Autoscaling Deployments with External Metrics

This tutorial demonstrates how to automatically scale your GKE workloads based on metrics available in Stackdriver (https://cloud.google.com/stackdriver/).

If you want to autoscale based on metric exported by your Kubernetes workload or a metric attached to Kubernetes object such as Pod or Node visit Autoscaling Deployments with Custom Metrics (https://cloud.google.com/kubernetes-engine/docs/tutorials/custom-metrics-autoscaling) instead.

This example shows autoscaling based on number of undelivered messages in a Cloud Pub/Sub (https://cloud.google.com/pubsub/) subscription, but the instructions can be applied to any metric available in Stackdriver.

Objectives

This tutorial covers the following steps:

1. How to deploy custom metrics Stackdriver adapter.
2. How to deploy HorizontalPodAutoscaler (HPA) resource to scale your Deployment based on Stackdriver metric from another Google Cloud Platform service.

Before you begin

Take the following steps to enable the Kubernetes Engine API:

1. Visit the Kubernetes Engine page (https://console.cloud.google.com/projectselector/kubernetes) in the Google Cloud Console.
2. Create or select a project.
3. Wait for the API and related services to be enabled. This can take several minutes.

4. Make sure that billing is enabled for your Google Cloud project. Learn how to confirm billing is enabled for your project (https://cloud.google.com/billing/docs/how-to/modify-project).

Install the following command-line tools used in this tutorial:

- `gcloud` is used to create and delete Kubernetes Engine clusters. `gcloud` is included in the Google Cloud SDK (https://cloud.google.com/sdk/docs/quickstarts).
- `kubectl` is used to manage Kubernetes, the cluster orchestration system used by Kubernetes Engine. You can install `kubectl` using `gcloud`:

```
gcloud components install kubectl
```

Set defaults for the `gcloud` command-line tool

To save time typing your project ID (https://support.google.com/cloud/answer/6158840) and Compute Engine zone (https://cloud.google.com/compute/docs/zones#available) options in the `gcloud` command-line tool, you can set the defaults:

```
gcloud config set project [PROJECT_ID]
gcloud config set compute/zone [COMPUTE_ENGINE_ZONE]
```

Create cluster

Create a GKE cluster (https://cloud.google.com/kubernetes-engine/docs/how-to/creating-a-container-cluster) running **Kubernetes version 1.10 or later**.

Deploy Pub/Sub subscriber application

To run application using Cloud Pub/Sub you need to create Google Cloud service account and provide credentials to your application. This is covered in Authenticating to Cloud Platform with Service Accounts (https://cloud.google.com/kubernetes-engine/docs/tutorials/authenticating-to-cloud-platform) tutorial.

The following steps will show how to autoscale the deployment created in Authenticating to Cloud Platform with Service Accounts

(https://cloud.google.com/kubernetes-engine/docs/tutorials/authenticating-to-cloud-platform). You should complete steps 1 to 6 of that tutorial before proceeding.

Step 1: Deploy Custom Metrics Stackdriver Adapter

To grant GKE objects access to metrics stored in Stackdriver, you need to deploy the Custom Metrics Stackdriver Adapter (https://github.com/GoogleCloudPlatform/k8s-stackdriver/tree/master/custom-metrics-stackdriver-adapter). In order to run Custom Metrics Adapter you must grant your user the ability to create required authorization roles by running the following Kubernetes command:

```
kubectl create clusterrolebinding cluster-admin-binding \
    --clusterrole cluster-admin --user "$(gcloud config get-value account)"
```

To deploy the adapter in your cluster, run the following command:

```
kubectl create -f https://raw.githubusercontent.com/GoogleCloudPlatform/k8s-stackdri
```

Step 2: Create HorizontalPodAutoscaler object

Once you have deployed Custom Metrics Stackdriver Adapter, you can deploy a HorizontalPodAutoscaler to autoscale your Deployment.

The following manifest file describes a HorizontalPodAutoscaler object that scales a Deployment based on the target based on number of unacknowledged messages in Pub/Sub subscription.

cloud-pubsub/deployment/pubsub-hpa.yaml (https://github.com/GoogleCloudPlatform/kubernetes-engine-samples/blob/master/cloud-pubsub/deployment/pubsub-hpa.yaml)

:M/KUBERNETES-ENGINE-SAMPLES/BLOB/MASTER/CLOUD-PUBSUB/DEPLOYMENT/PUBSUB-HPA.YAML)

```
apiVersion: autoscaling/v2beta1
kind: HorizontalPodAutoscaler
metadata:
  name: pubsub
```

```
spec:
  minReplicas: 1
  maxReplicas: 5
  metrics:
  - external:
      metricName: pubsub.googleapis.com|subscription|num_undelivered_messages
      metricSelector:
        matchLabels:
          resource.labels.subscription_id: echo-read
      targetAverageValue: "2"
    type: External
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: pubsub
```

To deploy this manifest, download it to your machine as hpa.yaml, and run:

```
kubectl apply -f hpa.yaml
```

Step 3: Generate load

The sample application deployed in this example processes a single message every few seconds. By continuously publishing you can cause messages to queue up, which will trigger autoscaling. You can use the following command to publish 200 messages to the Pub/Sub topic:

```
for i in {1..200}; do gcloud pubsub topics publish echo --message="Autoscaling #${i}
```

Step 4: Observe HPA creating additional replicas of your application

You can check the current number of replicas of your deployment by running:

```
kubectl get deployment pubsub
```

You can also inspect the state and recent activity of HPA by running:

```
kubectl describe hpa pubsub
```

Output:

```
Name:
...
Metrics:
"pubsub.googleapis.com|subscription|num_undelivered_messages" (target average value)
Min replicas:
Max replicas:
Conditions:
Type            Status  Reason            Message
----            ------  ------            -------
AbleToScale     True    SucceededRescale  the HPA controller was able to update the
ScalingActive   True    ValidMetricFound  the HPA was able to successfully calculate
ScalingLimited  True    TooManyReplicas   the desired replica count is more than the
Events:
Type    Reason             Age   From                       Message
----    ------             ----  ----                       -------
Normal  SuccessfulRescale  7s    horizontal-pod-autoscaler  New size: 4; reason:
external metric
pubsub.googleapis.com|subscription|num_undelivered_messages(&LabelSelector{MatchLabe
echo-read,},MatchExpressions:[],}) above target
```

The Metrics section gives the last value of metric observed by HPA. Fractional values are represented as milli-units. For example, in the output above there are 4 replicas of application and the current number of unacknowledged messages in Pub/Sub subscription is 9. So the average number of messages per replica is 2.25, or 2250m.

The Conditions section describes whether or not the HorizontalPodAutoscaler is able to scale. In the example above ScalingLimited condition shows that the number of replicas is at maximum allowed value and HPA will not increase it any further.

Finally, the HPA creates an event (see "Events" section in the output above) every time it changes the number of replicas of your application. In the example above, the number of replicas was changed to 4, because the number of unacknowledged messages was above target.

Note: External metrics are printed correctly in kubectl 1.10 and later. Metric value will not be listed as unknown when using older kubectl version.

Using a different metric

Note: Stackdriver Monitoring in the Cloud Console is now Generally Available and the default experience. For a limited period of time, you also have the option to use the classic Stackdriver Monitoring console. For more information, see Stackdriver Monitoring in the Cloud Console (https://cloud.google.com/monitoring/docs/monitoring_in_console).

Any metric available in Stackdriver can be used for autoscaling, as long as `metricType` is `INT64` or `DOUBLE`. To browse available metrics you can use Metrics Explorer or GCP Metrics List (https://cloud.google.com/monitoring/api/metrics_gcp). For metrics of kind `GAUGE` the current metric value is used for autoscaling. For `DELTA` and `CUMULATIVE` metrics the *rate* of metric - the metric change per second - is used instead. You can find more details on how the rate is calculated in Custom Metrics Adapter documentation (https://github.com/GoogleCloudPlatform/k8s-stackdriver/tree/master/custom-metrics-stackdriver-adapter#metrics-available-from-stackdriver)
.

To view the metrics for a monitored resource using Metrics Explorer, do the following:

1. In the Google Cloud Console, go to **Monitoring** or use the following button:
   GO TO MONITORING (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/MONITORING)
2. If **Metrics Explorer** is shown in the navigation pane, click **Metrics Explorer**. Otherwise, select **Resources** and then select **Metrics Explorer**.

Writing external metric specification

Once you choose a metric to autoscale on you need to write external metric specification for HorizontalPodAutoscaler. To do so you need to specify three fields:

- **metricName** is the name of the metric. Kubernetes API doesn't allow using slashes in metric names and you must replace them with pipe symbols (`|`). Note how `pubsub.googleapis.com/subscription/num_undelivered_messages` was specified as `pubsub.googleapis.com|subscription|num_undelivered_messages` in this tutorial.
- **metricSelector** can be optionally used to select a specific time series of the metric. If metricSelector matches multiple time series the sum of their values is used for autoscaling. In this tutorial metricSelector was used to limit HPA to only take into account messages in `echo-read` subscription. Learn more about Stackdriver Time Series and

Labels (https://cloud.google.com/monitoring/api/ref_v3/rest/v3/TimeSeries) and Kubernetes LabelSelector (https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/).

- **targetAverageValue** specifies how much of a total value of metric each replica can handle. This is useful when using metrics that describe some work or resource that can be divided between replicas, for example Pub/Sub messages or QPS. For other metrics, such as average request latency, it makes more sense to specify a global target value. You can do that by specifying **targetValue** instead of targetAverageValue.

Autoscaling based on multiple metrics

You can use multiple metrics with a single HorizontalPodAutoscaler, combining External metrics with other metric types described in [Autoscaling Deployments with Custom Metrics]. To do so specify each metric you want to use as a separate entry in `metrics` list in your HPA object specification. HPA will calculate the number of replicas based on each metric and pick the highest one.

Cleaning up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:

1. Clean up the Pub/Sub subscription and topic:

```
gcloud pubsub subscriptions delete echo-read
gcloud pubsub topics delete echo
```

2. Delete your GKE cluster by running the following command:

```
gcloud container clusters delete [CLUSTER_NAME]
```

What's next

- Learn more about custom and external metrics for scaling workloads (https://cloud.google.com/kubernetes-engine/docs/concepts/custom-and-external-metrics)

- Explore other Kubernetes Engine tutorials (https://cloud.google.com/kubernetes-engine/docs/tutorials).
- Try out other Google Cloud features for yourself. Have a look at our tutorials (https://cloud.google.com/docs/tutorials).

Last updated December 18, 2019.