Published in FAUN Publication

Johanes Glenn Follow
Jan 29, 2022 · 6 min read · Listen

Setting Up Anthos in GKE for Demo using Terraform

Background

Lately I have done several deployment of Anthos environment for demo/workshop/self-testing. I would recommend that on exploration/self-testing to run manually so that we will be able to understand more about the component, however to setup multiple time the similar configuration take a lot of effort for repetitie activity. That is the reason I'd love to refer to the documentation of **"Provisioning Anthos clusters with Terraform"**

Provisioning Anthos clusters with Terraform | Cloud Architecture Center | Google Cloud

This tutorial walks you through provisioning Anthos clusters and Anthos components using the Google Cloud modules for...

cloud.google.com

By utilizing terraform we can consistently deploy the environment (or destroy completely later on), and this may help a lot for anyone who want to:

[A] Test quickly Anthos on GCP and explore Anthos features or

[B] Try the terraform gcp modules setting up Anthos Service Mesh.

> ***Notes:*** *This story is not intended for production guide, more in conceptual test and ease in testing Anthos features.*

Concept

To set the idea, with the current configuration we will have:

2x GKE Cluster (I use two different region in this case)

ASM on each cluster

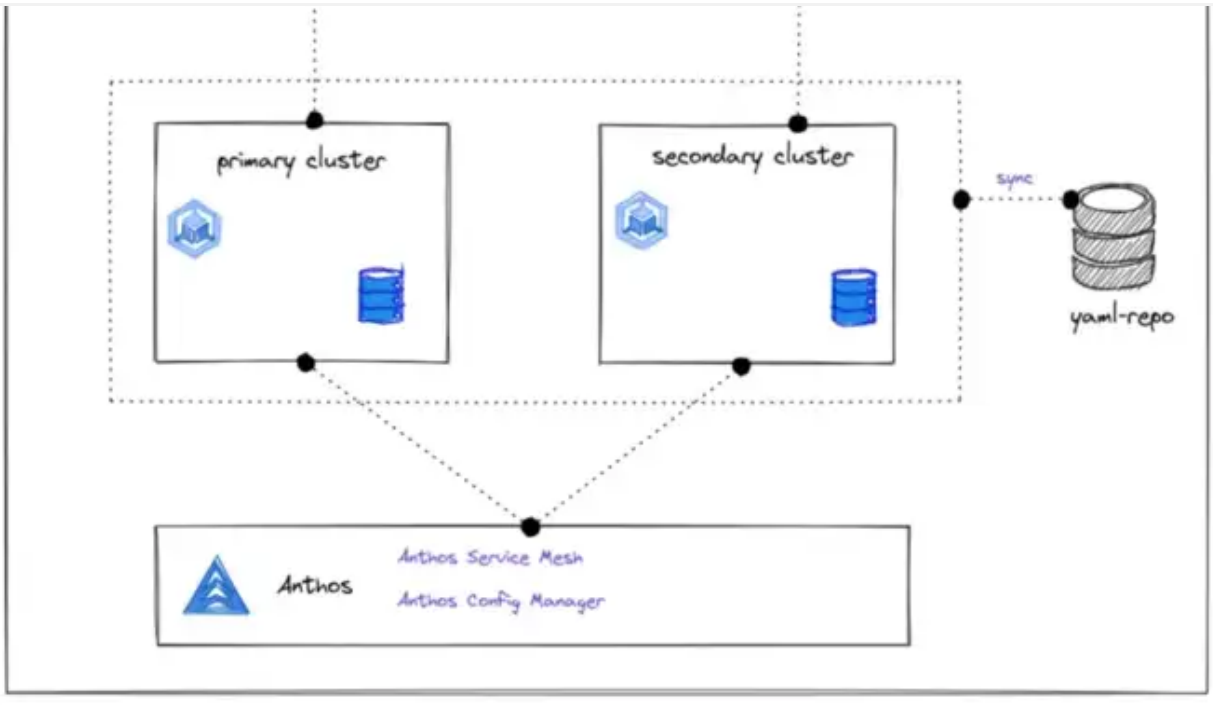
ACM registration later on (manual as I want to set through ui manually)

and will use Bank of Anthos as the example workloads

GitHub - GoogleCloudPlatform/bank-of-anthos: Anthos sample application for retail banking

⚠ ATTENTION: Apache Log4j 2 advisory. Due to vulnerabilities present in earlier versions of Log4j 2, we have taken down...

github.com



The highlevel of the environment setup

Terraform

We can directly use the tf files from the public documentation above, or modify it based on our needs. I think some of the variables need to be modify atleast (eg. project_id, region, zone and vpc + subnet). One of the biggest reasons is that the tf files on the sample will use the default vpc with autosubnet, unless we utilize the same vpc we require to change it.

```
module "project-services" {
  source  = "terraform-google-modules/project-factory/google//modules/project_services"

  project_id  = data.google_client_config.current.project
  disable_services_on_destroy = false
  activate_apis = [
    "compute.googleapis.com",
    "iam.googleapis.com",
    "container.googleapis.com",
    "cloudresourcemanager.googleapis.com",
    "anthos.googleapis.com",
    "cloudtrace.googleapis.com",
    "meshca.googleapis.com",
    "meshtelemetry.googleapis.com",
    "meshconfig.googleapis.com",
    "iamcredentials.googleapis.com",
    "gkeconnect.googleapis.com",
    "gkehub.googleapis.com",
    "monitoring.googleapis.com",
    "logging.googleapis.com"

  ]
}
```

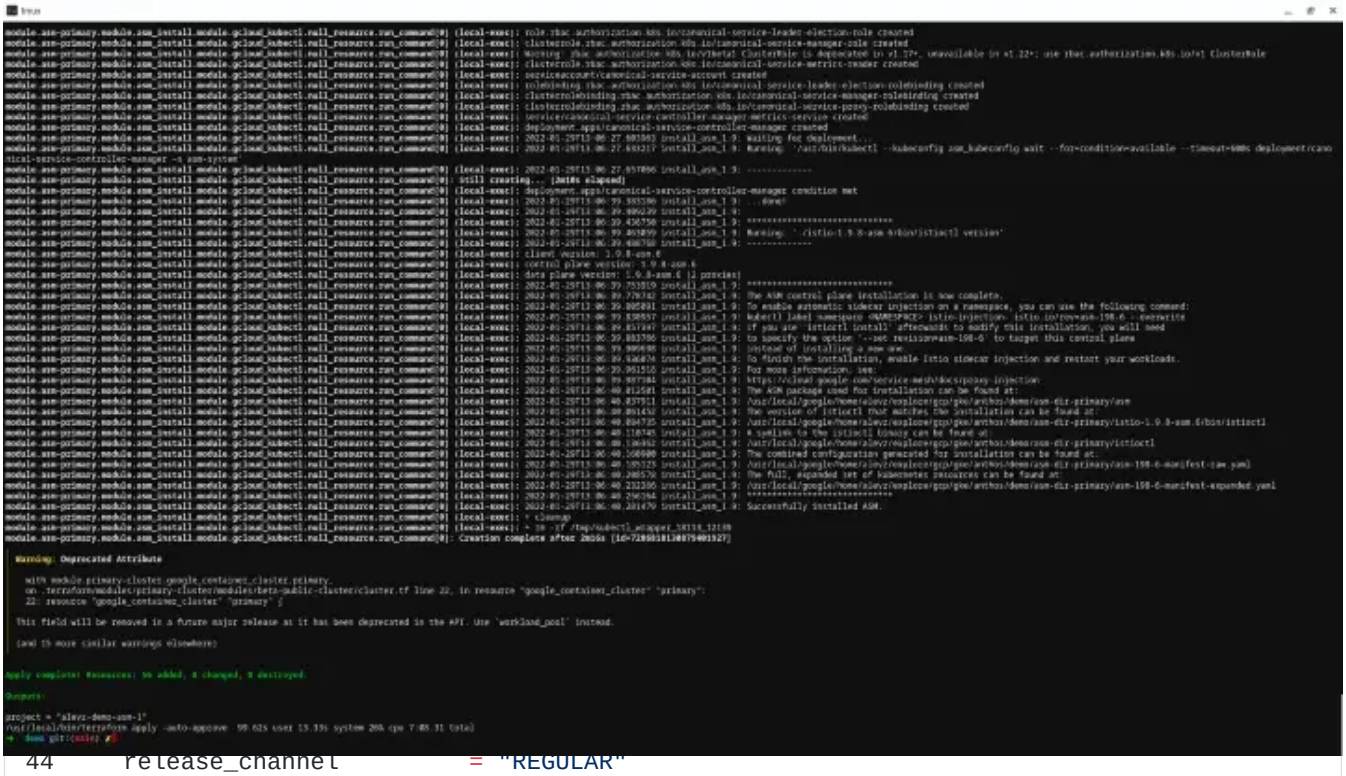
apis.tf hosted with ❤ by **GitHub** **view raw**

```
module "asm-primary" {
  source                = "terraform-google-modules/kubernetes-engine/google//modules/a
  version               = "18.0.0"
  project_id            = data.google_client_config.current.project
  cluster_name          = module.primary-cluster.name
  location              = module.primary-cluster.location
  cluster_endpoint      = module.primary-cluster.endpoint
  enable_all            = true
  outdir                = "./asm-dir-${module.primary-cluster.name}"

}

module "asm-secondary" {
  source                = "terraform-google-modules/kubernetes-engine/google//modules/a
  version               = "18.0.0"
  project_id            = data.google_client_config.current.project
  cluster_name          = module.secondary-cluster.name
  location              = module.secondary-cluster.location
  cluster_endpoint      = module.secondary-cluster.endpoint
  enable_all            = true
  outdir                = "./asm-dir-${module.secondary-cluster.name}"

```

```
23  }
```

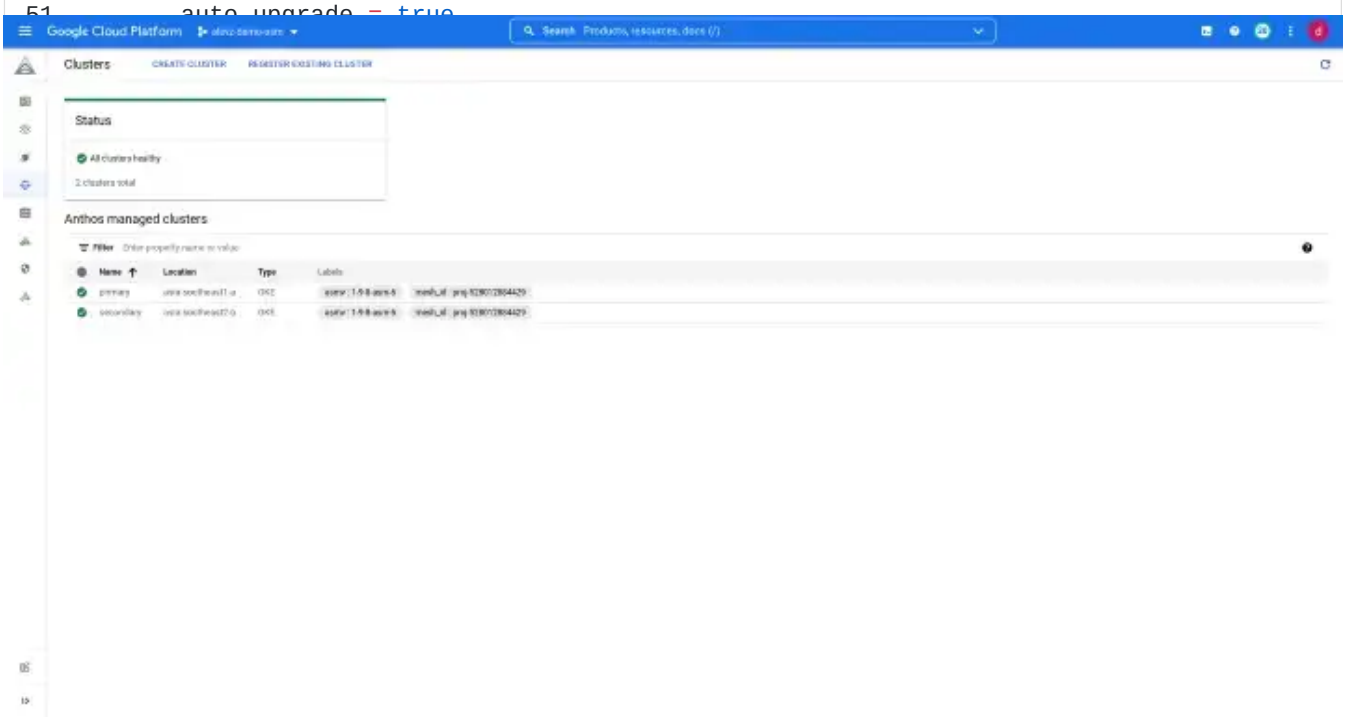
asm.tf hosted with ❤ by GitHub view raw

```
1   # Primary Cluster
2   module "primary-cluster" {
3     name                      = "primary"
4     project_id                = module.project-services.project_id
5     source                    = "terraform-google-modules/kubernetes-engine/google//modules
6     version                   = "18.0.0"
7     regional                  = false
8     region                    = var.primary_region
9     network                   = var.vpc
10    subnetwork                = var.subnet
11    ip_range_pods             = ""
12    ip_range_services         = ""
13    identity_namespace        = "enabled"
14    zones                     = var.primary_zones
15    release_channel           = "REGULAR"
16    cluster_resource_labels = { "mesh_id" : "proj-${data.google_project.project.number}"
17    node_pools = [
18      {
19        name         = "default-node-pool"
20        autoscaling  = false
21        auto_upgrade = true
22
23        node_count   = 3
24        machine_type = "e2-standard-4"
25      },
26    ]
```

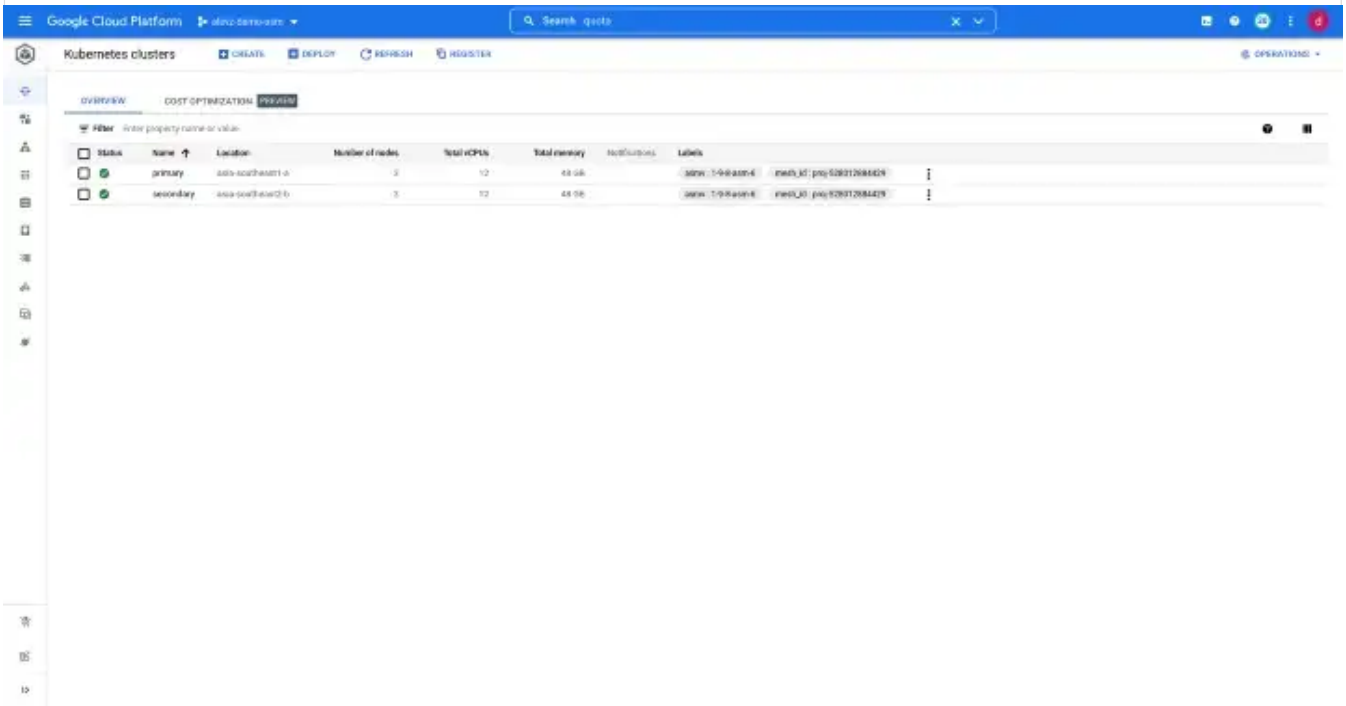


```
44    release_channel           = "REGULAR"
```

```
    cluster_resource_labels = { "mesh_id" : "proj-${data.google_project.project.number}"

    node_pools = [
      {
        name         = "default-node-pool"
        autoscaling  = false
        auto_upgrade = true
```

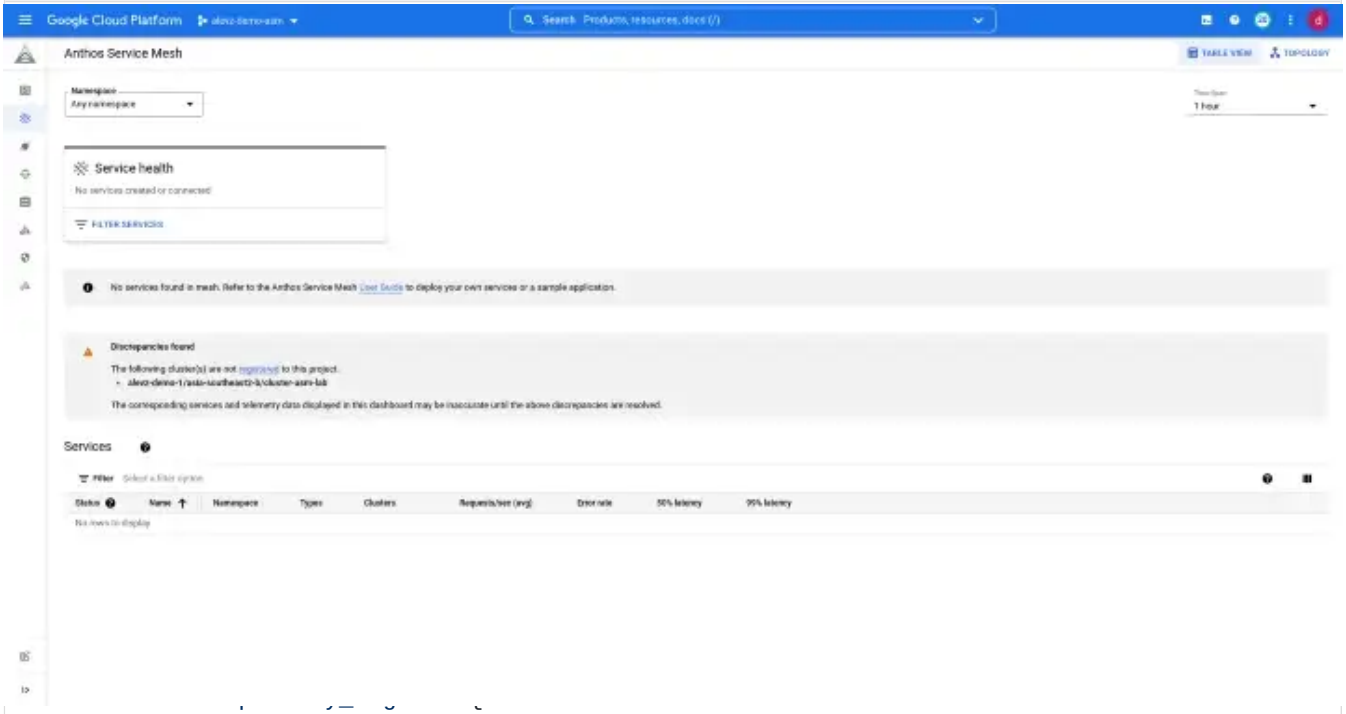



```
    cluster_endpoint = module.primary-cluster.endpoint
    gke_hub_membership_name = "primary"
    gke_hub_sa_name = "primary"
```

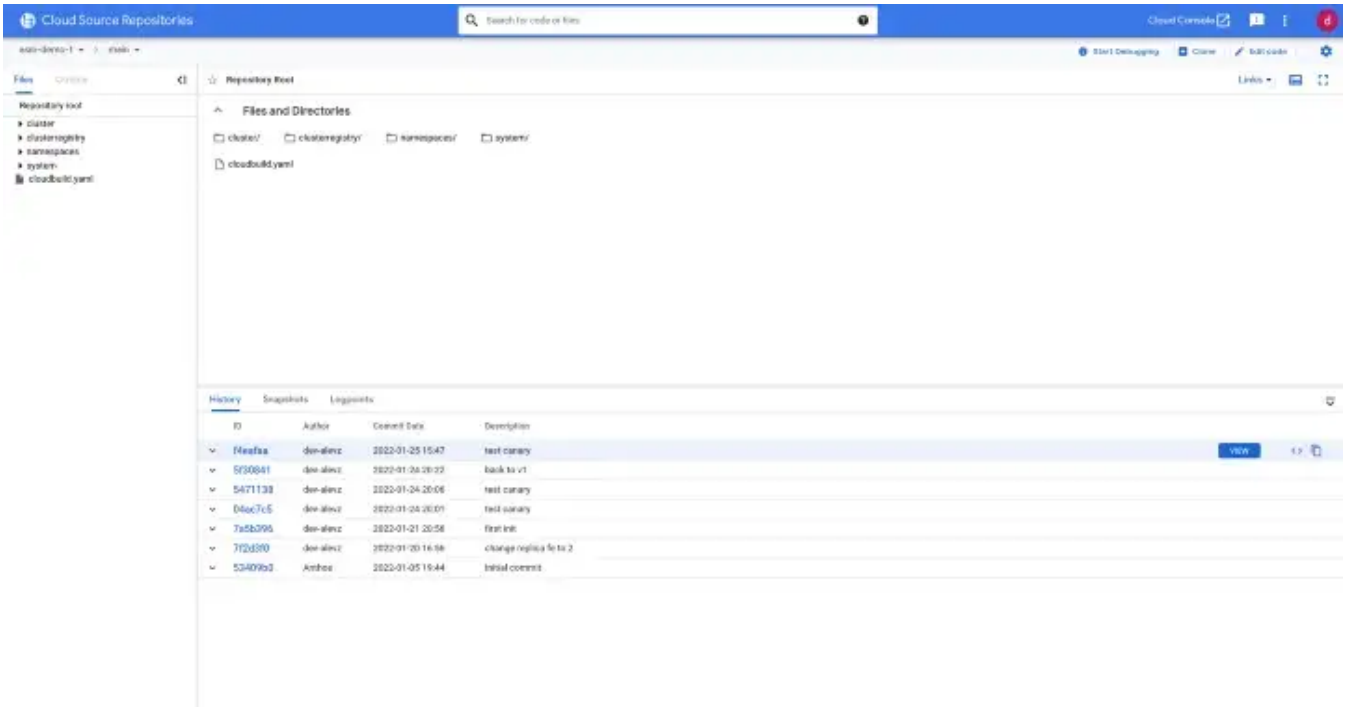



```
    region   = var.primary_region
  }

  data "google_client_config" "current" {}

  data "google_project" "project" {
    project_id = var.project_id
```

```
}

output "project" {
  value = data.google_client_config.current.project
}
```




```
  description = "The primary region to be used"
}
variable "primary_zones" {
  description = "The primary zones to be used"
}

variable "secondary_region" {
  description = "The secondary region to be used"
}
variable "secondary_zones" {
  description = "The secondary zones to be used"
}

variable "vpc" {
  description = "The default VPC value"
}

variable "subnet" {
  description = "The default subnet value"
```

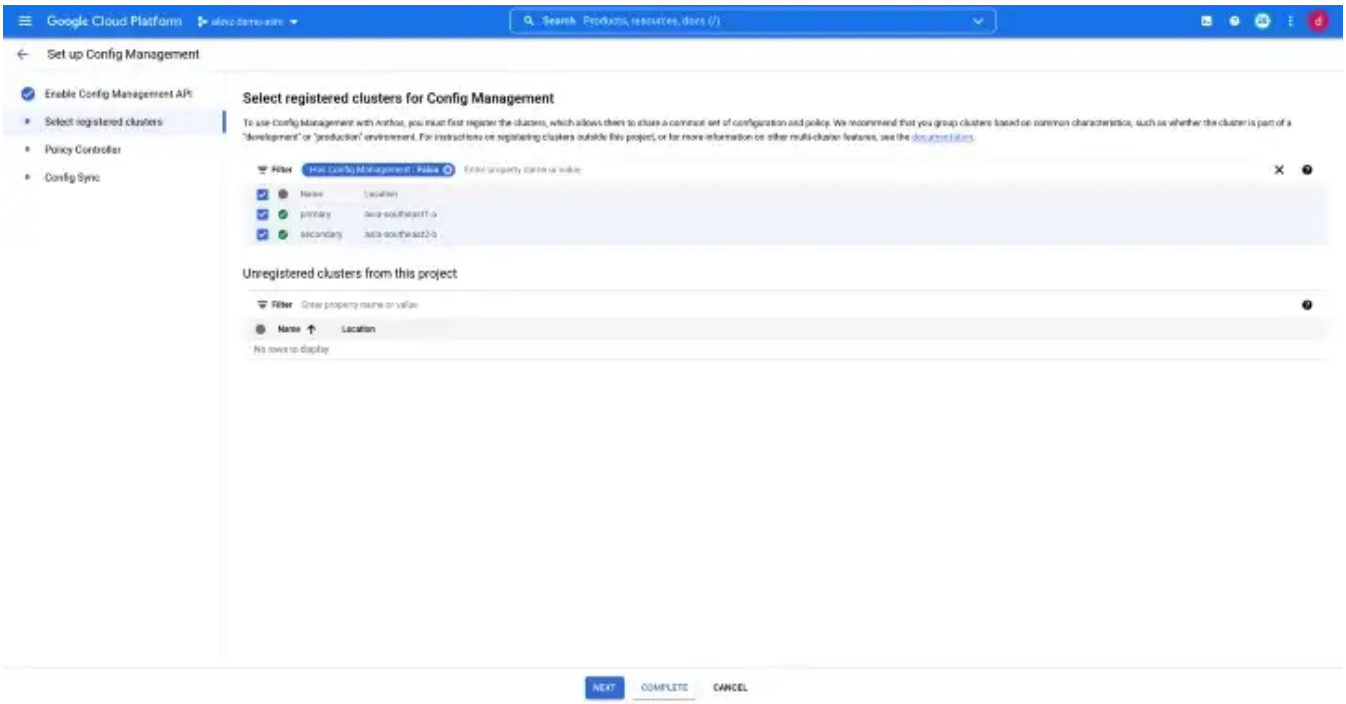


I push the manifests I need on the code source repo

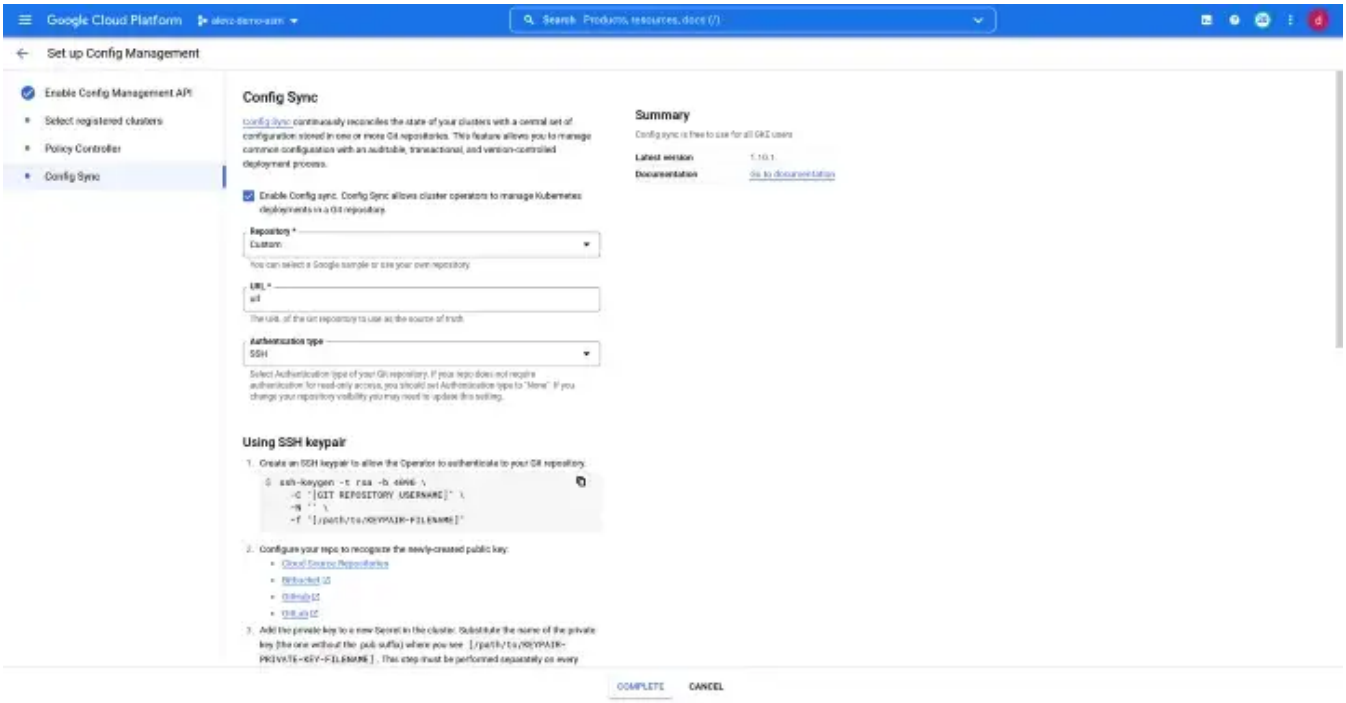
To run the setup:

[**One**] Go to Anthos UI and select Config Management

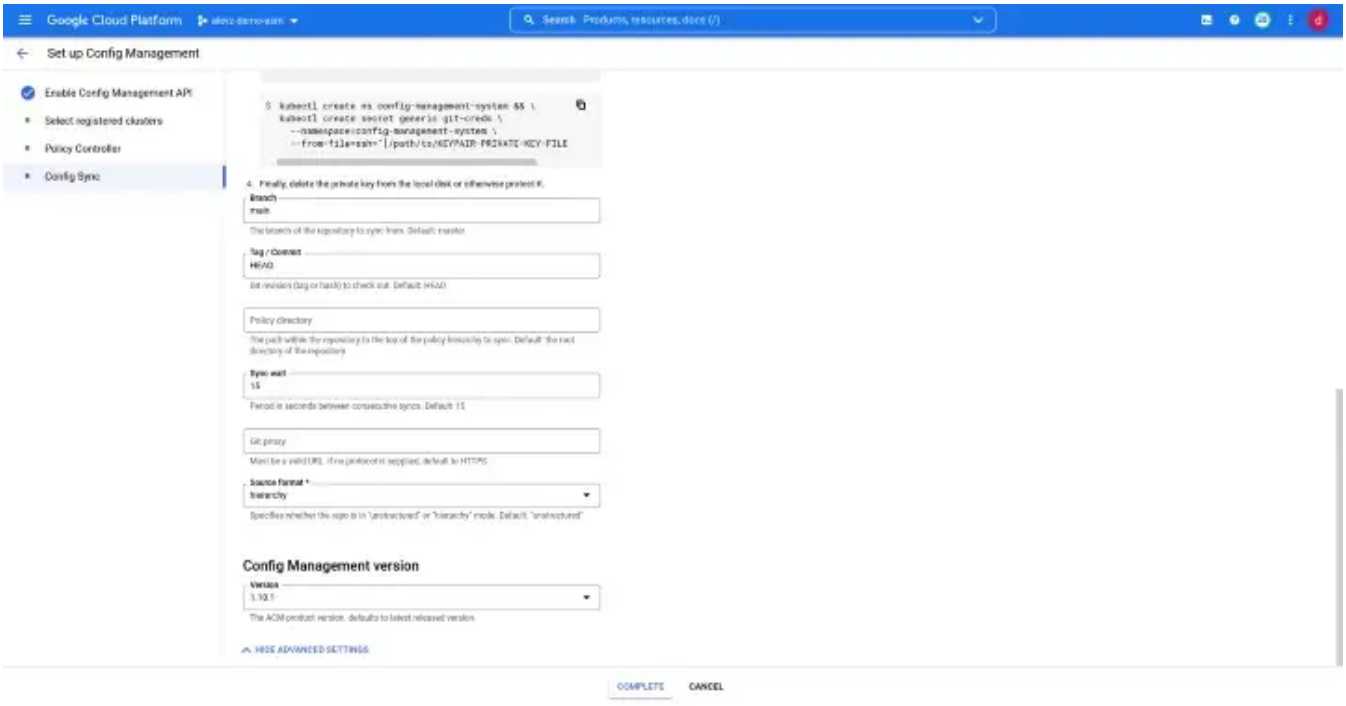
[**Two**] Select the cluster we want to register to the configuration



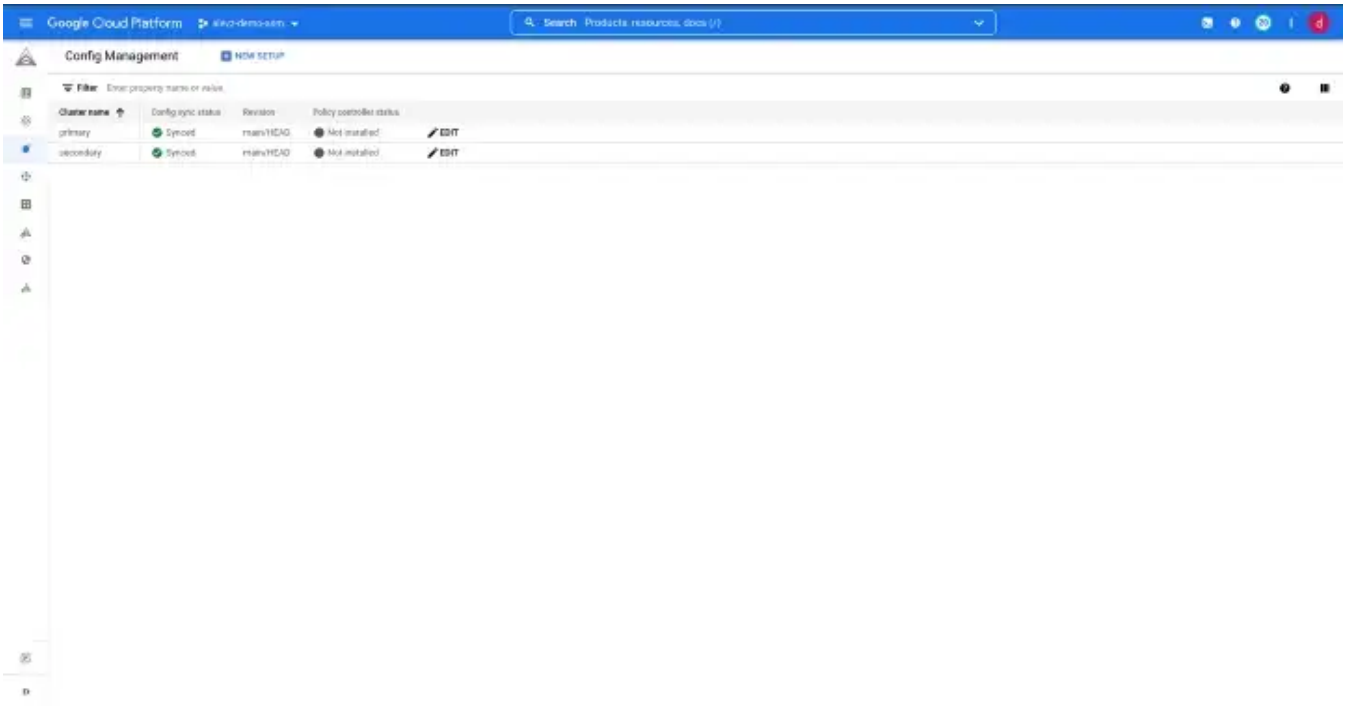
[**Three**] Create the configuration by input the registry url and also set the authentication we prefered. I choose SSH and follow the guides on how to register the secret on GKE.



[**Four**] Select the branch, tag/commit, and other configuration we prefer. As the demo of Bank of Anthos will use the hierarchy type so I will chose it as the source format.



[**Five**] Complete and wait till the cluster sync.



Synced !

That's it particularly we have finished the setup 2x GKE Cluster with ASM and ACM in Anthos platform, now we can modify the istio configuration or any other test if we want to test. Notes: there are many other config eg. multicluster service / multicluster ingress that is interesting to test.

Multi Cluster Serice communication in GKE

Multi Cluster Serice communication in GKE

Multi Cluster Serice communication in GKEfaun.pub

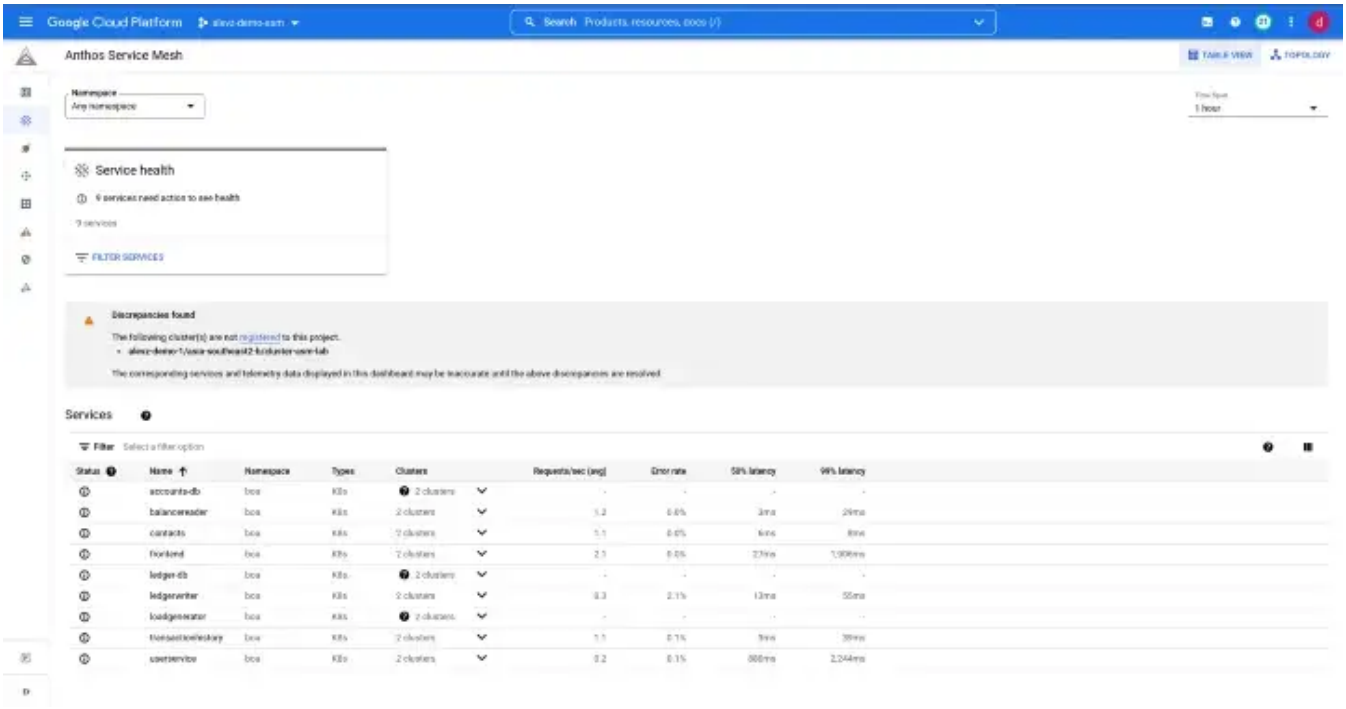
Multi Cluster Ingress — GKE

Creating Multi Cluster Ingress in GKE

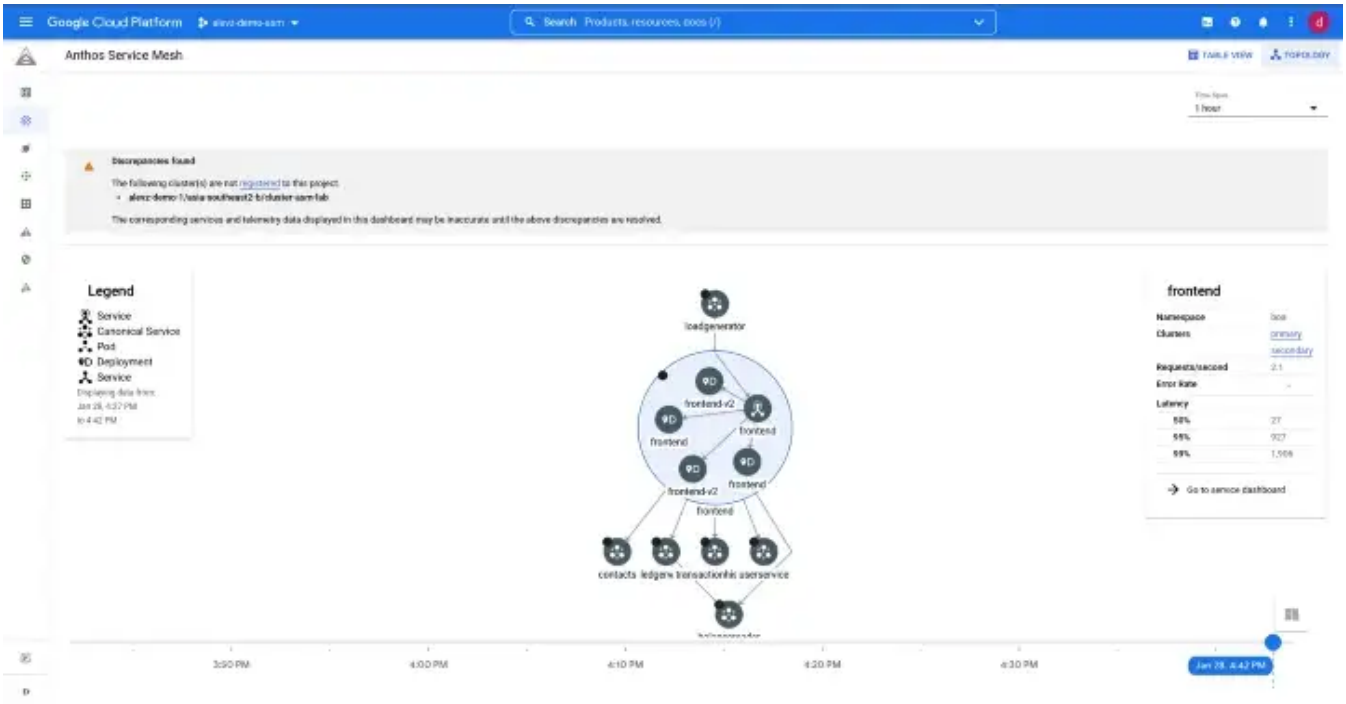
faun.pub

to see the changes we can check the details on GKE or through ASM dashboard:

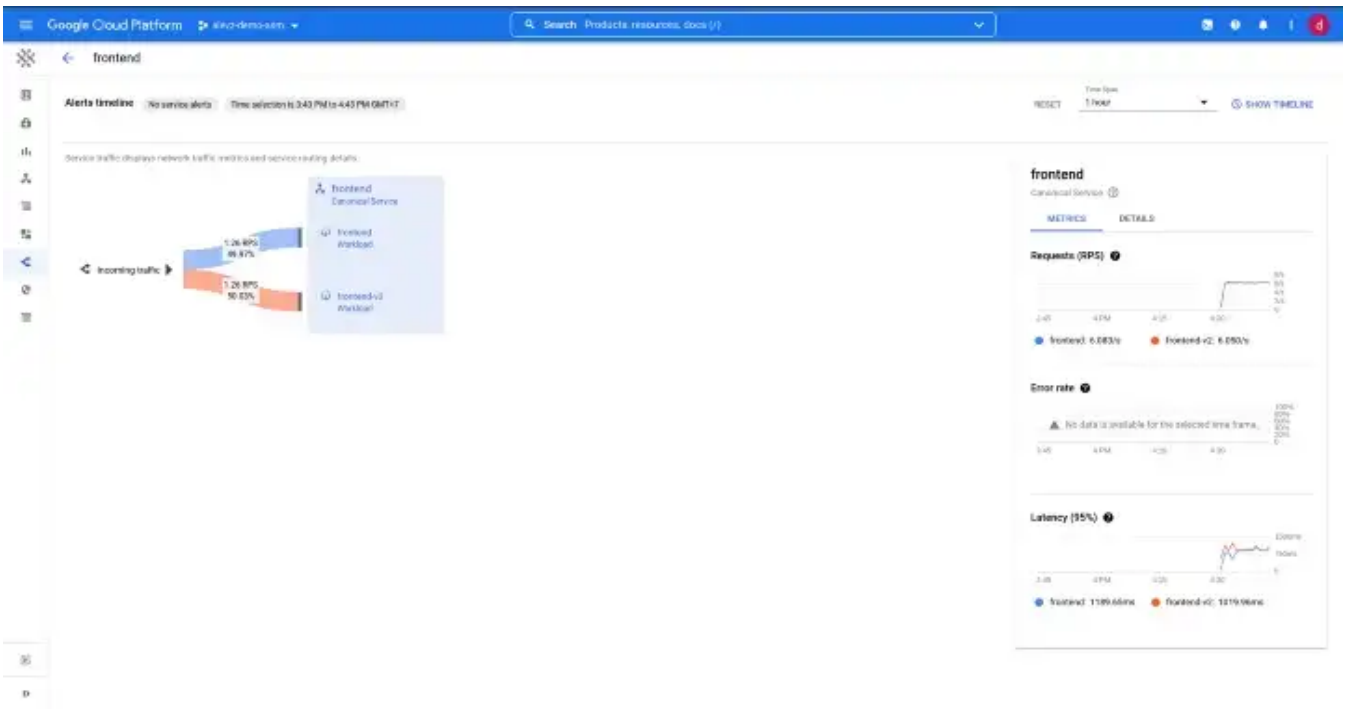
Now we can see the services in ASM Dashboard and helpful information through it.



We can see also from Topology the summary of the services in topology format, which will includes the objects from both clusters in this case (see that we have copies of frontend and frontend-v2 deployment shown).



We can go and focus in each services, however last thing as I add traffic splitting on frontend we can see the rps on each version.



Join FAUN: Website 💻|Podcast 🎙️|Twitter 🐦|Facebook 👥|Instagram 📷|Facebook Group 🗣️|Linkedin Group 💬| Slack 📱|Cloud Native News 📰|More.

If this post was helpful, please click the clap 👋 button below a few times to show your support for the author 👇

Anthos Service Mesh Anthos Config Management Canary Deployments