



Published in FAUN Publication



Johaness Glenn

[Follow](#)Jan 29, 2022 · 6 min read · [Listen](#)

Save



# 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 repetitive 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](https://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

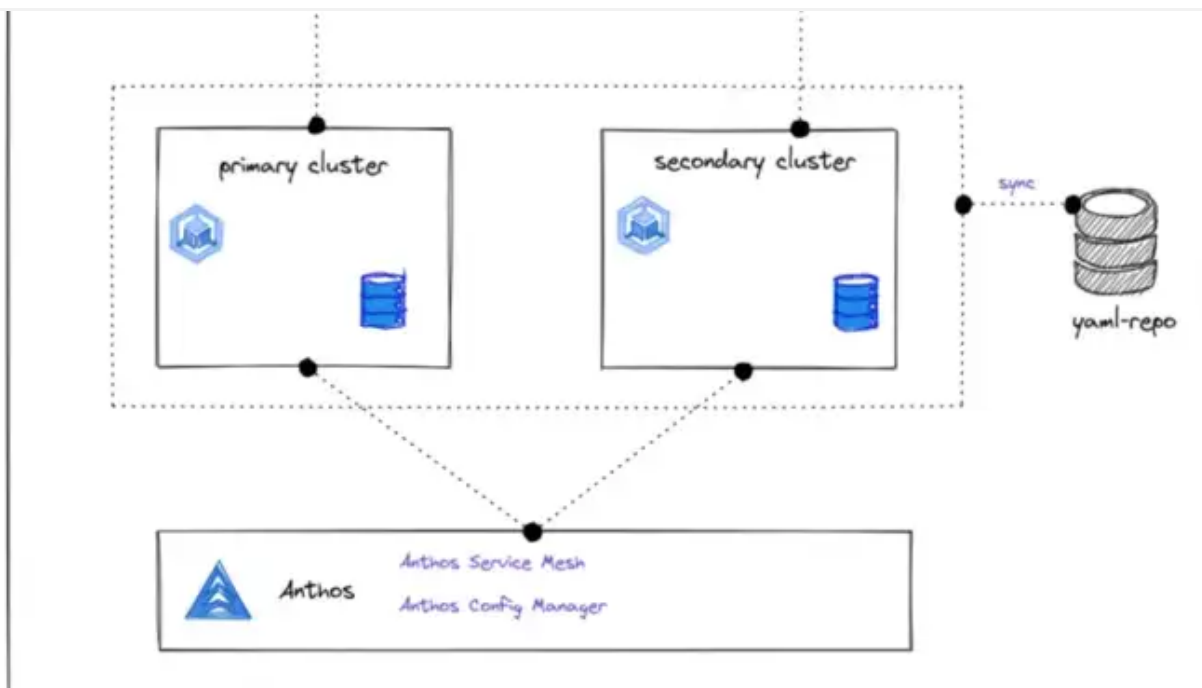
### GitHub - GoogleCloudPlatform/bank-of-anthos: Anthos sample application for retail banking

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

github.com

Open in app ↗

Get unlimited access



## 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.



29



```
1  module "project-services" {
2      source = "terraform-google-modules/project-factory/google//modules/project_services"
3
4      project_id = data.google_client_config.current.project
5      disable_services_on_destroy = false
6      activate_apis = [
7          "compute.googleapis.com",
8          "iam.googleapis.com",
9          "container.googleapis.com",
10         "cloudresourcemanager.googleapis.com",
11         "anthos.googleapis.com",
12         "cloudtrace.googleapis.com",
13         "meshca.googleapis.com",
14         "meshtelemetry.googleapis.com",
15         "meshconfig.googleapis.com",
16         "iamcredentials.googleapis.com",
17         "gkeconnect.googleapis.com",
18         "gkehub.googleapis.com",
19         "monitoring.googleapis.com",
20         "logging.googleapis.com"
21     ]
22 }
23 }
```

apis.tf hosted with ❤ by GitHub

[view raw](#)

```
1  module "asm-primary" {
2      source          = "terraform-google-modules/kubernetes-engine/google//modules/asm"
3      version          = "18.0.0"
4      project_id       = data.google_client_config.current.project
5      cluster_name     = module.primary-cluster.name
6      location         = module.primary-cluster.location
7      cluster_endpoint = module.primary-cluster.endpoint
8      enable_all       = true
9      outdir           = "./asm-dir-${module.primary-cluster.name}"
10
11 }
12
13 module "asm-secondary" {
14     source          = "terraform-google-modules/kubernetes-engine/google//modules/asm"
15     version          = "18.0.0"
16     project_id       = data.google_client_config.current.project
17     cluster_name     = module.secondary-cluster.name
18     location         = module.secondary-cluster.location
19     cluster_endpoint = module.secondary-cluster.endpoint
20     enable_all       = true
21     outdir           = "./asm-dir-${module.secondary-cluster.name}"
22 }
```

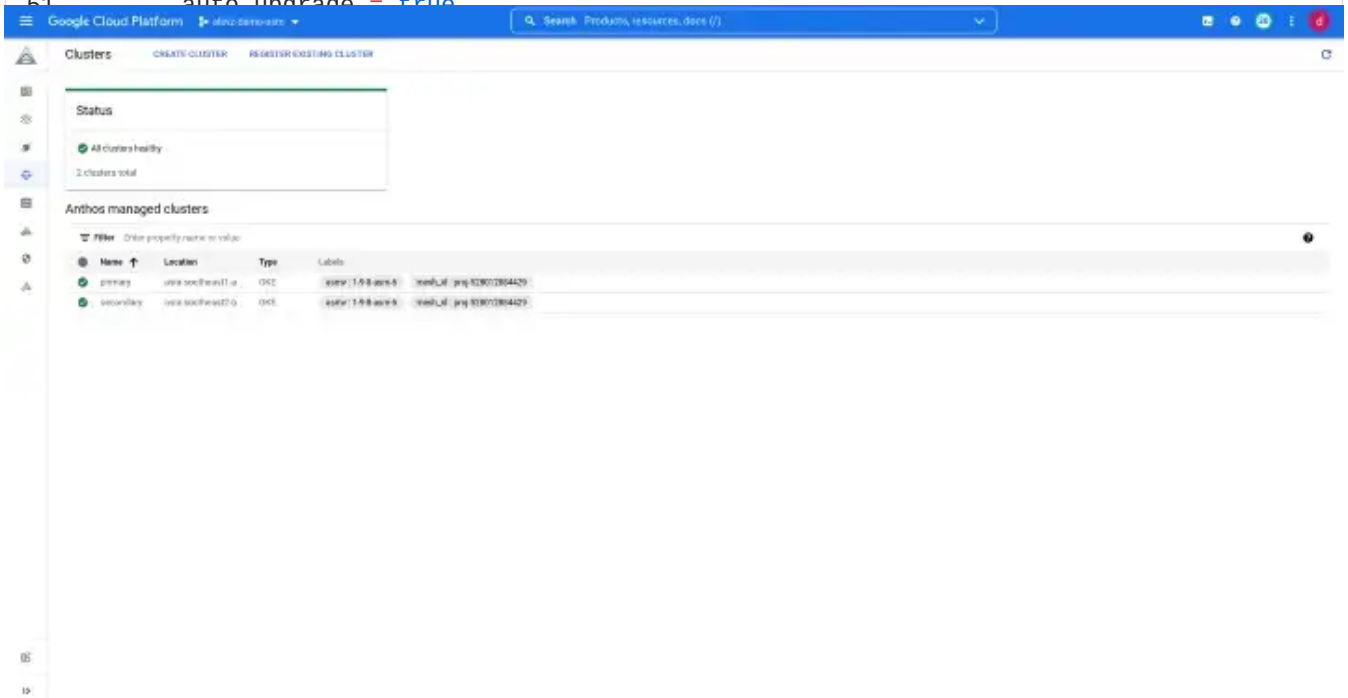
Downloaded from <http://ajph.org/> on November 10, 2015

[view raw](#)[illegible]

```

45   cluster_resource_labels = { "mesh_id" : "proj-${data.google_project.project.number}"
46
47   node_pools = [
48     {
49       name          = "default-node-pool"
50       autoscaling    = false
51       auto_upgrade   = true

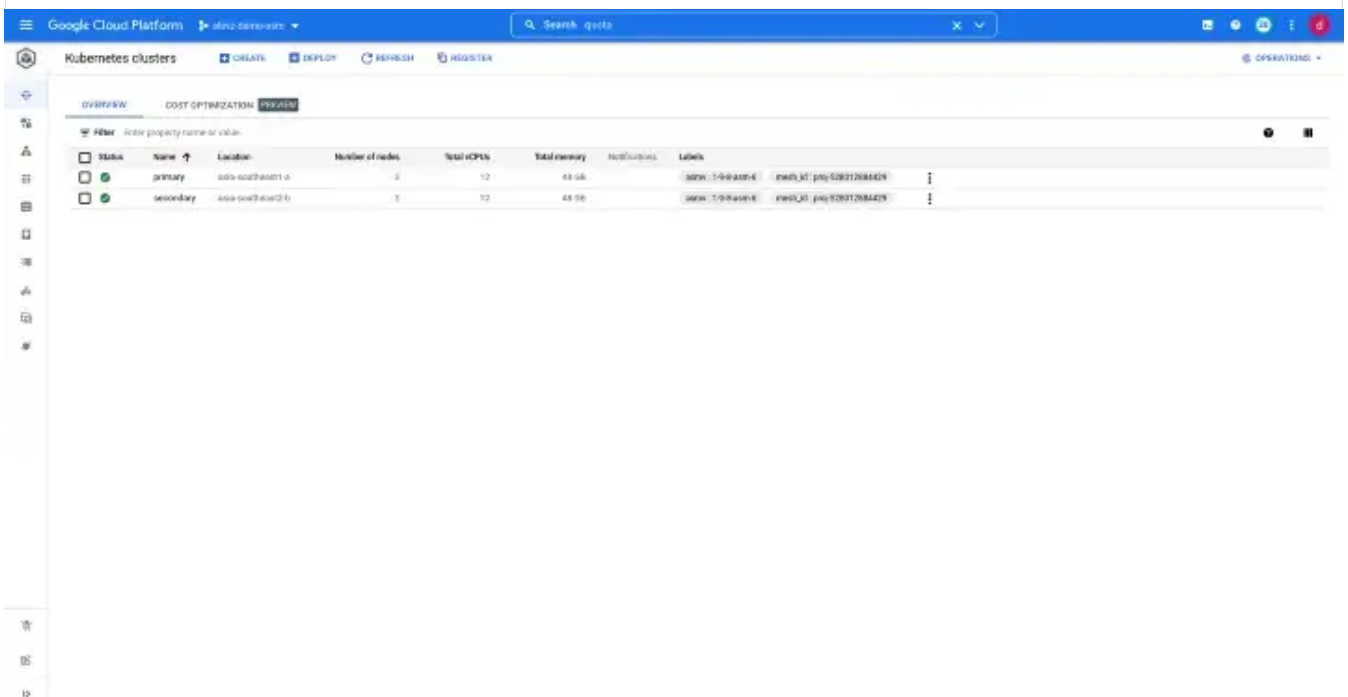
```



```

7   cluster_endpoint = module.primary-cluster.endpoint
8   gke_hub_membership_name = "primary"
9   gke_hub_sa_name = "primary"

```



```

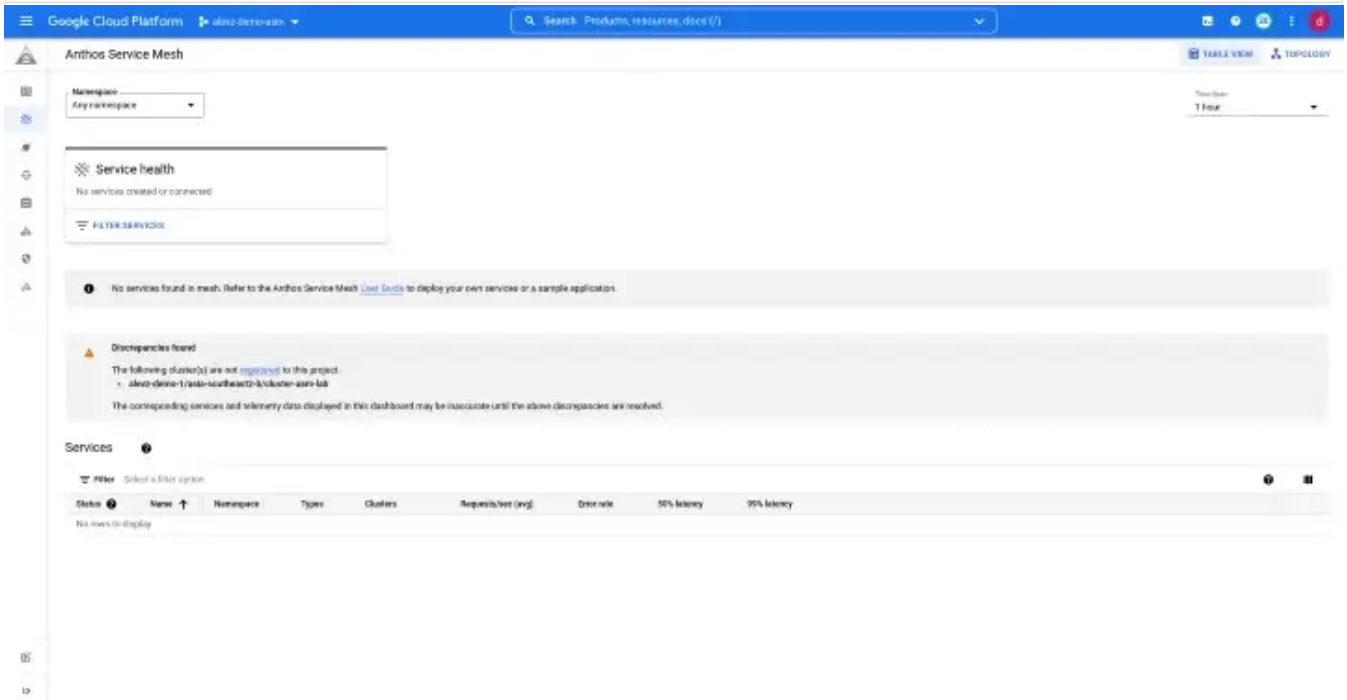
3   region = var.primary_region
4 }
5
6 data "google_client_config" "current" {}
7
8 data "google_project" "project" {
9   project_id = var.project_id

```

```

10 }
11
12 output "project" {
13     value = data.google_client_config.current.project
14 }

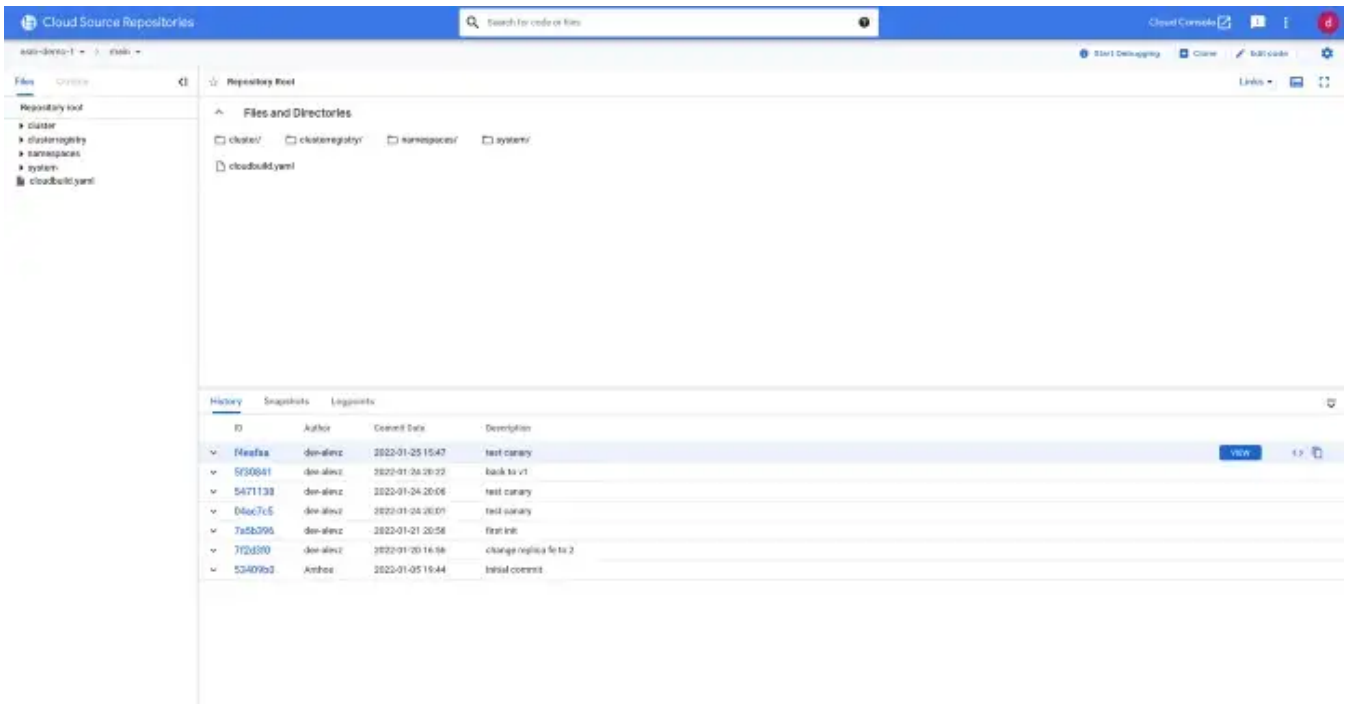
```



```

6     description = "The primary region to be used"
7 }
8 variable "primary_zones" {
9     description = "The primary zones to be used"
10 }
11
12 variable "secondary_region" {
13     description = "The secondary region to be used"
14 }
15 variable "secondary_zones" {
16     description = "The secondary zones to be used"
17 }
18
19 variable "vpc" {
20     description = "The default VPC value"
21 }
22
23 variable "subnet" {
24     description = "The default subnet value"

```

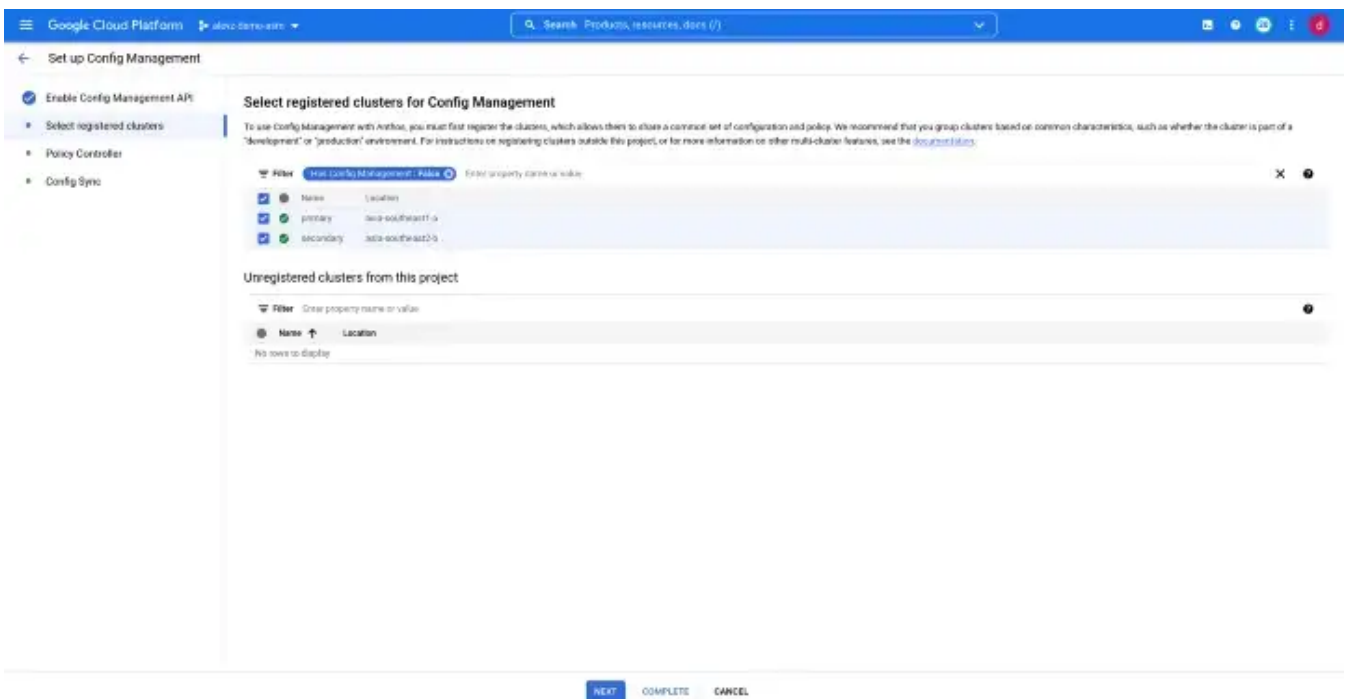


I push the manifests I need on the code source repo

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 preferred. I choose SSH and follow the guides on how to register the secret on GKE.



**Set up Config Management**

- Enable Config Management API
- Select registered clusters
- Policy Controller
- Config Sync**

**Config Sync**

Config Sync continuously reconciles the state of your clusters with a central set of configuration stored in one or more Git repositories. This feature allows you to manage customer configuration with an auditable, transactional, and version-controlled deployment process.

☒ Enable Config Sync. Config Sync allows cluster operators to manage Kubernetes deployments via a Git repository.

Repository \*

Custom

You can select a Google sample to use your own repository.

URL \*

url

The URL of the Git repository to use as the source of truth.

Authentication type

SSH

Select Authentication type of your Git repository. If your repo does not require authentication for read-only access, you should set Authentication type to "None". If you change your repository visibility you may need to update this setting.

**Using SSH keypair**

- Create an SSH keypair to allow the Operator to authenticate to your Git repository.
 

```
$ ssh-keygen -t rsa -b 4096 \
  -C "[GIT_REPOSITORY_USERNAME]" \
  -N "" \
  -f "[path/to/KEYPAIR-FILENAME]"
```
- Configure your repo to recognize the newly-created public key.
  - Cloud Source Repositories
  - Bitbucket
  - GitLab
  - GitHub
- Add the private key to a new Secret in the cluster. Substitute the name of the private key (the one without the pub suffix) where you see `[path/to/KEYPAIR-FILENAME-KEY-FILENAME]`. This step must be performed separately on every

**Summary**

Config Sync is free to use for all GKE users.

Latest version: 1.10.1

Documentation: [Go to documentation](#)

COMPLETE CANCEL

[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.

**Set up Config Management**

- Enable Config Management API
- Select registered clusters
- Policy Controller
- Config Sync**

**Config Sync**

Config Sync continuously reconciles the state of your clusters with a central set of configuration stored in one or more Git repositories. This feature allows you to manage customer configuration with an auditable, transactional, and version-controlled deployment process.

☒ Enable Config Sync. Config Sync allows cluster operators to manage Kubernetes deployments via a Git repository.

Repository \*

Custom

You can select a Google sample to use your own repository.

URL \*

url

The URL of the Git repository to use as the source of truth.

Authentication type

SSH

Select Authentication type of your Git repository. If your repo does not require authentication for read-only access, you should set Authentication type to "None". If you change your repository visibility you may need to update this setting.

**Using SSH keypair**

- Create an SSH keypair to allow the Operator to authenticate to your Git repository.
 

```
$ ssh-keygen -t rsa -b 4096 \
  -C "[GIT_REPOSITORY_USERNAME]" \
  -N "" \
  -f "[path/to/KEYPAIR-FILENAME]"
```
- Configure your repo to recognize the newly-created public key.
  - Cloud Source Repositories
  - Bitbucket
  - GitLab
  - GitHub
- Add the private key to a new Secret in the cluster. Substitute the name of the private key (the one without the pub suffix) where you see `[path/to/KEYPAIR-FILENAME-KEY-FILENAME]`. This step must be performed separately on every

**Advanced Settings**

4. Finally, delete the private key from the local disk or otherwise protect it.

Branch

main

The branch of the repository to sync from. Default: master

Tag / Commit

HEAD

An revision (tag or hash) to check out. Default: HEAD

Policy directory

The path within the repository to the top of the policy directory to sync. Default: the root directory of the repository

Sync wait

15

Period in seconds between consecutive syncs. Default: 15

Git proxy

Must be a valid URL, if no protocol is supplied, default to HTTPS

Source format \*

hierarchy

Specifies whether the repo is in "unstructured" or "hierarchy" mode. Default: "unstructured"

**Config Management version**

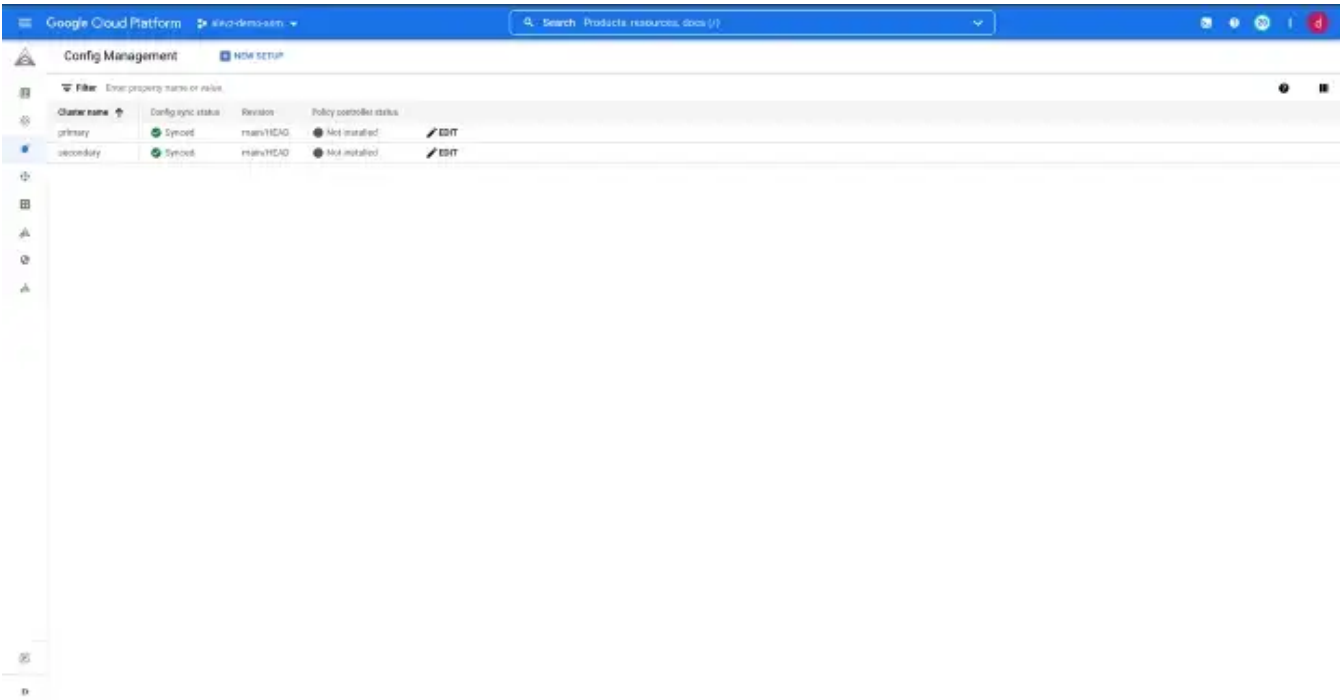
Version

1.10.1

The ACM product version, defaults to latest released version.

COMPLETE CANCEL

[Five] Complete and wait till the cluster sync.



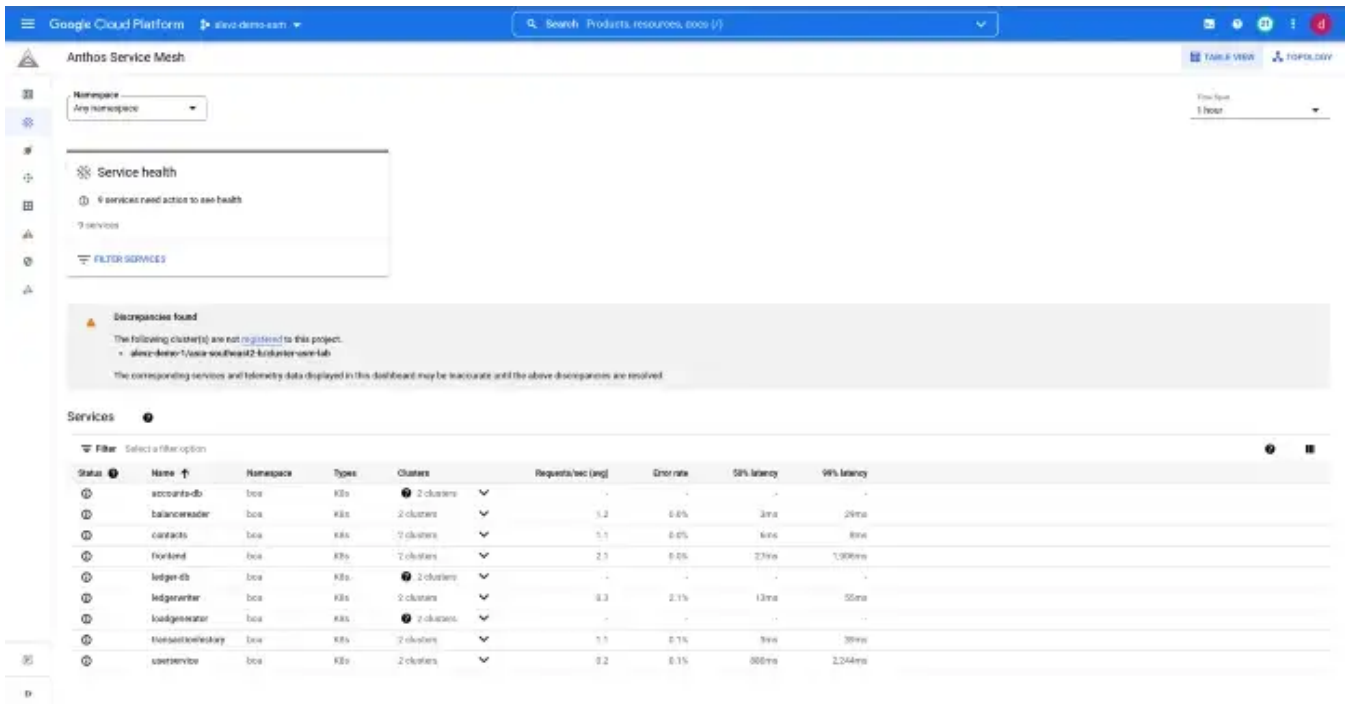
Synced !

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.

<div><b>Multi Cluster Serice communication in GKE</b></div> <div>Multi Cluster Serice communication in GKE</div> <div>Multi Cluster Serice communication in GKEfaun.pub</div>	
<div><b>Multi Cluster Ingress — GKE</b></div> <div>Creating Multi Cluster Ingress in GKE</div> <div>faun.pub</div>	

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

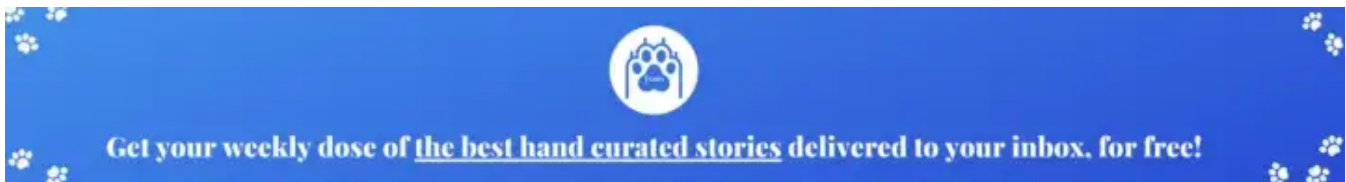
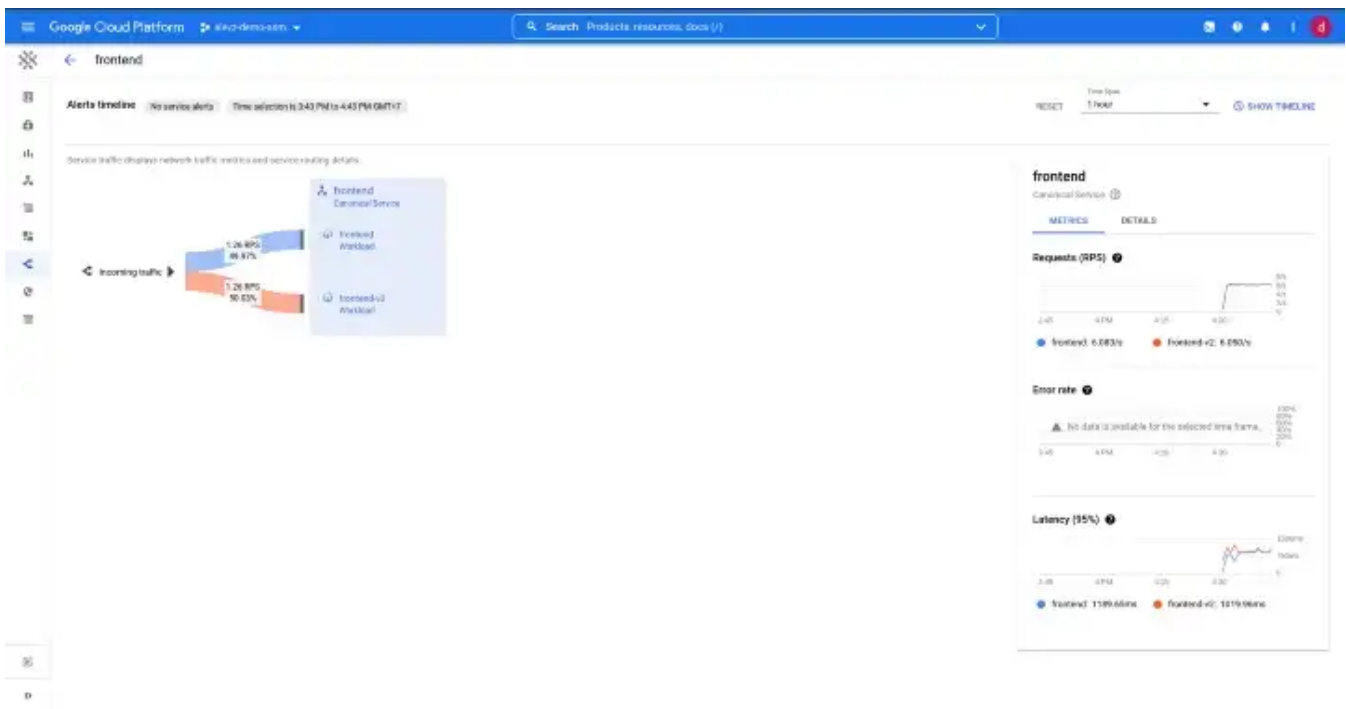
Now we can see the services in ASM Dashboard and helpful information through it.



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](#)

---

## Sign up for FAUN

By FAUN Publication

Medium's largest and most followed independent DevOps publication. Join thousands of developers and DevOps enthusiasts. [Take a look.](#)

Emails will be sent to hamdi.bouhani@dealroom.co. [Not you?](#)



Get this newsletter