Table of Contents

1. Document overview

This document includes detail steps on deploying GitHub selfhost runner on existing GKE standard cluster.

- The GKE cluster is provisioned using IaC as per the build sheet, includes all the infrastructure details configured for this setup.
- Details including
    - Cluster Deployment
    - Installation steps
    - Project Name
    - Cluster details
    - Network details
    - Deployment type (IaC/Cloud CLI/GUI)

2. Deployment of GitHub selfhost runner on GKE Standard cluster

Overview:

Runners are machines on which we can run our CICD pipelines on GitHub. GitHub supports 2 types of runners as mentioned below.

1. GitHub-Hosted Runners
    a. Machines configured by GitHub
    b. Offer a quicker, simpler way to run workflows
2. Self-Host runners
    a. Custom machines
    b. Highly configurable way to run workflows on our own custom environment.

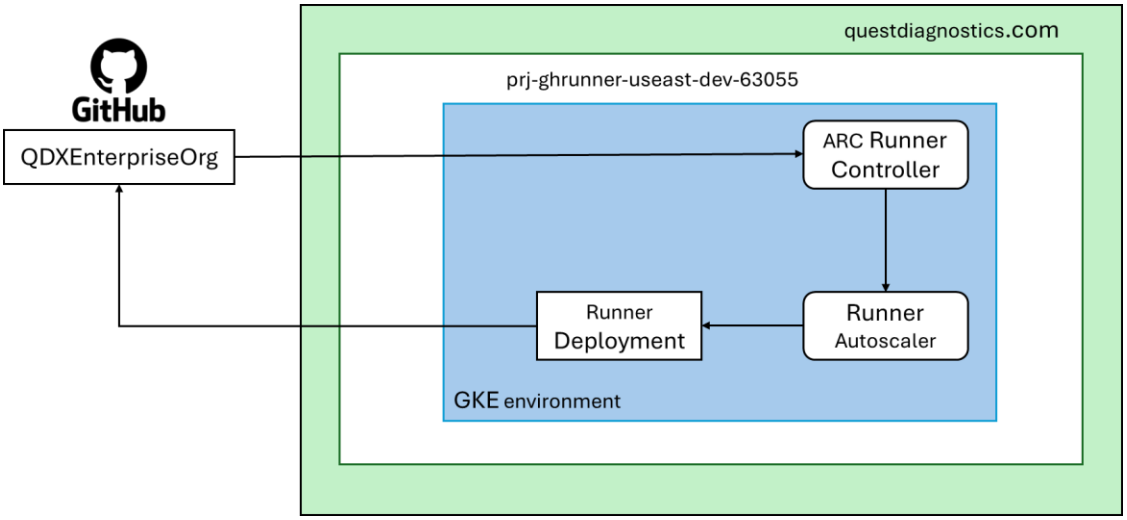
2.1: Pre-requisites for GitHub self-host runner:

- GKE Standard cluster
- Artifact registry
- Cloud Storage Bucket
- Jump-server (VM)
- GitHub app private key
- arc-controller-values.yml and arc-rs-values.yml files

GKE Standard cluster, Artifact Registry, Cloud storage bucket and jump-server has been **provisioned via terraform**.

GitHub repo for the above resources: [dso-gcpfoundation-iac-gha-runner/terraform-google-cloud-ghrunner at gha-config-files · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

Reference Architecture:

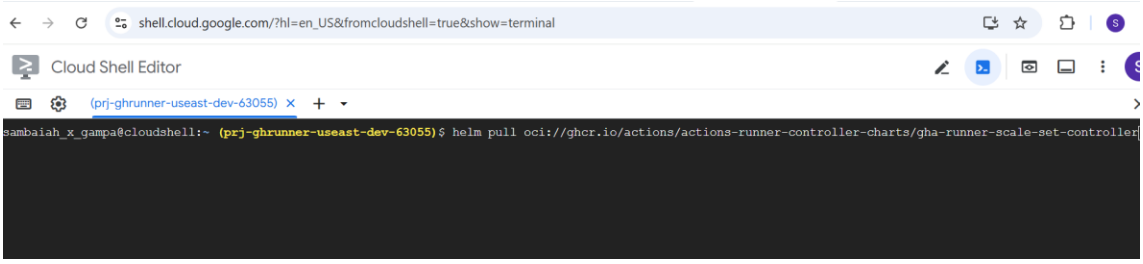


2.2: Download the below 2 helm charts for runner-controller and runner-scale-set from public repos for installation of controller and runner.

The below step1 and step2 need to be executed in cloud shell

Step1: gha-runner-scale-set-controller

helm pull oci://ghcr.io/actions/actions-runner-controller-charts/gha-runner-scale-set-controller



Step2: gha-runner-scale-set

helm pull oci://ghcr.io/actions/actions-runner-controller-charts/gha-runner-scale-set

2.3: Pull the below 3 images and push to GCP Artifact Registry

Artifact Registry:

Overview: Artifact Registry provides a single location for storing and managing your packages and Docker container images.
GitHub repo for Artifact Registry: [dso-gcpfoundation-iac-gha-runner/terraform-google-cloud-ghrunner/artifact-registry.tfvars at gha-config-files · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

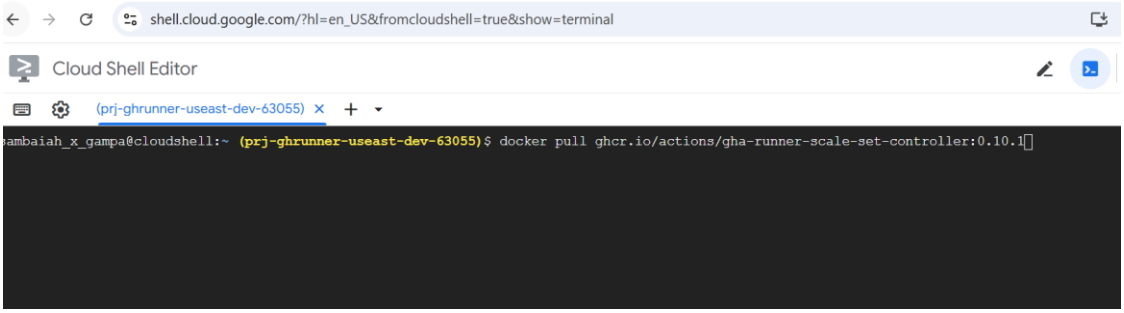
URL of GCP Artifact Registry used for this Project:

https://console.cloud.google.com/artifacts/docker/prj-ghrunner-useast-dev-63055/us-east4/af-ghrunner-useast-dev-77942?project=prj-ghrunner-useast-dev-63055

2.3.1: gha-runner-scale-set-controller

The step1, step2 and step3 need to be executed in cloud shell

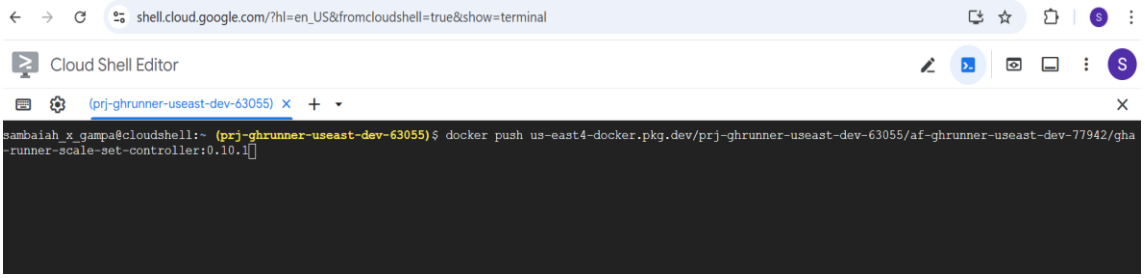
Step1: docker pull ghcr.io/actions/gha-runner-scale-set-controller:0.10.1



Step2: docker tag ghcr.io/actions/gha-runner-scale-set-controller:0.10.1 us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/gha-runner-scale-set-controller:0.10.1

Step3: Push the image to Artifact Registry

docker push us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/gha-runner-scale-set-controller:0.10.1

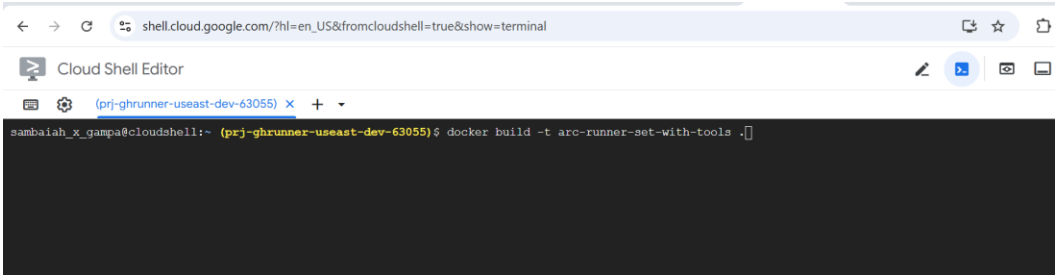


2.3.2: actions-runner

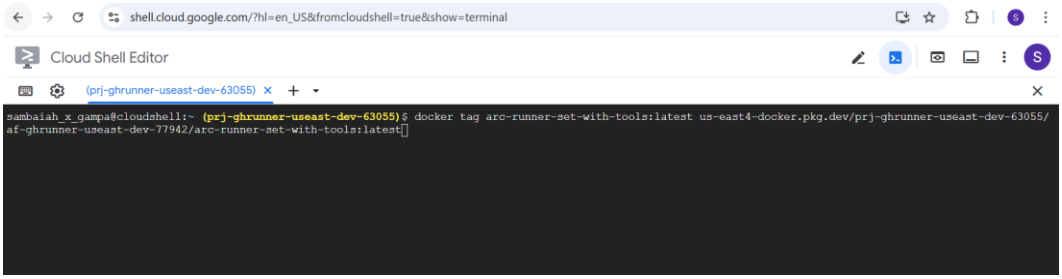
- The Dockerfile for actions-runner shared by Quest: https://github.com/QDXEnterpriseOrg/dso-gha-runner-docker-images/blob/main/ubuntu/Dockerfile
- Modified the Dockerfile accordingly (added TFlint and built the custom image and pushed the image to Artifact Registry)

The step1, step2 and step3 need to be executed in cloud shell

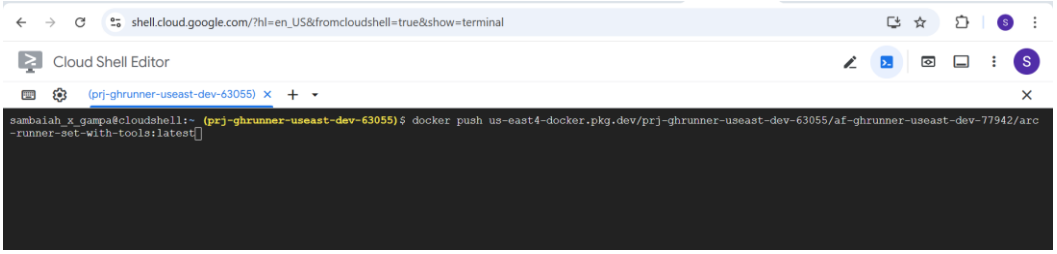
Step1: docker build -t arc-runner-set-with-tools .



Step2: docker tag arc-runner-set-with-tools:latest us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/arc-runner-set-with-tools:latest



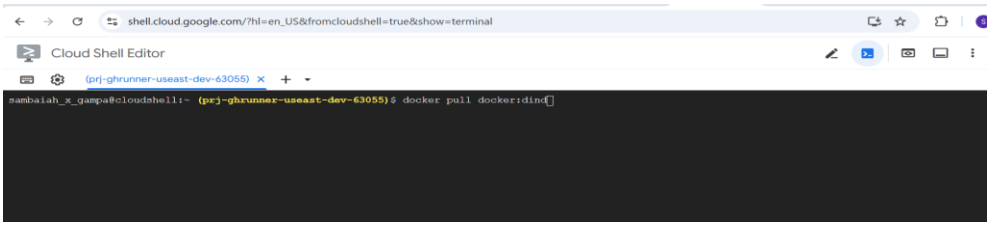
Step3: docker push us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/arc-runner-set-with-tools:latest



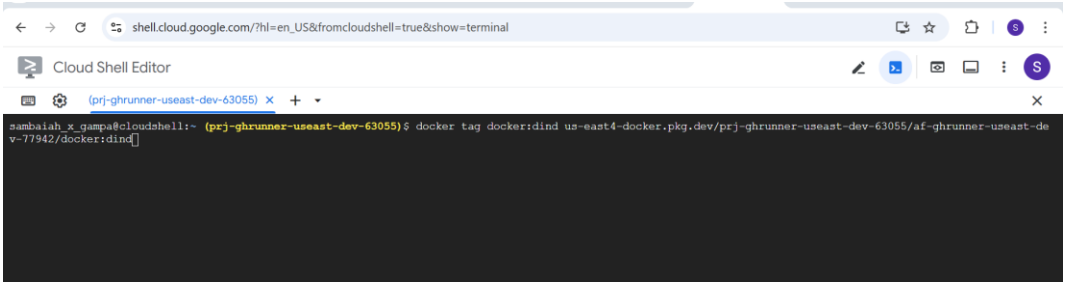
2.2.3: docker:dind

The step1, step2 and step3 need to be executed in cloud shell

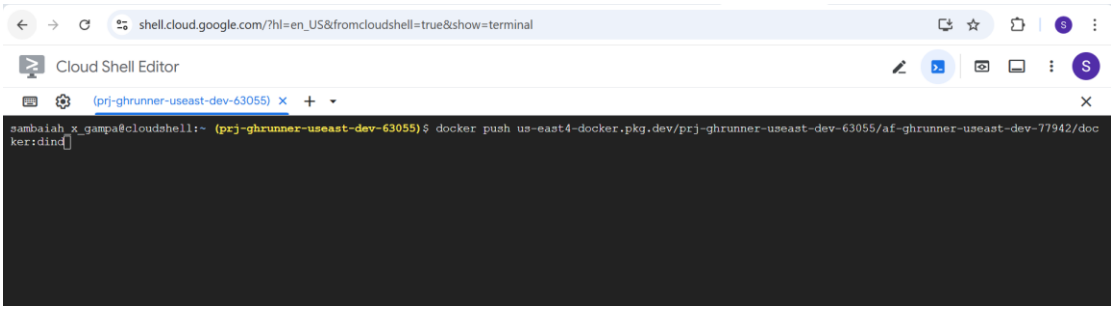
Step1: docker pull docker:dind



Step2: docker tag docker:dind us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/docker:dind



Step3: docker push us-east4-docker.pkg.dev/prj-ghrunner-useast-dev-63055/af-ghrunner-useast-dev-77942/docker:dind



2.4: Update values.yaml configuration file parameters

Overview: Configuration files required for installation of runner-controller and runner-scale-set. Modified the parameters accordingly based on the image path in Artifact Registry and uploaded the files in below GitHub repo.

GitHub repo: [dso-gcpfoundation-iac-gha-runner/gha-runner-configs at gha-config-files · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

2.5: Deployment steps

Jump-server: Used to connect to the private cluster

Github repo for jump-server: [dso-gcpfoundation-iac-gha-runner/terraform-google-cloud-ghrunner/vm.tfvars at gha-config-files · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

Install kubectl and google cloud cli gke auth plugin on the server using below commands

Below commands needs to be run on jump-server

Step1: sudo apt-get install kubectl

Step2: sudo apt-get install google-cloud-cli-gke-gcloud-auth-plugin

2.5.1: Connect to cluster via jump-server and check the connectivity.

Download helm on the jump-server using below commands:

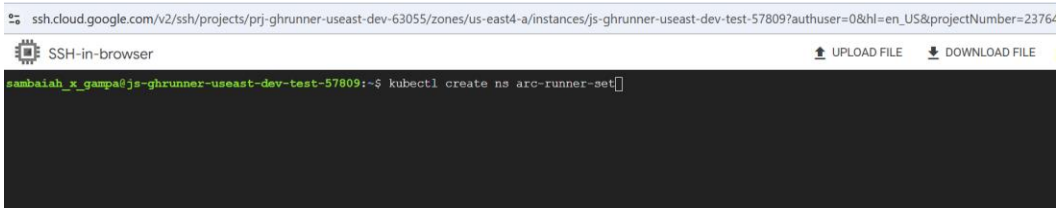
Below commands needs to be run on jump-server

- curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
- chmod 700 get_helm.sh
- ./get_helm.sh

Create below 3 namespaces:

1. arc-runner-set
2. arc-controller
3. cert-manager

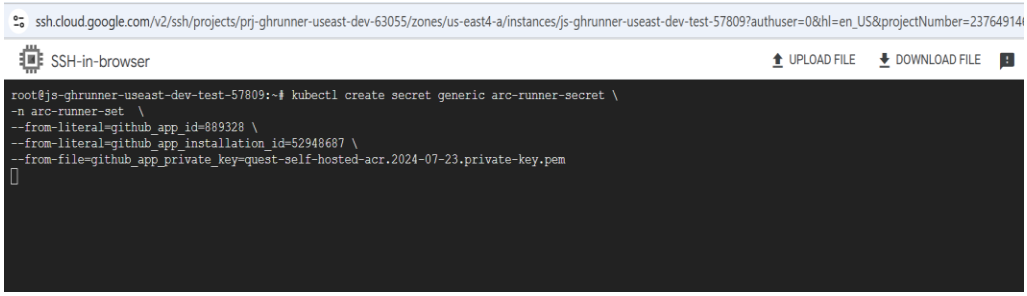
Example: kubectl create ns **namespace-name**



Create a generic secret in arc-runner-set namespace which is required to register GitHub runner

Below command needs to be run on jump-server

```
kubectl create secret generic arc-runner-secret \
-n arc-runner-set  \
--from-literal=github_app_id=889328 \
--from-literal=github_app_installation_id=52948687 \
--from-file=github_app_private_key=quest-self-hosted-acr.2024-07-23.private-key.pem
```

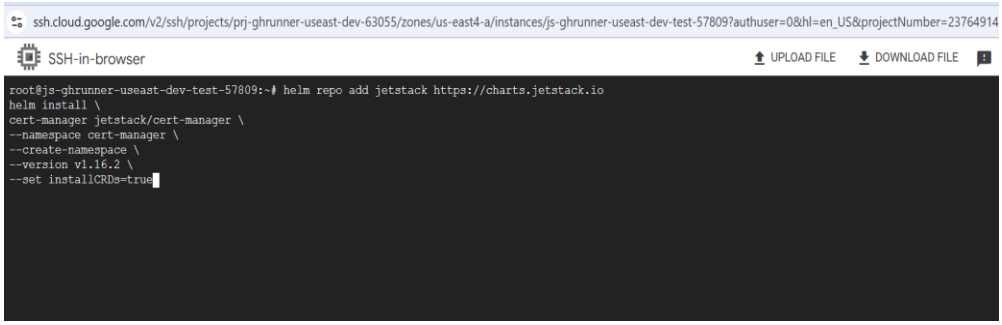


Install cert-manager in cert-manager namespace:

Overview: By default, actions-runner-controller uses cert-manager for certificate management of admission webhook, so we have to make sure cert-manager is installed on Kubernetes before we install actions-runner-controller

Below command needs to be run on jump-server

```
helm repo add jetstack https://charts.jetstack.io

helm install \

cert-manager jetstack/cert-manager \

--namespace cert-manager \

--create-namespace \

--version v1.16.2 \

--set installCRDs=true
```



- Copy the helm packages downloaded in step 1 to present working directory.
- Copy the arc-controller-values.yml and arc-rs-values.yml config files to present working directory by updating the image path accordingly.

2.5.2: Steps to install runner

Step1: Install arc-controller:

Overview: ARC is a Kubernetes controller that creates self-hosted runners on your Kubernetes cluster. It comprises several custom resource definitions such as (Runner, Runner Set, Runner Deployment, Runner Replica Set, and Horizontal Runner AutoScaler).

The following command installs arc-controller into arc-controller namespace

Below command needs to be run on jump-server

```
helm install arc-controller \
--namespace arc-controller \
-f arc-controller-values.yml  \
./gha-runner-scale-set-controller-0.10.1.tgz --debug
```

Step2: Install runner-scale-set:

The following command installs runner-scale-set into arc- runner-set namespace

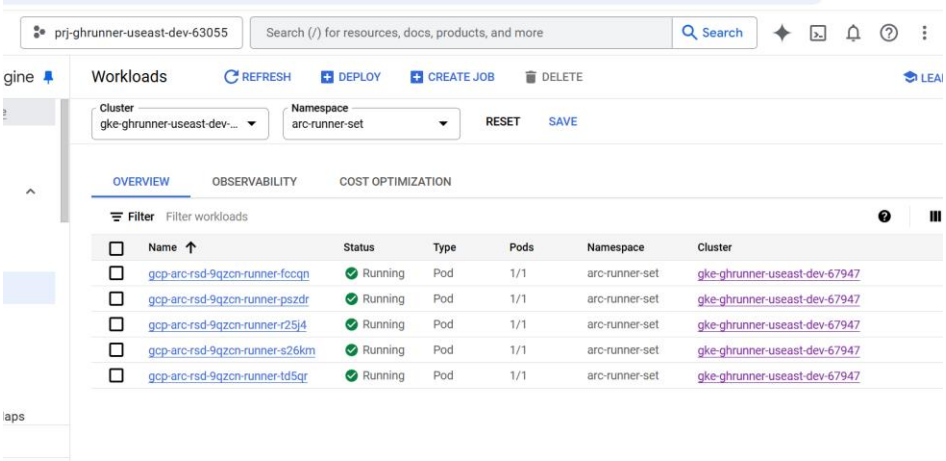
Below command needs to be run on jump-server

```
helm install gcp-arc-rs \
--namespace arc-runner-set \
-f arc-rs-values.yml \
./gha-runner-scale-set-0.10.1.tgz --debug
```

Note: The configuration files and Dockerfile used in previous steps are in below github repo [dso-gcpfoundation-iac-gha-runner/gha-runner-configs at gha-config-files · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

2.5.3: Validate all the workloads deployed on GCP on the respective namespaces:

Path: https://console.cloud.google.com/kubernetes/workload/overview?project=prj-ghrunner-useast-dev-63055



2.6: Validation of Runners on GitHub

Goto GitHub actions->self-host runners and validate the deployed runners.

Path: **Runners · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner**

Troubleshooting:

1. Jump-server not able to connect to internet to authenticate GCP account. IP ranges of jump-server has been added in Firewall (fw-allow-ingress-fortigate-hub-nonprod, fw-allow-egress-fortigate-hub-nonprod) to allow. [**Project_id:** prj-shrd-ntwk-3]
2. Added network internal lb ip 10.141.128.5 for sending traffic to the forti-proxy instances (route: route-ue4-nprd-to-rinlb) [**Project_id:** prj-shrd-ntwk-3]

3. Workload Identity Federation:

3.1: Overview on Workload Identity Federation

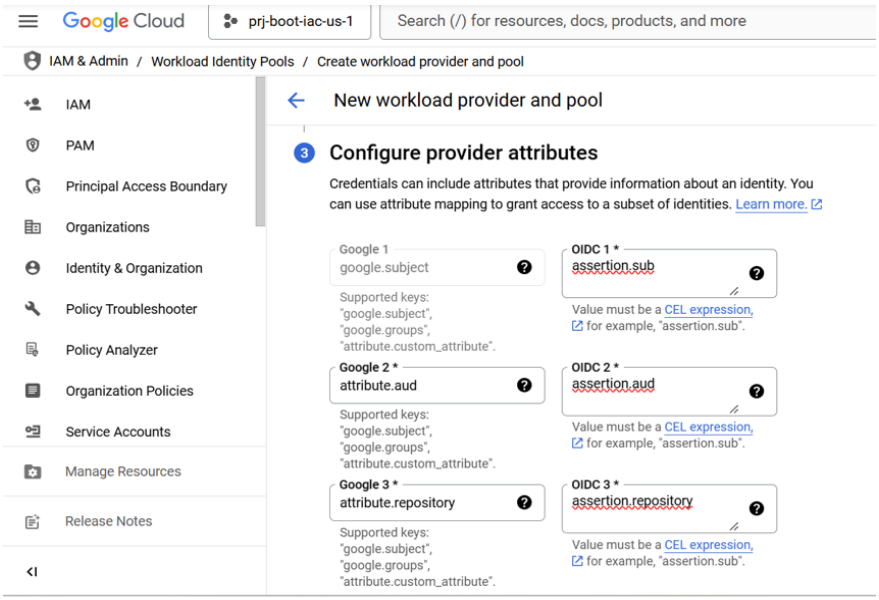
Overview: To Authenticate GitHub to GCP using WIF. Traditionally, applications running outside Google Cloud can use service account keys to access Google Cloud resources. However, service account keys are powerful credentials, and can present a security risk if they are not managed correctly. Workload Identity Federation eliminates the maintenance and security burden associated with service account keys.
Project_id: prj-boot-iac-us-1

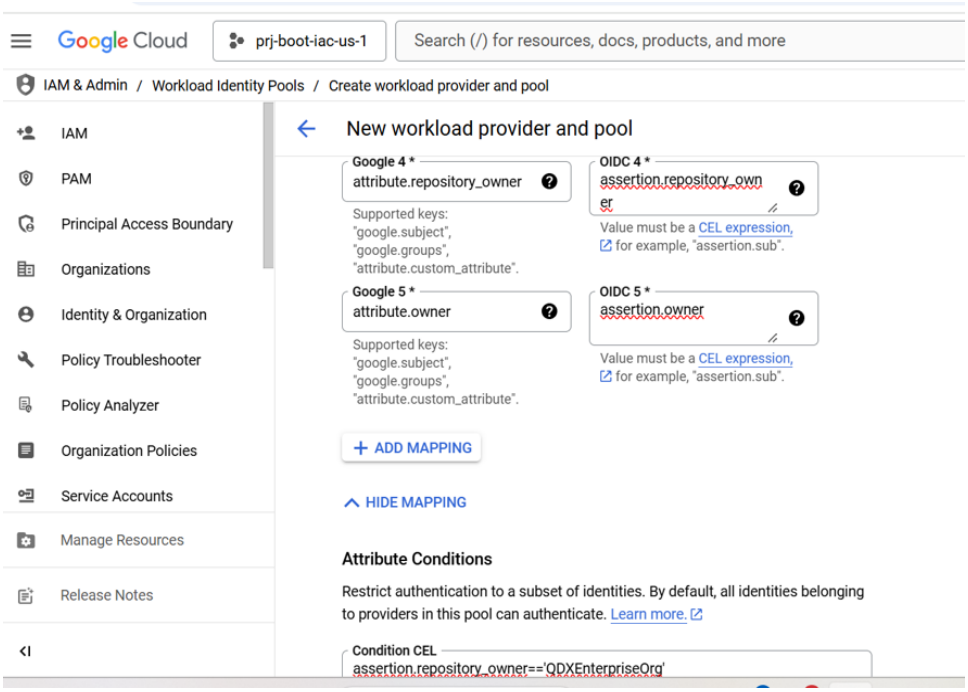
3.2: Implementation

WIF Configured via GUI console (One-time activity)

- Create a workload identity pool and provider.
- Provider: OpenID Connect (OIDC)
- Issuer: https://token.actions.githubusercontent.com

Provider Attributes and Attribute Condition:





Command for integration at repo level:

Run the below command on cloud shell

```
gcloud iam service-accounts add-iam-policy-binding "id-sa-boot-iac-us-4000@prj-boot-iac-us-1.iam.gserviceaccount.com" \
--project="prj-boot-iac-us-1" \
--role="roles/iam.workloadIdentityUser" \
--member="principalSet://iam.googleapis.com/projects/974223802243/locations/global/workloadIdentityPools/github/attribute.repository/QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner"
```

```
sambaiah_x_gampa@cloudshell:~ (prj-ghrunner-useast-dev-63055)$ gcloud iam service-accounts add-iam-policy-binding "id-sa-boot-iac-us-40000@prj-boot-iac-us-1.iam.g
serviceaccount.com" \
--project="prj-boot-iac-us-1" \
--role="roles/iam.workloadIdentityUser" \
--member="principalSet://iam.googleapis.com/projects/974223802243/locations/global/workloadIdentityPools/github/attribute.repository/QDXEnterpriseOrg/dso-gcpfoun
dation-iac-gha-runner"
```

Note: Need to run this command for all the required repos or use below workflow by providing repo.

WIF workflow for integration at repo level:

Non-prod:

[WIF integration of GCP repos · Workflow runs · QDXEnterpriseOrg/dso-gcpfoundation-iac-gha-runner]

Prod:

[WIF integration of GCP prod repos · Workflow runs · QDXEnterpriseOrg/dso-gcpfoundation-iac-prod-gha-runner]

4. Metadata:

- GCP Project : prj-ghrunner-useast-dev-63055
- Cluster name : gke-ghrunner-useast-dev-67947
- VPC : vpc-non-prod-shared-host
- Primary Subnet : sn-ue4-gke-ghrunner-dev-1
- Secondary subnet Pod : sipr-ue4-ghrunnerpod-dev-1 (100.64.0.0/20)
- Secondary subnet Services : sipr-ue4-ghrunnerserv-dev-2 (100.62.0.0/24)
- Jump-Server (VM) : js-ghrunner-useast-dev-test-57809
- GCS Bucket : gh-selfhosted-runner
- Artifact Registry : af-ghrunner-useast-dev-77942
- Service Account : sa-gkeghrunner-dev

  IAM role for service account : roles/artifactregistry.admin, roles/compute.networkUser, roles/container.admin