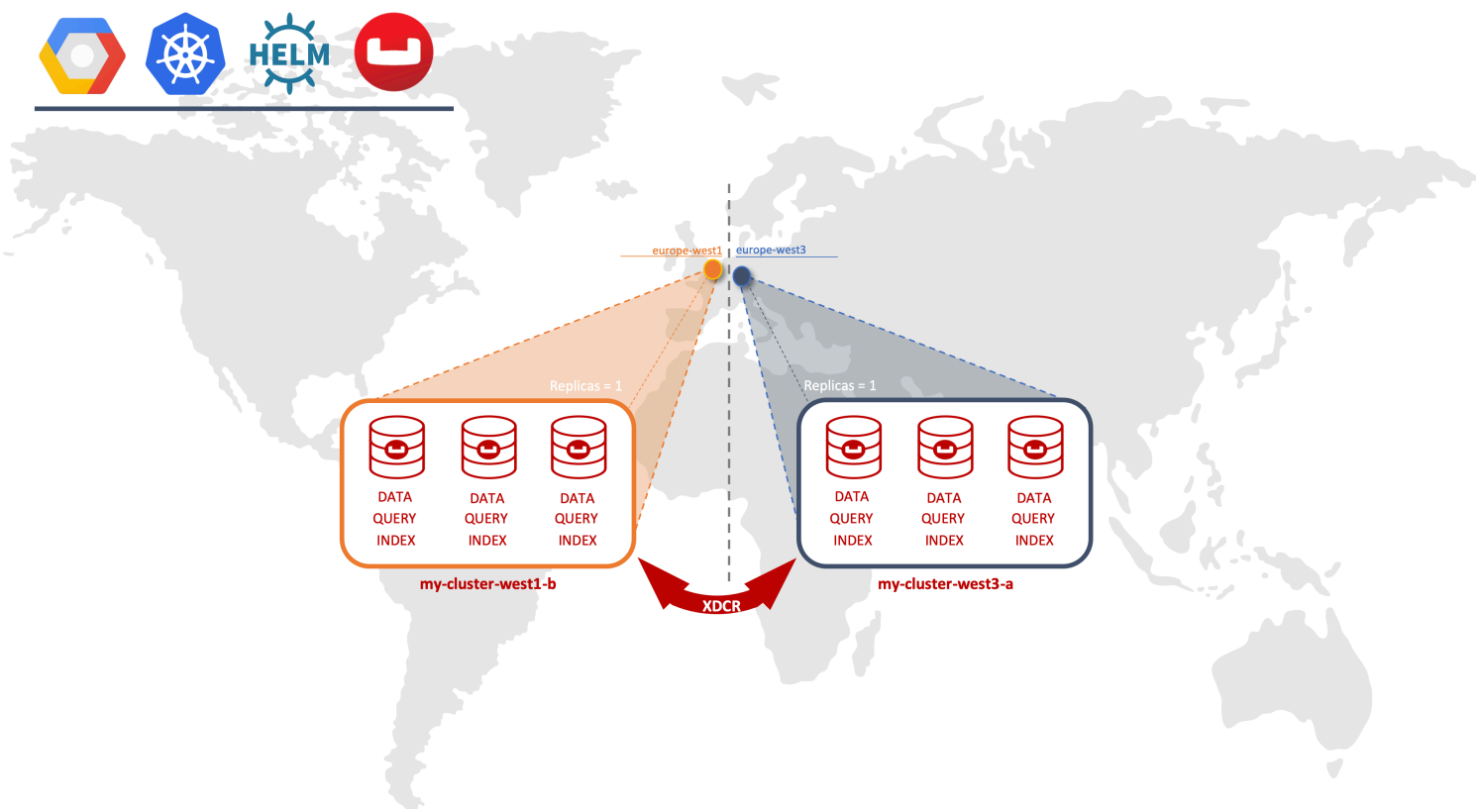
Couchbase Helm Charts on Google Kubernets Engine (GKE)

Scope

This page describes how to set up XDCR between two clusters (Active/Pasive or Active/Active) using Helm to properly support the Couchbase Autonomous Operator in GKE environment.



0. Prerequisites

1. Google Cloud Account
2. Install Google Cloud SDK
3. Install Kubernetes

1. Quick Command Steps

1. Google SDK Setup & Create your project ***my-couchbase-helm*** `$ gcloud init --console-only`
2. **Log in** `$ gcloud auth login`
3. **Set default config values**

config	command
Set your default project	`$ gcloud config set project my-couchbase-helm`
Set default region	`$ gcloud config set compute/region europe-west3`
Set default zone	`$ gcloud config set compute/zone europe-west3-a`

1. **Setup Network Setup**

4.1. Create Custom Network configuration `$ gcloud compute networks create helm-network --subnet-mode custom`

4.2. Create Subnet on region europe-west1

```
$ gcloud compute networks subnets create my-subnet-europe-west1 --network helm-network --region europe-west1 --range 10.0.0.0/12
```

4.3. Create Subnet on region europe-west3

```
$ gcloud compute networks subnets create my-subnet-europe-west3 --network helm-network --region europe-west3 --range 10.16.0.0/12
```

4.4. Add Firewall rules:
`$ gcloud compute firewall-rules create my-network-allow-all-private --network helm-network --direction INGRESS --source-ranges 10.0.0`

2. **Provisioning Instances for the Kubernetes-Helm Cluster**

5.1. Create instances for Cluster 1 in **europe-west1 zone b**
`$ gcloud container clusters create my-cluster-europe-west1-b --machine-type n1-standard-2 --cluster-version 1.13.6-gke.0 --zone europ`
5.2. Create three instances for cluster 2 in **europe-west3 zone a**
`$ gcloud container clusters create my-cluster-europe-west3-a --machine-type n1-standard-2 --cluster-version 1.13.6-gke.0 --zone europ`
5.3. List Clusters `$ gcloud container clusters list`

3. **Get Cluster Credentials** and **setup Kubernetes** environment

6.1. Get Cluster Credentials from **my-cluster-europe-west1-b**
`$ gcloud container clusters get-credentials my-cluster-europe-west1-b --zone europe-west1-b --project my-couchbase-helm` 6.2. Setup your local Kubernetes with the GKE Cluster
`$ kubectl create clusterrolebinding your-admin-binding --clusterrole cluster-admin --user $(gcloud config get-value account)`

4. **Helm setup**

7.1. Create Helm Server (Tiller) Service Account: `$ kubectl create -f rbac-tiller-development.yaml` 7.2. Initialize Helm
`$ helm init --service-account tiller` 7.3. Add the chart repository to **helm**
`helm repo add couchbase https://couchbase-partners.github.io/helm-charts/`

5. Deploy default **Couchbase Operator** `$ helm install couchbase/couchbase-operator`

6. Deploy custom **Couchbase Server**

9.1. Config the cluster to deploy into myvalues-clustercfg.yaml file and install:
`$ helm install --values myvalues-clustercfg.yaml --set couchbaseCluster.name=my-cluster-west1b couchbase/couchbase-cluster` 9.2. List pods: `$ kubectl get pods --watch`

7. **Repeat steps from 6 to 9 referencing to my-cluster-europe-west3-a**

10.1. Get Cluster Credentials from **my-cluster-europe-west3-a**
`$ gcloud container clusters get-credentials my-cluster-europe-west3-a --zone europe-west3-a --project my-couchbase-helm` 10.2. Setup your local Kubernetes with the GKE Cluster
`$ kubectl create clusterrolebinding your-admin-binding --clusterrole cluster-admin --user $(gcloud config get-value account)`
10.3. **Helm setup** `$ kubectl create -f rbac-tiller-development.yaml` `$ helm init --service-account tiller`
`$ helm repo add couchbase https://couchbase-partners.github.io/helm-charts/` 10.4. **Deploy default Couchbase Operator**
`$ helm install couchbase/couchbase-operator` 10.5. **Deploy custom Couchbase Server**
`$ helm install --values myvalues-clustercfg.yaml --set couchbaseCluster.name=my-cluster-west3a couchbase/couchbase-cluster` 10.6. **List pods** `$ kubectl get pods --watch`

8. **Setup XDCR between both clusters using internal IP addresses**

2. What is Helm?

Helm is the package manager for Kubernetes. It is the easiest way to install new applications on your Kubernetes cluster.

This website contains a complete directory of all charts available to install from the official repository. We have grouped them into categories to make browsing for new applications quick and easy.

But how do you actually install them? The first step is to install Helm on your laptop by following this guide.

Make sure you have kubectl working with a valid context pointing to an existing cluster, or local minikube installation, then simply run helm init and it will automatically configure everything required to start installing charts.

Behind the scenes running helm init will install the server side component called Tiller. This all happens magically so you shouldn't need to do anything.

If you find an application that you'd like to install while browsing this site simply click the visit website button on the listing. You'll generally find the install command that you can copy and paste into your terminal.

3. GKE Setup

Like all cloud providers there are two main steps in configuring your cloud environment - the virtual network provisioning then the Kubernetes cluster provisioning.

1. Setup Google SDK
2. Setup Network
3. Setup Cluster

3.1. Google SDK Setup

1. Init Google Cloud SDK

```
$ gcloud init --console-only
```

Set you Google Cloud SDK configuration Your first time to setup your Google SDK will create the default configuration. Otherwise it will ask you to choose if you want to create a new configuration or reset the current default values.

```
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

You can skip diagnostics next time by using the following flag:
  gcloud init --skip-diagnostics

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.

Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

You must log in to continue. Would you like to log in (Y/n)?  Y
```

Log in with your user account

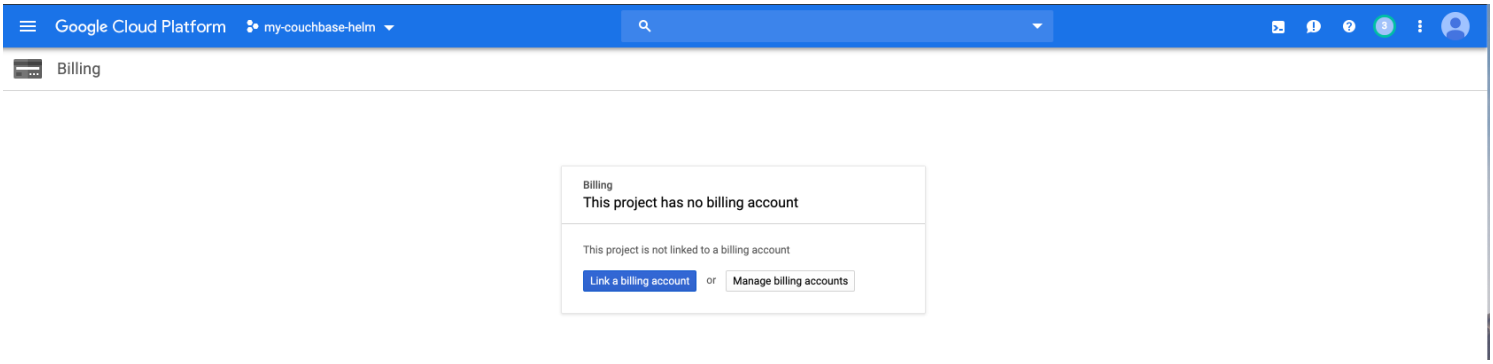
```
You must log in to continue. Would you like to log in (Y/n)?  Y

Go to the following link in your browser:

    https://accounts.google.com/o/oauth2/auth?redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&prompt=select_account&response_type=code&cli
ent_id=11111945654.apps.googleusercontent.com&scope=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.c
om%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fappengine.admin+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcompute+http
s%3A%2F%2Fwww.googleapis.com%2Fauth%2Faccounts.reauth&access_type=offline


Enter verification code: 3/WBSSSDEw_phCo1KwrXUsdywps331h9qe-qMVfcfGOZaxkm_ZyXAqapq
You are logged in as: [your.email@domain.com].

Pick cloud project to use:
 [1] other-project
 [2] Create a new project
Please enter numeric choice or text value (must exactly match list
item):
```

Create your own project

```
Pick cloud project to use:
 [1] other-project
 [2] Create a new project
Please enter numeric choice or text value (must exactly match list
item):  2

Enter a Project ID. Note that a Project ID CANNOT be changed later.
Project IDs must be 6-30 characters (lowercase ASCII, digits, or
hyphens) in length and start with a lowercase letter. my-couchbase-helm
Your current project has been set to: [my-couchbase-helm].

Not setting default zone/region (this feature makes it easier to use
[gcloud compute] by setting an appropriate default value for the
--zone and --region flag).
See https://cloud.google.com/compute/docs/gcloud-compute section on how to set
default compute region and zone manually. If you would like [gcloud init] to be
able to do this for you the next time you run it, make sure the
Compute Engine API is enabled for your project on the
https://console.developers.google.com/apis page.

Your Google Cloud SDK is configured and ready to use!
```

Associate Billing Account to your project.

Go to your Google Cloud Platform, select your new project and go to billing tab.



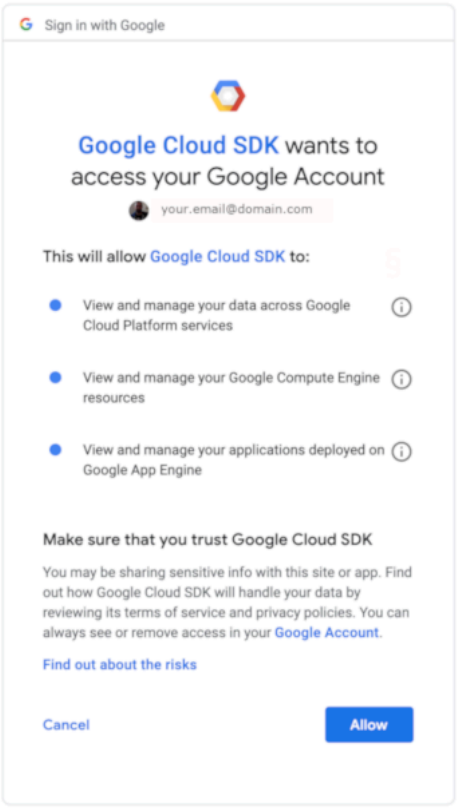
Link your project to your billing account.

Note: Google Cloud Platform offers you a Free Trial of 300$ during 365 days, it is a great deal.

1. **Log in**

When the gcloud command has been installed and added to your PATH, log in with the following command:

```
$ gcloud auth login
```



Tip: The gcloud command can support multiple logins. You can select the user to run as with:

```
gcloud config set account your.email@domain.com
```

You can also remove the login authentication locally with:

```
gcloud auth revoke your.email@domain.com
```

1. **Set your default configuration**

Set your default project: `$ gcloud config set project my-couchbase-helm Updated property [core/project]`

Set default region:

```
$ gcloud config set compute/region europe-west3
Updated property [compute/region].
```

Set default zone: `$ gcloud config set compute/zone europe-west3-a Updated property [compute/zone].`

Check your nearby available region/zones here.

3.2 Setup Network

For the purposes of this document we will manually configure our subnets so we are able to add in the necessary firewall rules to allow XDCR between Couchbase clusters in different GKE clusters. We create two non-overlapping subnets in the 10.0.0.0/8 RFC-1918 private address space in different regions, then allow all ingress traffic from the 10.0.0.0/8 prefix via a firewall rule. By default network traffic is dropped between different GKE clusters.

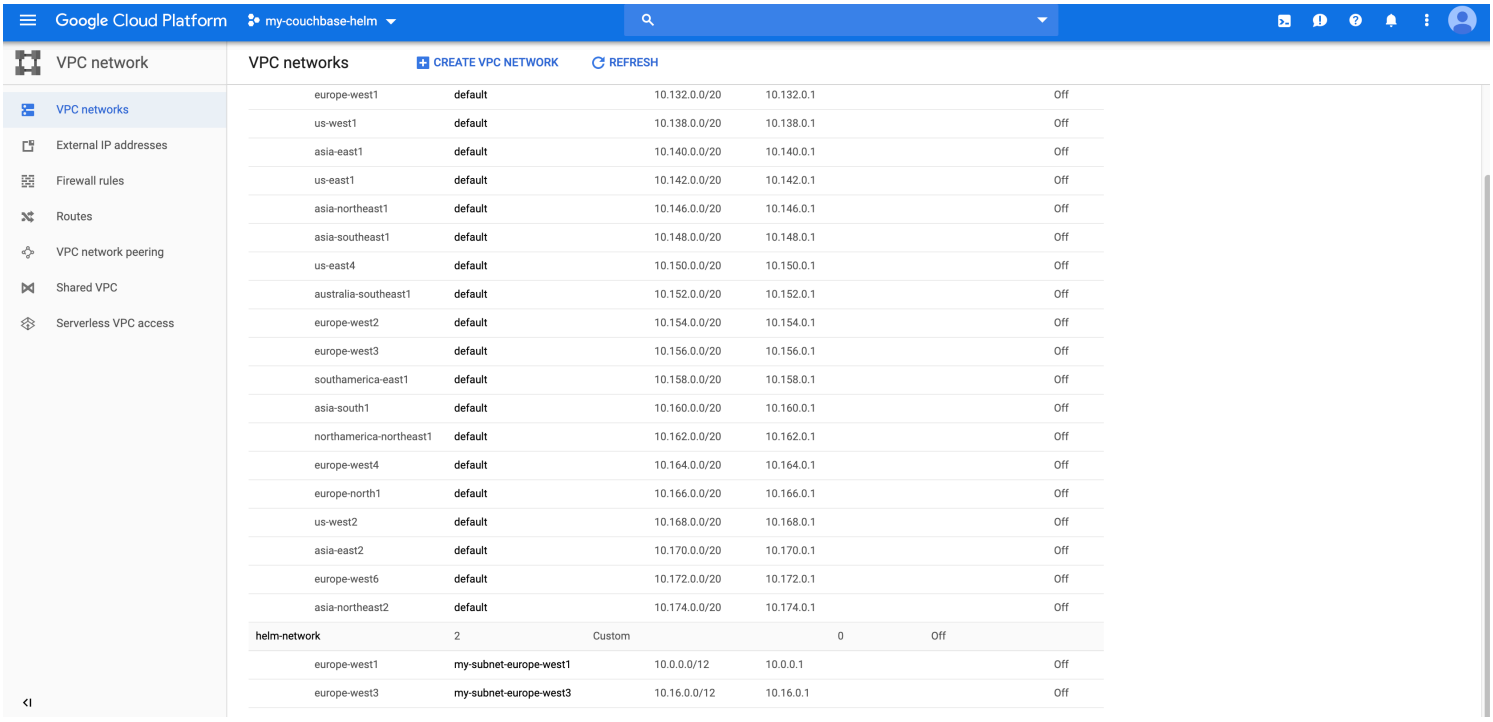
```
$ gcloud compute networks create helm-network --subnet-mode custom
Created [https://www.googleapis.com/compute/v1/projects/my-couchbase-helm/global/networks/helm-network].
NAME          SUBNET_MODE  BGP_ROUTING_MODE  IPV4_RANGE  GATEWAY_IPV4
helm-network  CUSTOM       REGIONAL
```

Let's create now the subnet for the first cluster in region europe-west1:

```
$ gcloud compute networks subnets create my-subnet-europe-west1 --network helm-network --region europe-west1 --range 10.0.0.0/12
Created [https://www.googleapis.com/compute/v1/projects/my-couchbase-helm/regions/europe-west1/subnetworks/my-subnet-europe-west1].
NAME                  REGION        NETWORK       RANGE
my-subnet-europe-west1  europe-west1  helm-network  10.0.0.0/12
```

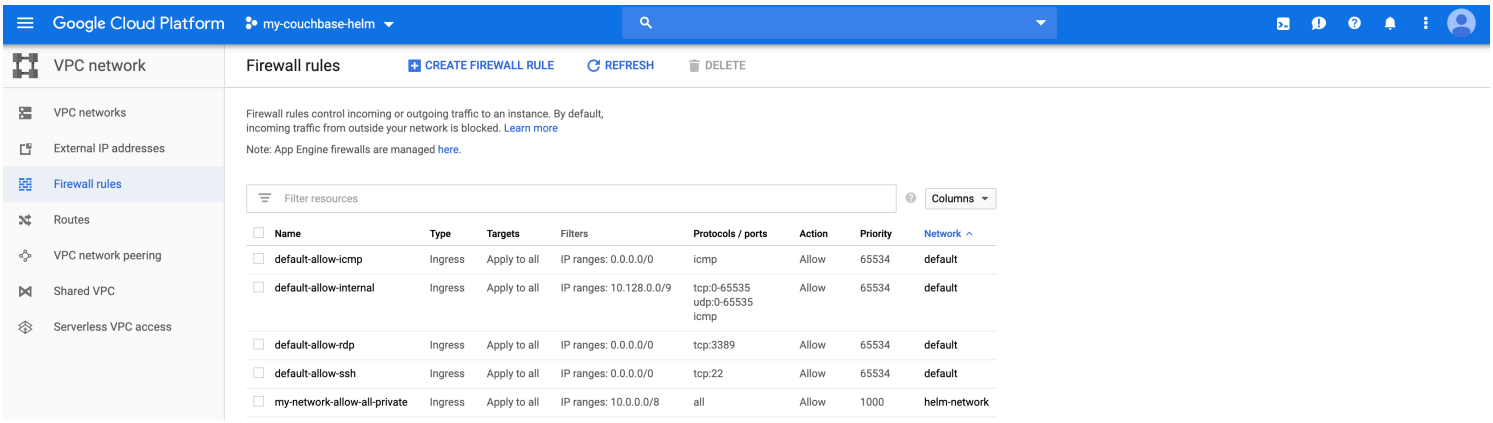
And Subnet for the second cluster in region europe-west3:

```
$ gcloud compute networks subnets create my-subnet-europe-west3 --network helm-network --region europe-west3 --range 10.16.0.0/12
Created [https://www.googleapis.com/compute/v1/projects/my-couchbase-helm/regions/europe-west3/subnetworks/my-subnet-europe-west3].
NAME                  REGION        NETWORK       RANGE
my-subnet-europe-west3  europe-west3  helm-network  10.16.0.0/12
```

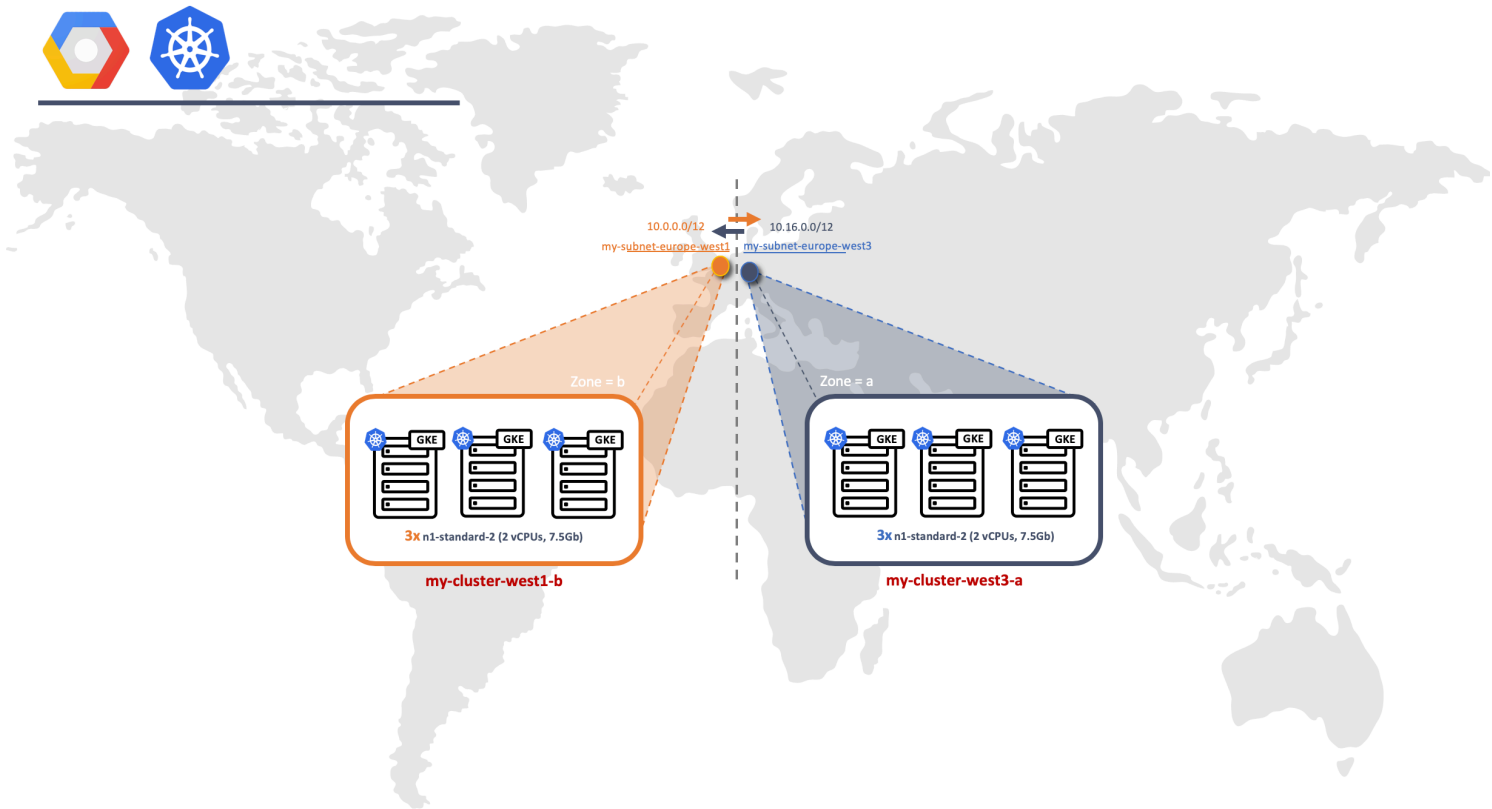


Now we need to configure the firewall rules to enable the communication between both subnets:

```
$ gcloud compute firewall-rules create my-network-allow-all-private --network helm-network --direction INGRESS --source-ranges 10.0.0.0/8 --allow all
Creating firewall...⠹Created [https://www.googleapis.com/compute/v1/projects/my-couchbase-helm/global/firewalls/my-network-allow-all-private].
Creating firewall...done.

NAME                          NETWORK       DIRECTION  PRIORITY  ALLOW  DENY  DISABLED
my-network-allow-all-private  helm-network  INGRESS    1000      all          False
```



3.3. Provisioning Instances for the Kubernetes-Helm Cluster



1. **Check your cluster version per region/zone**

You can check which Kubernetes versions are available and default in a given zone from Google Cloud Platform Console or by using the gcloud command-line tool.

```
$ gcloud container get-server-config --zone europe-west3-a
Fetching server config for europe-west3-a
defaultClusterVersion: 1.12.7-gke.10
defaultImageType: COS
validImageTypes:
- COS_CONTAINERD
- COS
- UBUNTU
validMasterVersions:
- 1.13.6-gke.0
- 1.13.5-gke.10
- 1.12.7-gke.17
- 1.12.7-gke.10
- 1.12.6-gke.11
- 1.11.9-gke.13
- 1.11.9-gke.8
- 1.11.8-gke.6
validNodeVersions:
- 1.13.6-gke.0
- 1.13.5-gke.10
- 1.12.7-gke.17
- 1.12.7-gke.10
- 1.12.7-gke.7
...
- 1.7.12-gke.2
- 1.6.13-gke.1
```

1. **Check** Google cloud machine types

A machine type specifies a particular collection of virtualized hardware resources available to a virtual machine (VM) instance, including the system memory size, virtual CPU (vCPU) count, and maximum persistent disk capability.

```
gcloud compute machine-types list
```

Note: Trial Google Account is limited to 12 vCPUs. In the context of this example, we will choose **n1-standar-2** (2 vCPUs, RAM 7.5Gb). Choose the right instance size for hosting Couchbase Server. Please check the Couchbase sizing guidelines and the best practices with Multidimensional Scaling (MDS) to properly sizing your project and/or contact Couchbase Professional Services.

1. **Create instances for Cluster 1 in europe-west1 zone b**

```
$ gcloud container clusters create my-cluster-europe-west1-b --machine-type n1-standard-2 --cluster-version 1.13.6-gke.0 --zone europe-west
1-b --network helm-network --subnetwork my-subnet-europe-west1 --num-nodes 3

...

To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload\_/gcloud/europe-west1-b/my-cluster-eur
ope-west1-b?project=my-couchbase-helm
kubeconfig entry generated for my-cluster-europe-west1-b.
NAME                      LOCATION        MASTER_VERSION  MASTER_IP      MACHINE_TYPE   NODE_VERSION  NUM_NODES  STATUS
my-cluster-europe-west1-b  europe-west1-b  1.13.6-gke.0    35.195.13.148  n1-standard-2  1.13.6-gke.0  3          RUNNING
```

1. **Create instances for cluster 2 in europe-west3 zone a**

```
$ gcloud container clusters create my-cluster-europe-west3-a --machine-type n1-standard-2 --cluster-version 1.13.6-gke.0 --zone europe-west
3-a --network helm-network --subnetwork my-subnet-europe-west3 --num-nodes 3

...

kubeconfig entry generated for my-cluster-europe-west3-a.
NAME                      LOCATION        MASTER_VERSION  MASTER_IP      MACHINE_TYPE   NODE_VERSION  NUM_NODES  STATUS
my-cluster-europe-west3-a  europe-west3-a  1.13.6-gke.0    35.198.169.157  n1-standard-2  1.13.6-gke.0  3          RUNNING
```

1. **List Clusters**

```
$ gcloud container clusters list
NAME                      LOCATION        MASTER_VERSION  MASTER_IP      MACHINE_TYPE   NODE_VERSION  NUM_NODES  STATUS
my-cluster-europe-west1-b  europe-west1-b  1.13.6-gke.0    35.195.13.148  n1-standard-2  1.13.6-gke.0  3          RUNNING
my-cluster-europe-west3-a  europe-west3-a  1.13.6-gke.0    35.198.169.157  n1-standard-2  1.13.6-gke.0  3          RUNNING
```

1. **Get Credentials** When the clusters are running (the gcloud command will block and complete when the clusters are healthy) you can install credentials into your Kubernetes configuration with the following:

Credentials from **my-cluster-europe-west1-b** Cluster

```
gcloud container clusters get-credentials my-cluster-europe-west1-b --zone europe-west1-b --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west1-b.
```

Credentials from **my-cluster-europe-west3-a** Cluster

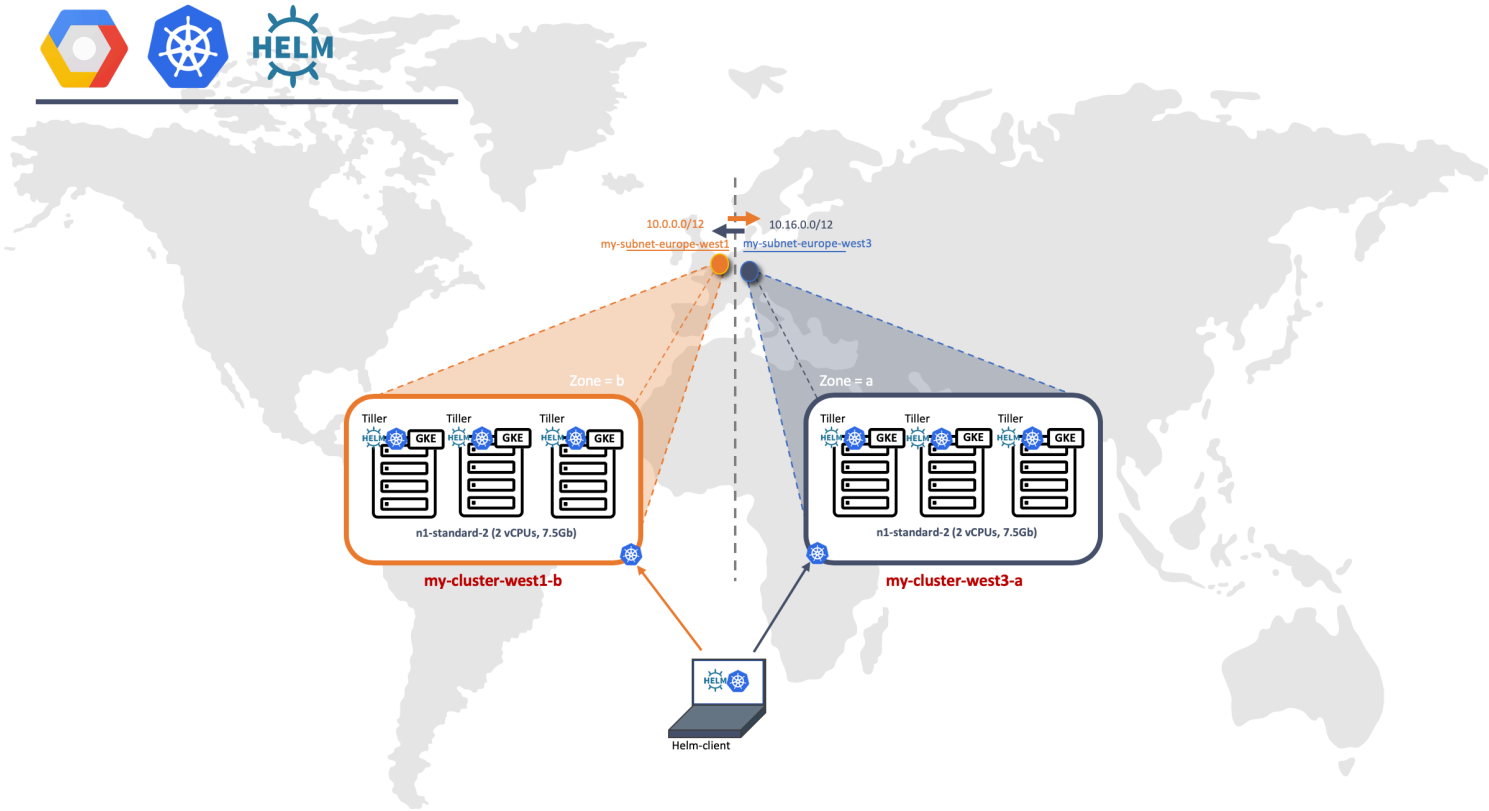
```
$ gcloud container clusters get-credentials my-cluster-europe-west3-a --zone europe-west3-a --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west3-a.
```

4. Deploying Couchbase Operator and Server with Helm charts

4.1. Kubernetes environment Setup

```
$ kubectl create clusterrolebinding your-admin-binding --clusterrole cluster-admin --user $(gcloud config get-value account)
clusterrolebinding.rbac.authorization.k8s.io/jose-molina-admin-binding created
```

4.2. Setup Helm



Setting up Helm consists of installing the Helm client (**helm**) on your computer, and installing the Helm server (Tiller) on your Kubernetes or OpenShift cluster. Once you've set up Helm, you can then use official Couchbase Helm charts to deploy the Operator and the Couchbase Server cluster.

1. Install helm client on your local machine.
2. Installing Helm Server (Tiller) for development

For development use-cases, the Tiller service can be given access to deploy charts into any namespace of your Kubernetes cluster. This also means that resources created by the chart, such as the custom resource definition (CRD), are allowed when Tiller is given this level of privilege.

To create RBAC rules for Tiller with cluster-wide access create the following ServiceAccount and save to rbac-tiller.yaml:

```
$ nano rbac-tiller-development.yaml
```

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: tiller
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: tiller
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
  - kind: ServiceAccount
    name: tiller
    namespace: kube-system
```

- **my-cluster-europe-west1-b** Cluster

Get Credentials

```
gcloud container clusters get-credentials my-cluster-europe-west1-b --zone europe-west1-b --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west1-b.
```

Create Service Account

```
$ kubectl create -f rbac-tiller-development.yaml
serviceaccount/tiller created
clusterrolebinding.rbac.authorization.k8s.io/tiller created
```

Initialize Helm

```
$ helm init --service-account tiller
$HELM_HOME has been configured at /Users/username/.helm.
Tiller (the Helm server-side component) has been installed into your Kubernetes Cluster.

Please note: by default, Tiller is deployed with an insecure 'allow unauthenticated users' policy.
To prevent this, run `helm init` with the --tiller-tls-verify flag.
For more information on securing your installation see: https://docs.helm.sh/using_helm/#securing-your-helm-installation
```

- **my-cluster-europe-west3-a** Cluster

Get Credentials

```
$ gcloud container clusters get-credentials my-cluster-europe-west3-a --zone europe-west3-a --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west3-a.
```

Create Service Account

```
$ kubectl create -f rbac-tiller-development.yaml
serviceaccount/tiller created
clusterrolebinding.rbac.authorization.k8s.io/tiller created
```

Initialize Helm

```
$ helm init --service-account tiller
```

Referenced Links:

- Installing Tiller official Steps
- Couchbase Helm Setup

- **Add the chart repository to helm**

```
helm repo add couchbase https://couchbase-partners.github.io/helm-charts/
```

4.3. Deploy Couchbase Operator

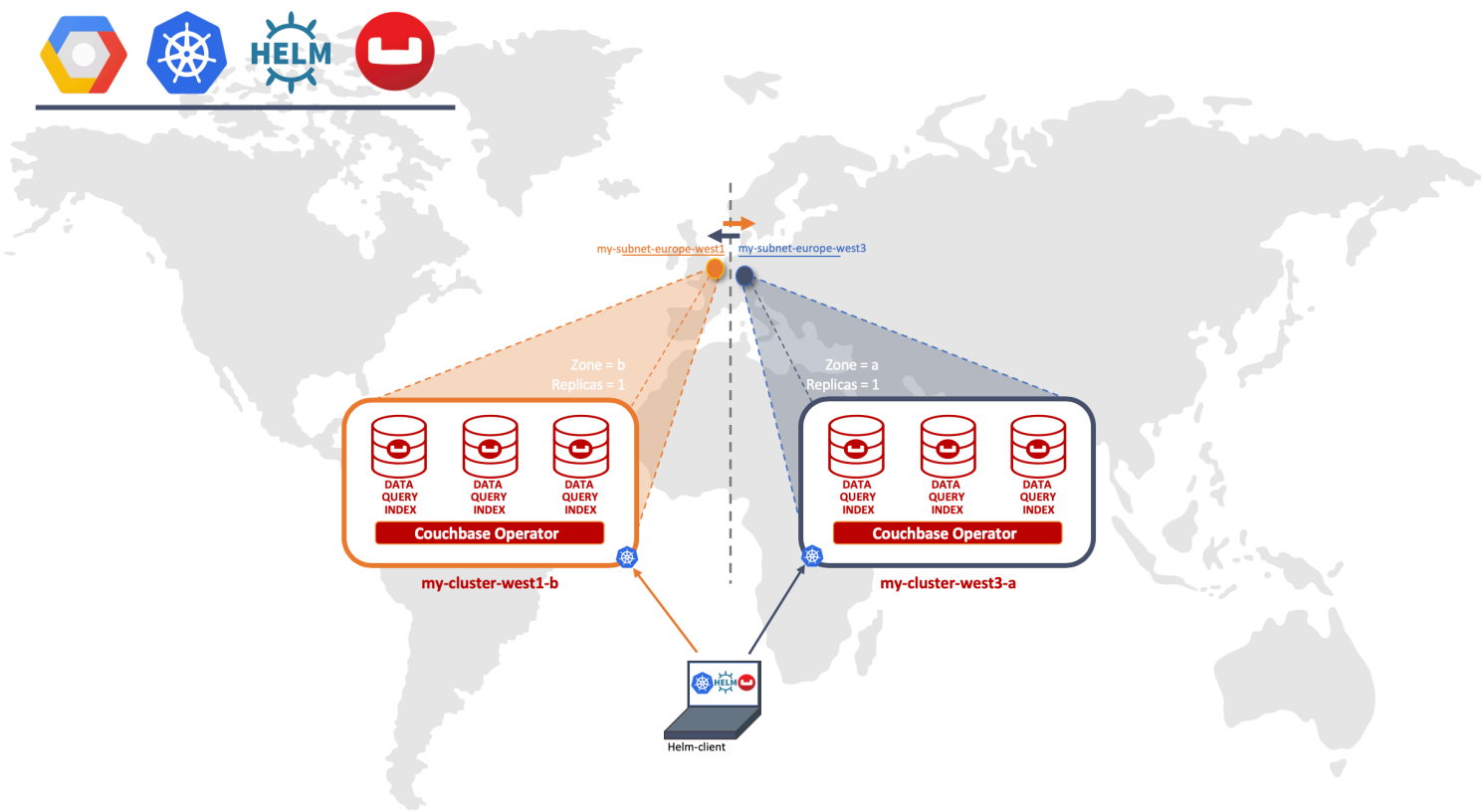
Two Helm charts are available for deploying Couchbase. The Couchbase Operator Chart deploys the admission controller and the Operator itself. The Couchbase Cluster Chart deploys the Couchbase Server cluster.

For production deployments, you'll only use the Operator Chart. For development environments, the Couchbase Cluster Chart is available to help you quickly set up a test cluster.

1. Add the chart repository to helm: `helm repo add couchbase https://couchbase-partners.github.io/helm-charts/`
2. Install the operator chart

```
$ helm install couchbase/couchbase-operator
```

4.4. Deploy Couchbase Server Cluster



On this document, we will modify the default helm chart to config the cluster to deploy into myvalues-clustercfg.yaml file.

```
couchbaseCluster:
  version: "enterprise-6.0.1"
  cluster:
    indexStorageSetting: plasma
    autoFailoverTimeout: 30
    autoFailoverMaxCount: 3
    autoFailoverOnDataDiskIssues: true
    autoFailoverOnDataDiskIssuesTimePeriod: 30
  buckets:
    default:
      name: demo
      type: couchbase
      memoryQuota: 128
      replicas: 1
      ioPriority: high
      evictionPolicy: valueOnly
      conflictResolution: seqno
      enableFlush: true
      enableIndexReplica: false
  servers:
    all_services:
      size: 3
      services:
        - data
        - index
        - query
```

This file will override the default bucket name (demo), its evictionPolicy (valueOnly) and the default services (data, index, query)

Let's deploy it with helm into each cluster:

- **my-cluster-europe-west1-b** Cluster

Get Credentials

```
$ gcloud container clusters get-credentials my-cluster-europe-west1-b --zone europe-west1-b --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west1-b.
```

helm install

```
$ helm install --values myvalues-clustercfg.yaml --set couchbaseCluster.name=my-cluster-west1b couchbase/couchbase-cluster
```

- **my-cluster-europe-west3-a** Cluster

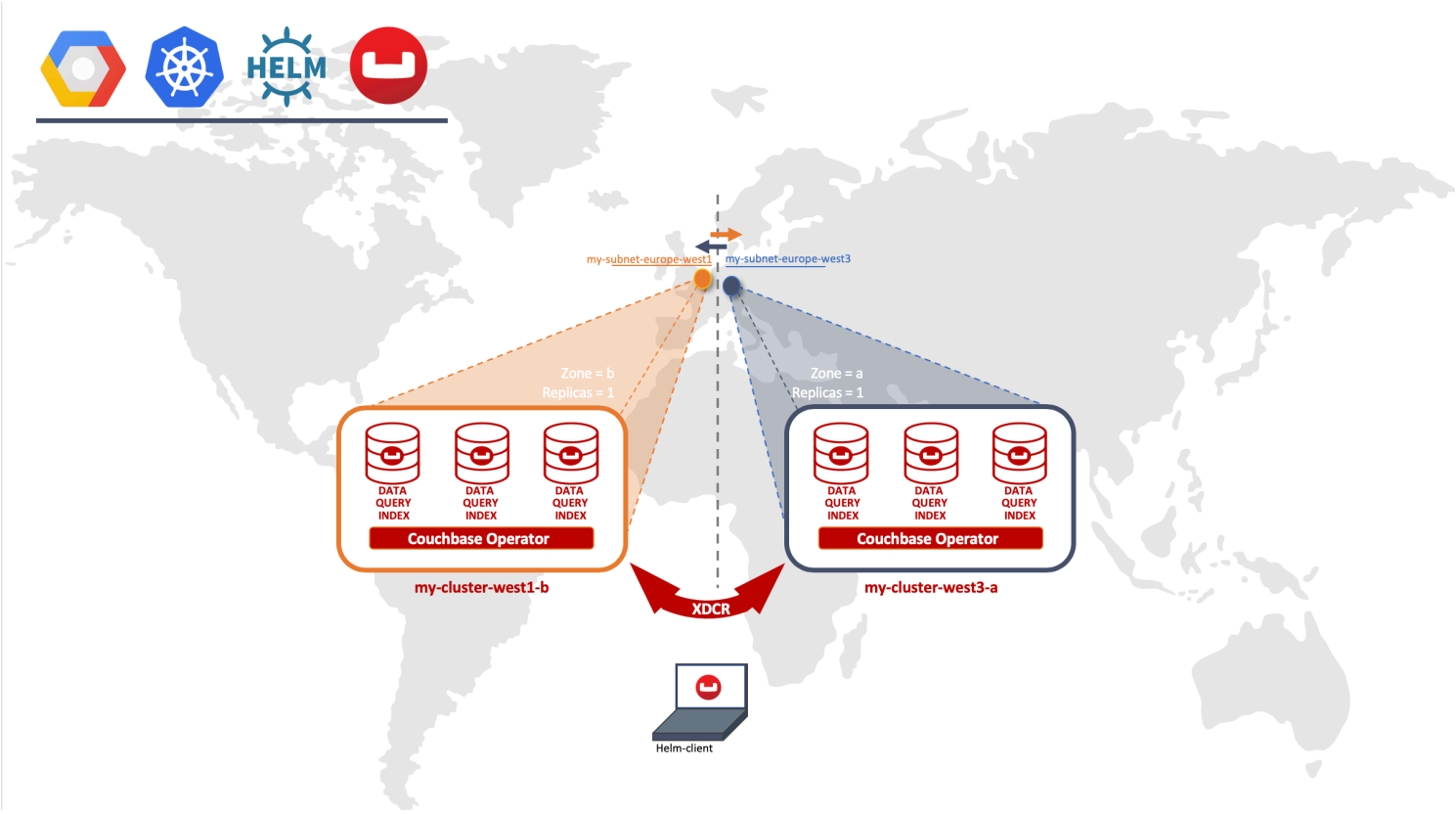
Get Credentials

```
$ gcloud container clusters get-credentials my-cluster-europe-west3-a --zone europe-west3-a --project my-couchbase-helm
Fetching cluster endpoint and auth data.
kubeconfig entry generated for my-cluster-europe-west3-a.
```

helm install

```
$ helm install --values myvalues-clustercfg.yaml --set couchbaseCluster.name=my-cluster-west3a couchbase/couchbase-cluster
```

5. Config XDCR



5.1. Determine the correct connection string on the XDCR target cluster

1. List all Couchbase pods

```
$ kubectl get pods
NAME                                                      READY   STATUS    RESTARTS   AGE
my-cluster-west1b-0000                                    1/1     Running   0          4m13s
my-cluster-west1b-0001                                    1/1     Running   0          3m26s
my-cluster-west1b-0002                                    1/1     Running   0          2m43s
plundering-dachshund-couchbase-admission-controller-6d4b8484tx2   1/1     Running   0          4m34s
plundering-dachshund-couchbase-operator-7fb96d5944-cx5kb          1/1     Running   0          4m34s
```

1. Choose one of the Couchbase pods and get its underlying node's IP address:
   `$ kubectl get pod my-cluster-west1b-0000 -o yaml | grep hostIP hostIP: 10.0.0.4`
2. Get the port number that maps to the admin port (8091)
   `$ kubectl get service my-cluster-west1b-0000-exposed-ports NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE my-cluster-west1b-0000-expose`

If you were logged into the Couchbase Server Web Console on Cluster 1, and establishing the XDCR connection to Cluster 2, you'd use the connection string **10.0.0.4:30995** based on the example above.

```
$ kubectl get pods
NAME                                                    READY   STATUS    RESTARTS   AGE
dunking-magpie-couchbase-admission-controller-755998bc87-7kt8n   1/1     Running   0          3m32s
dunking-magpie-couchbase-operator-5f74df58fd-dg9xn                1/1     Running   0          3m32s
my-cluster-west3a-0000                                  1/1     Running   0          3m12s
my-cluster-west3a-0001                                  1/1     Running   0          2m4s
my-cluster-west3a-0002                                  1/1     Running   0          79s
```

```
$ kubectl get pod my-cluster-west3a-0000 -o yaml | grep hostIP
  hostIP: 10.16.0.2
```

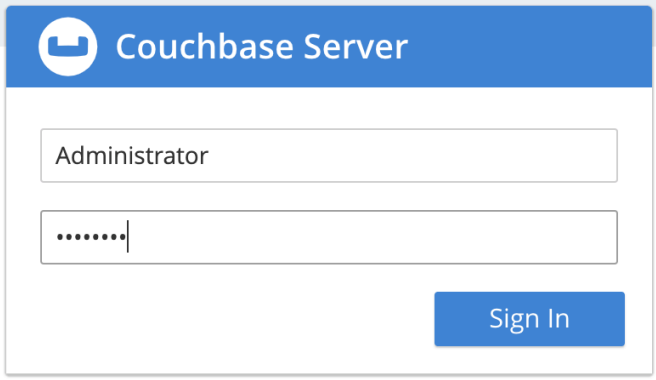
```
$ kubectl get service my-cluster-west3a-0000-exposed-ports
NAME                                TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)
                                AGE
my-cluster-west3a-0000-exposed-ports   NodePort   10.63.242.103   <none>        8091:32179/TCP,18091:30171/TCP,8092:31741/TCP,18092:31097/T
CP,11210:30879/TCP,11207:30569/TCP   3m4s
```

5.2. Accessing the Couchbase Web UI

To use the web console we will need setup port forwarding. We do that with the kubectl command:

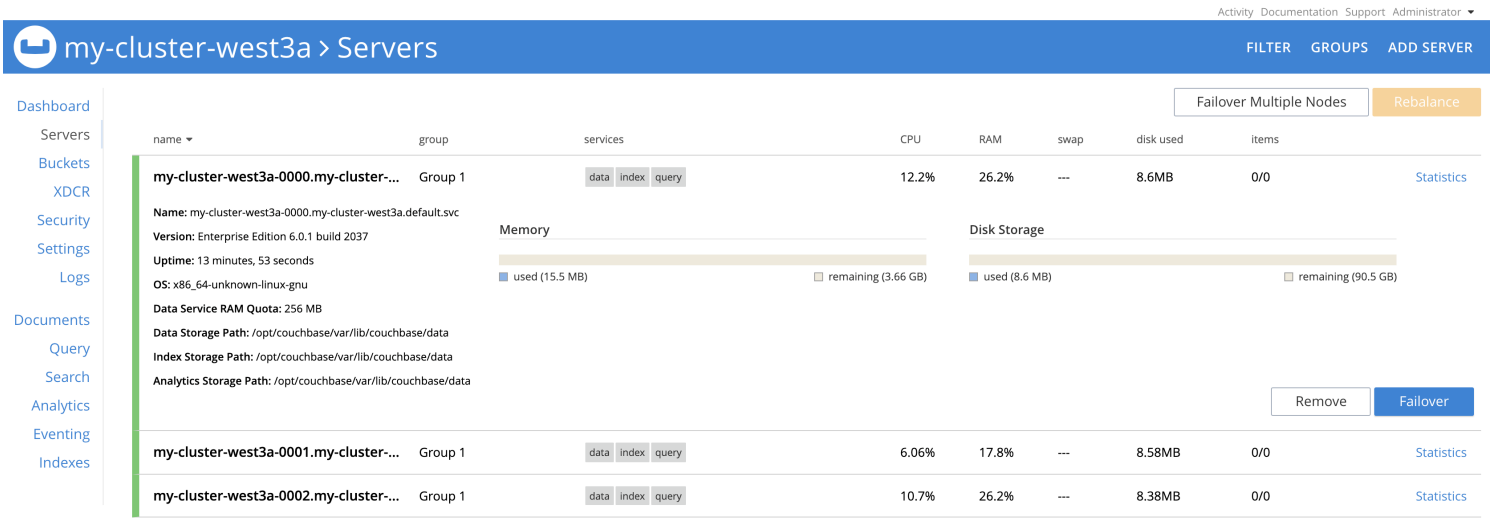
```
kubectl port-forward my-cluster-west3a-0000 8091:8091
```

We need to make sure we leave the command running in the terminal. Now we can open up a browser at http://localhost:8091



We will login using `username=Administrator` and `password=password`.

We are in! Click the 'Servers' link on the left side and we should see our clusters running.



Open a new terminal, **get the credentials for my-cluster-west1b**, and setup port forwarding for my-cluster-west1b cluster on port **8092**.

```
kubectl port-forward my-cluster-west1b-0000 8092:8091
```

Now we can open up a browser at http://localhost:8092

5.3. Create XDCR replication between clusters

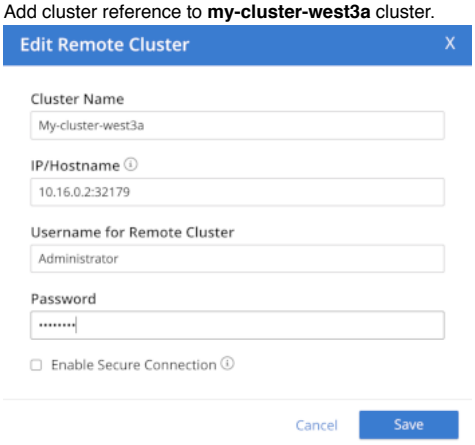
Let's configure a bi-directional replication between both clusters. You can find more details in the XDCR documentation.

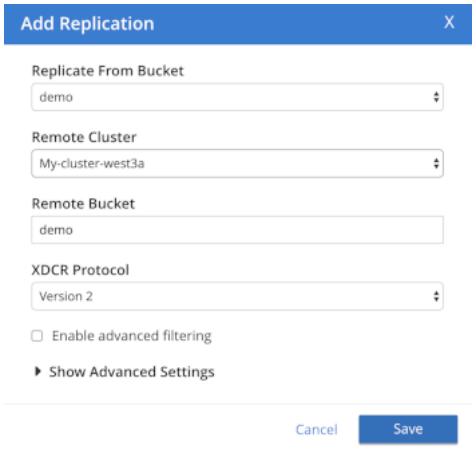
First, let's get the connetion string on west1b and west3a clusters

- Cluster 1: **10.0.0.4:30995**
- Cluster 2: **10.16.0.2:32179**

In **my-cluster-west1b** cluster http://localhost:8092:

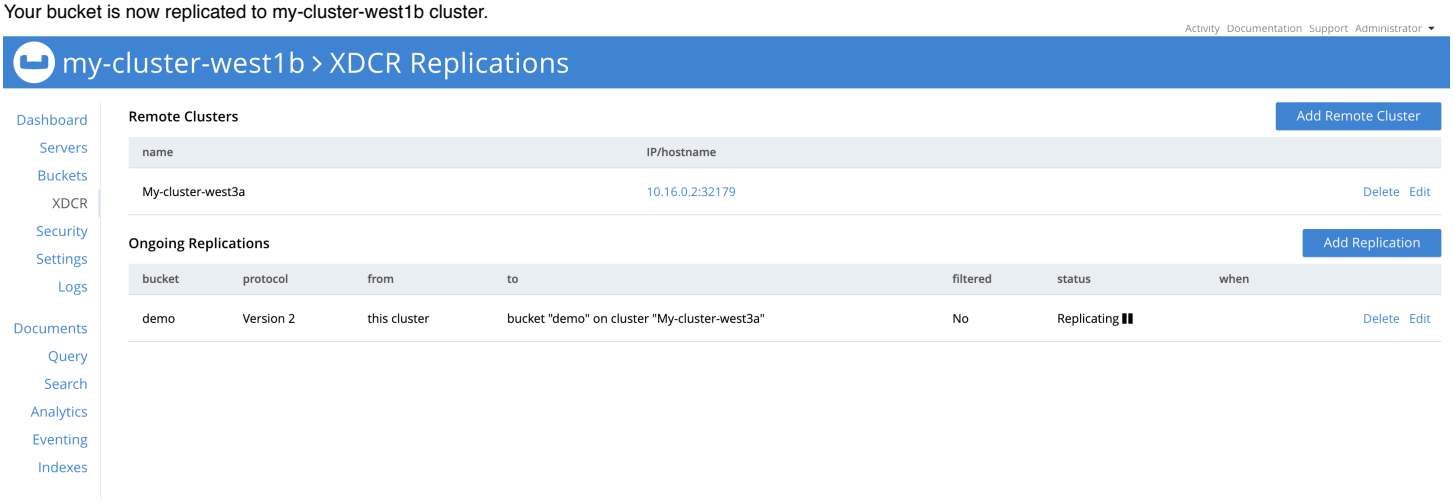
1. Add cluster reference to **my-cluster-west3a** cluster.





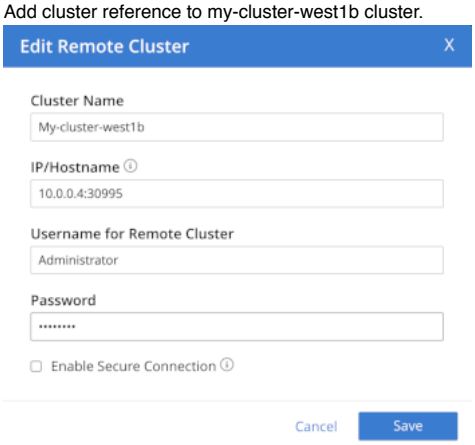
2. Add bucket replication from **my-cluster-west3b.demo** bucket to **my-cluster-west1b.demo** bucket.

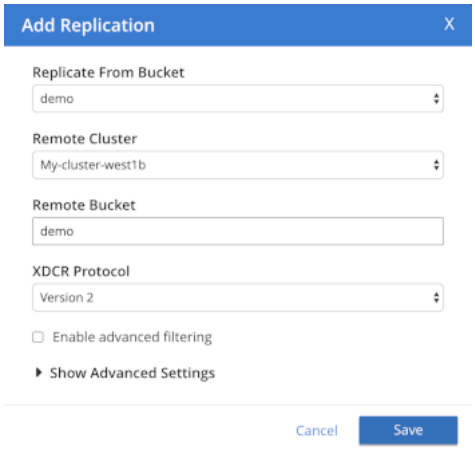
3. Your bucket is now replicated to my-cluster-west1b cluster.



In my-cluster-west3a cluster http://localhost:8091:

1. Add cluster reference to my-cluster-west1b cluster.





2. Add bucket replication from my-cluster-west3a.demo bucket to my-cluster-west1b.demo bucket.
3. Your bucket is now replicated bi-directional in both clusters.

6. References

- https://docs.couchbase.com/operator/current/install-gke.html
- https://docs.couchbase.com/operator/current/helm-setup-guide.html
- https://kubedex.com/beginners-guide-to-helm/
- Setup XDCR
- https://docs.couchbase.com/server/6.0/manage/manage-xdcr/xdcr-management-overview.html