Kubernetes Source to Prod

SHARED PUBLICLY

We will be following the codelab posted here with a couple of changes for the sake of making setup easier. The goal of the codelab is to familiarize you with the Spinnaker deployment process with respect to Kubernetes, as well as how some of the Spinnaker concepts translate to Kubernetes concepts.

The following instructions take care of the "setup" portion of the above codelab by making sure all dependencies are created, provisioning a GKE cluster with 4 nodes, and installing Spinnaker on it. To avoid dealing with configuring authentication, we simply tunnel into the provisioned clusters network, and access Spinnaker directly.

Note: If you get stuck, check the Troubleshooting section below. If it doesn't help, feel free to call us over!

Dependency Setup

1. In the API Manager, make sure your project has the following APIs enabled:
    a. Compute Engine API
    b. Google Cloud Storage *Should be auto-enabled with compute engine*
    c. Google Container Engine API
    d. Google Cloud Resource Manager API
    e. Google Identity and Access Management (IAM) API
2. Create (or use an existing) GitHub account. *Your GitHub account will host the code for the service we will build into a container, and deploy to Kubernetes.*
    a. You can either fork my sample code here, or create your own.
3. Create (or use an existing) DockerHub account. *We're using DockerHub because it's easy to setup build triggers, and free for hosting public images.*
    a. Once your account is ready, create a repository. The codelab uses the name "spin-kub-demo". We recommend picking this name to match the instructions.
    b. Finally, follow this guide to setup automatic build triggers between the docker and git repositories you've created.
4. Make sure gcloud is installed, authenticated, and pointing at your project

```
gcloud auth login

gcloud config set project $PROJECTID
```

5. There will be a step in the code lab that asks you to create a static IP for a load balancer in your cluster. Let's do this now, and record the IP you've allocated for later use:

```
gcloud compute addresses create for-spin-demo \
    --region=us-central1
```

Spinnaker Setup

Next we need to provision a cluster, and install Spinnaker on it. Since we are doing this in a fixed, known environment, we've written a script to simplify the installation process.

1. Prepare your dev environment. This is only needed to get Spinnaker and Kubernetes up and running. There are two ways do this:
    a. Install git and kubectl (`gcloud components install kubectl` or follow this) on your local dev machine.
    b. Create a VM with the necessary dependencies and IAM scopes: *Only run these commands if you want to bootstrap Spinnaker and Kubernetes from a VM rather than your local machine!*

```
gcloud iam service-accounts create \
    spinnaker-bootstrap-account \
    --display-name spinnaker-bootstrap-account

SA_EMAIL=$(gcloud iam service-accounts list \
    --filter="displayName:spinnaker-bootstrap-account" \
    --format='value(email)')

PROJECT=$(gcloud info --format='value(config.project)')

gcloud projects add-iam-policy-binding $PROJECT \
    --role roles/owner --member serviceAccount:$SA_EMAIL

gcloud compute instances create bootstrap \
    --image spinnaker-kubernetes-base \
    --image-project marketplace-spinnaker-release \
    --scopes $SA_EMAIL=cloud-platform

gcloud compute ssh bootstrap \
    --ssh-flag "-L 9000:localhost:9000" \
    --ssh-flag "-L 8084:localhost:8084"
```

2. Run the following commands in your terminal (either the VM you just ssh'd into, or your local dev environment):

```
git clone https://github.com/lwander/spinnaker

cd spinnaker

git checkout codelab-prestartup

cd experimental/kubernetes/simple/

bash scripts/prestartup.sh # this creates a cluster,
                           # and generates the necessary
                           # config with a few questions
                           # and gcloud commands.

bash scripts/startup-all.sh

kubectl get pods --namespace spinnaker --watch
                           # once all pods are “READY”,
                           # hit ctrl-C and continue.
                           #
                           # This shouldn’t take more than 5
                           # minutes.

bash scripts/connect.sh deck 9000 &

bash scripts/connect.sh gate 8084
```

At this point, navigate to localhost:9000 and start the codelab here.

Troubleshooting

1. `scripts/prestartup.sh` failed.
   a. Check the Container Engine page to make sure the spinnaker-host cluster was created.
   b. Check the contents of config/clouddriver-local.yml to see if your username/password were correctly copied.

    c. Check the contents of config/spinnaker-local.yml under the front50 heading to make sure your project and bucket names are correct.
    d. Check https://console.cloud.google.com/iam-admin/iam/project to make sure that the spinnaker-gcs-account was created and as associated with your project.
2. Some containers are never "READY".
    a. First try solutions 1.b, 1.c, and 1.d.
    b. If they are "RUNNING", take the component name (i.e. clouddriver), and run `bash scripts/logs.sh clouddriver` to see if anything suspicious shows up.
    c. If they keep restarting, try the same as 2.b
    d. If they are stuck in "CONTAINERCREATING", record the name of the pod, and run `kubectl describe pod $PODNAME --namespace spinnaker` and look for anything suspicous.
3. The UI isn't showing up.
    a. If the container is "READY", run `bash scripts/logs.sh deck` and check for any obvious error messages. If there are none, wait for another minute or so and try reloading the UI - sometimes the container reports as ready briefly before the UI is reachable.

Helpful Commands

```
scripts/update-config.sh
```

Makes all local config in ./config available to the spinnaker components. They will have to be restarted to access the new config.

```
scripts/update-component.sh $COMPONENTNAME # e.g. clouddriver
```

Restarts a specific spinnaker component - useful after having updated config for a particular component.

```
scripts/cleanup-all.sh
```

Deletes everything in the spinnaker namespace. Will poll for about a minute until everything is gone.

```
scripts/logs.sh $COMPONENTNAME # e.g. clouddriver
```

Reads logs from the component.

Cleanup

Once you want to delete your created resources completely, run the following

```
gcloud container clusters delete spinnaker-host –-zone us-central1-f

gcloud iam service-accounts delete spinnaker-gcr-account@$(gcloud config list \
    --format=’value(config.project)’).iam.gserviceaccount.com

# if you ran some commands from a VM...

gcloud compute instances delete bootstrap –-zone us-central1-f

gcloud iam service-accounts delete spinnaker-bootstrap-account@$(gcloud config list \
    --format=’value(config.project)’).iam.gserviceaccount.com
```