval whether the serviceNow ticket is approved by ISO or not.

Now, let's see the provisioning application which actually provisions the resources (both GCP project & Github repo).

Once the message is published by the frontend portal app in the cloud pubsub topic, the provisioning application which pulls the message through pull based subscription. And, it provisions the resources. The advantage is we can scale this workload based on the number of unacknowledged messages in the pubsub topic by creating HPA with an external metric.

**How resources can be provisioned by the provisioner application?**

Cloud Resource Manager API should be used in the provisioning application to provision the GCP resources. The GCP project should be created using the baseline template. The baseline template includes creation of subnets, Terraform service account, Log routers, IAM Groups and Labels.

**How can this solution be secured?**

Both authentication and authorization should be implemented in the frontend portal application.

Authentication using Okta SAML Login and Authorization using Sailpoint Access Manager.

Only frontend application service account should be assigned with Pub/Sub publisher IAM role in the pubsub topic (resource level binding). So, no one can publish the topic bypassing the frontend application. We should make sure no service account or individual should be given this IAM role in the project/organization/folder level.

And, provisioner application’s service account should be assigned with the Pub/Sub subscriber IAM role in the pubsub subscription (resource level binding).

A custom IAM role should be created with following permissions and assigned to the provisioner application’s service account (binded with workload identity KSA).

1. resourcemanager.projects.create

2. compute.subnetworks.create

3. iam.serviceAccounts.create

4. iam.groups.create

5. logging.configWriter

6. servicemanagement.services.bind

7. secretmanager.secrets.list

8. secretmanager.secrets.get

9. secretmanager.secrets.version

And, generate client id & client secret in Github & serviceNow and store it in the GCP Secret manager service by creating secrets using the obtained client id & client secrets.

Make sure that this application’s logs ship to the centralized logging.

Setup monitoring alerts for any admin activity which deletes or modifies crucial resources.

Finally, add VPC service controls to make it more secure.

**Will the front-end and CI/CD components also be stored within Version Control? Why?**

Obviously!!

Even the applications in the platform need bug fixes and improvements/releases over the time period. So, it is ideal to use a version control system for this application. And, it needs CICD pipeline too to automate the build & deployment process including scan in the sonarqube, build the code, push it to nexus, build & push the image to GCR and deploy it to the GKE cluster.

**Which CI/CD tool would you use? Why?**

I will choose Jenkins (hosted in kubernetes engine) because of the reasons listed below.

1. Opensource: Jenkins is an open source CICD tool. Hence, no license cost.
2. Ease of use: more documentation & community support.
3. Scalability: Jenkins in kubernetes provides good scalability.
4. Security: Setting up Jenkins in kubernetes provides workload identity which is nothing but it helps restricting users to make changes in their GCP project only.

I won’t say that other CICD tools like cloud build in the GCP are useless. But, Jenkins is user friendly, easier than other tools and most of the engineers are familiar with Jenkins.

**How will changes to the Github repository trigger the CI/CD pipeline?**

1. Generate the webhook url in the jenkins. This will generate a unique webhook url.
2. Copy the webhook url.
3. Go to the settings -> webhooks in your github repository.
4. Paste the webhook url in the Add webhook section.
5. Select which event should trigger the jenkins pipeline.
6. Click on the Add webhook button.

**where the web app will run and**

**how will it access Github and provision new GCP Projects for the users?**

The web application and provisioner application will run in the GKE cluster and the provisioner application will access Github using the Github client library.

1. Obtain the client id and client secret from the github app’s settings.
2. Create a secret using the client id and client secret in the secret manager service.
3. Assign the IAM role to the provisioner app SA to access secret manager service from application.
4. Use SecretManagerServiceClient in the application to retrieve the secret.
5. Using Github API, create the github repo and associate the github app in that newly created repo.

Do the same for ServiceNow Integration.

**Furthermore, you will need to architect the created GCP Project. Does it require any default GCP resources?**

Below are the fields the user should enter in order to request for gcp project creation.

1. Folder
2. Env
3. Security ISO approval (if env is prod)
4. Region (select from the list of available GCP regions)
5. Subnet needs to be created for GCP service (GKE/GKE LoadBalancer/GCE)
6. Service Project Name (in this format <Team>-<Project>)
7. Custom Terraform Service Account (which is already available in different project. if not provided, TF SA will be created by default)
8. IAM group for viewer / developer / admin (specify if custom group or leave it blank if you want portal to create IAM group for your project)
9. APIs to be enabled (from a list of whitelisted APIs in LBG)
10. Cost center
11. Data classification

Baseline Template

Below are the resources created as part of GCP project creation.

**Log routers/sinks** (which ships logs from your service project to centralized SIEM logging)

**Subnets** (created in host project. Later, compute network user role can be assigned in subnets for app SAs in service project)

**APIs to be enabled** (List of APIs that will be enabled in the service project from the list of whitelisted APIs in LBG)

**Custom Terraform Service Account** (Please specify custom TF SA If you plan to provision resources through an existing TF SA from a different GCP project. If not, new TF SA will be created in this new GCP project)

**IAM Groups** will be created for viewers, developers and admins. If you already have IAM groups created, You can specify that.

**Labels** (environment, cost center & data classification should be created as labels which is used to track costs with respect to business application)

**How will you secure the orchestration project and ensure users are limited to make changes to only their projects and Github repositories?**

Create a self service jenkins pipeline to provision a build pod template. In addition to that, this pipeline will create a new KSA for that build pod template and it will bind the user’s project’s terraform service account.

**What is build pod template?**

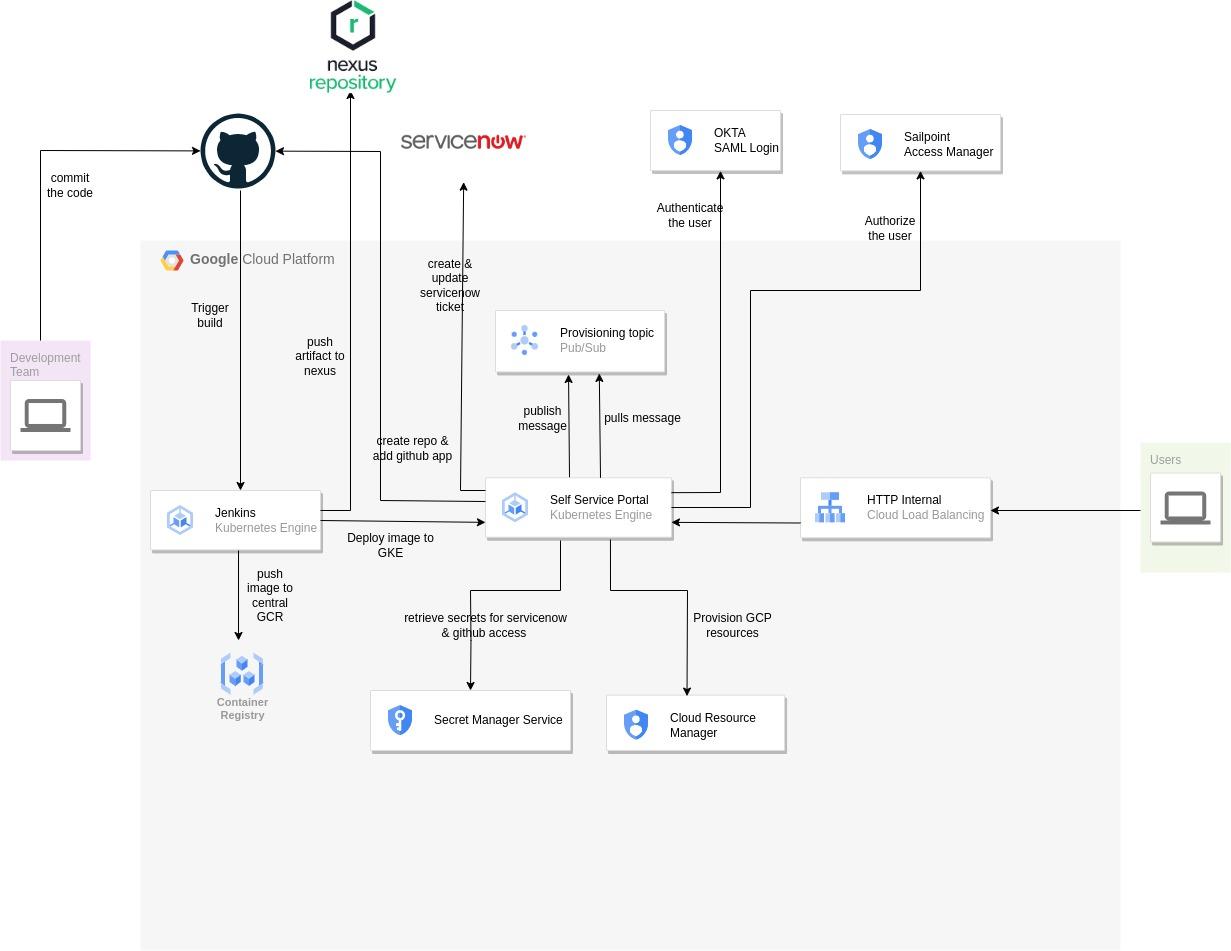
A well defined slave or build agent which will have access to make changes only in the particular user’s GCP project.

When a build is triggered by a user, a new independent pod(not under deployment / replicaset) will come up and execute the CICD steps on behalf of the user’s project’s terraform service account.

**Please scroll down for the designs !!!**

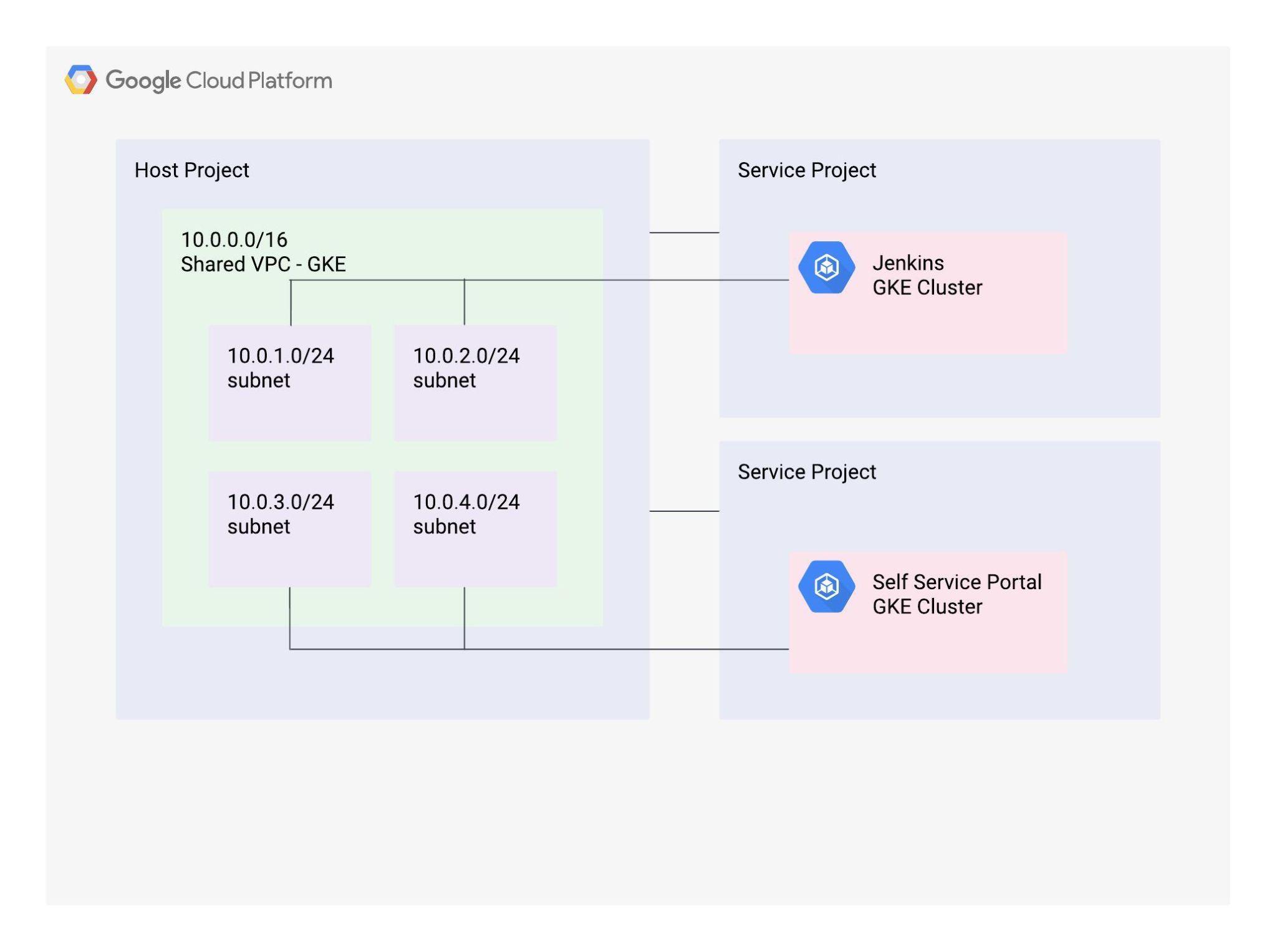
**Components Architecture**

Below is the components architecture which represents the big picture of this system.

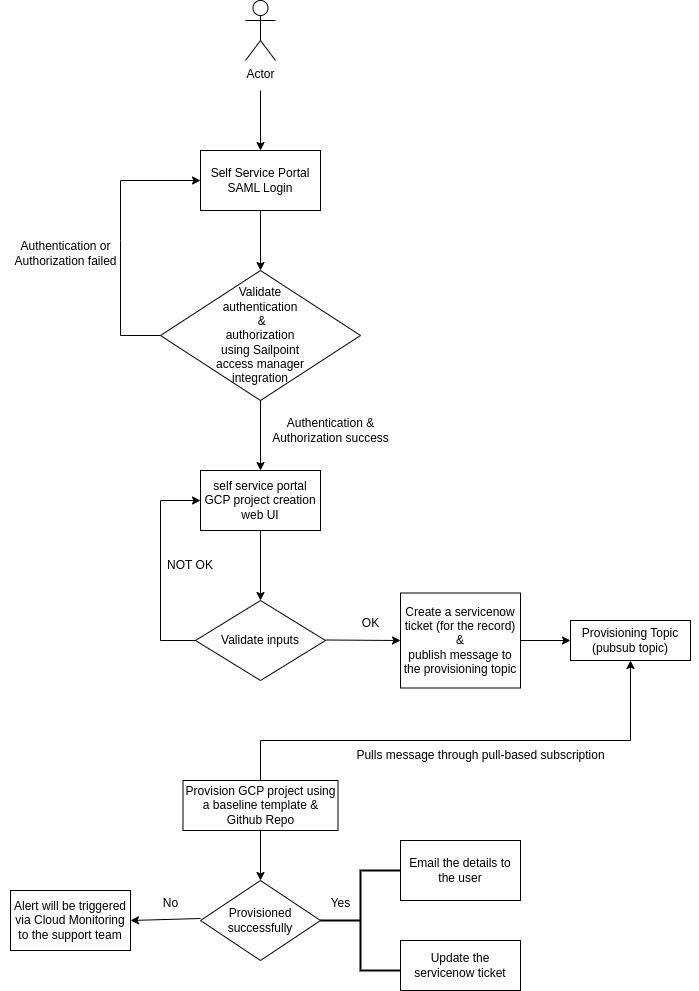


**Host Project & Service Project**

Below is the diagram which represents how resources are created in shared vpc.



**Application Flowchart**



**Kubernetes Architecture**

