Combine GCP StackDriver with a GKE Cluster

This lesson focuses on combining GCP StackDriver with a GKE cluster.

WE'LL COVER THE FOLLOWING

- GKE's Fluentd DaemonSet
- Output the logs of Fluentd containers
  - Wait until the action propagates

GKE's Fluentd DaemonSet

If you're using the GKE cluster, logging is already set up, even though you might not know about it. By default, every GKE cluster comes with a **Fluentd DaemonSet** that is configured to forward logs to **GCP StackDriver**. It is running in the `kube-system` Namespace.

Let's describe **GKE's Fluentd DaemonSet** and see whether there is any useful information we might find.

```
kubectl -n kube-system \
  describe ds -l k8s-app=fluentd-gcp
```

The **output**, limited to the relevant parts, is as follows.

```
...
Pod Template:
  Labels:       k8s-app=fluentd-gcp
                kubernetes.io/cluster-service=true
                version=v3.1.0
...
  Containers:
   fluentd-gcp:
    Image: gcr.io/stackdriver-agents/stackdriver-logging-agent:0.3-1.5.34-
1-k8s-1
    ...
```

We can see that, among others, the DaemonSet's Pod Template has the label `k8s-app=fluentd-gcp`. We'll need it soon. Also, we can see that one of the containers is based on the `stackdriver-logging-agent` image. Just as **Papertrail** extended **Fluentd**, Google did the same.

Output the logs of Fluentd containers

Now that we know that Stackdriver-specific **Fluentd** is running in our cluster as a *DaemonSet*, the logical conclusion would be that there is already a UI we can use to explore the logs. UI is indeed available but, before we see it in action, we'll output the logs of the **Fluentd** containers and verify that everything is working as expected.

```
kubectl -n kube-system \
  logs -l k8s-app=fluentd-gcp \
  -c fluentd-gcp
```

Unless you already enabled *Stackdriver Logging API*, the output should contain at least one message similar to the one that follows.

```
...
18-12-12 21:36:41 +0000 [warn]: Dropping 1 log message(s) error="7:Stackdr
iver Logging API has not been used in project 152824630010 before or it i
s disabled. Enable it by visiting https://console.developers.google.com/ap
is/api/logging.googleapis.com/overview?project=152824630010 then retry. I
f you enabled this API recently, wait a few minutes for the action to prop
agate to our systems and retry." error_code="7"
```

Fortunately, the warning already tells us not only what the issue is, but also what to do. Open the link from the log entry in your favorite browser, and click the *ENABLE* button.

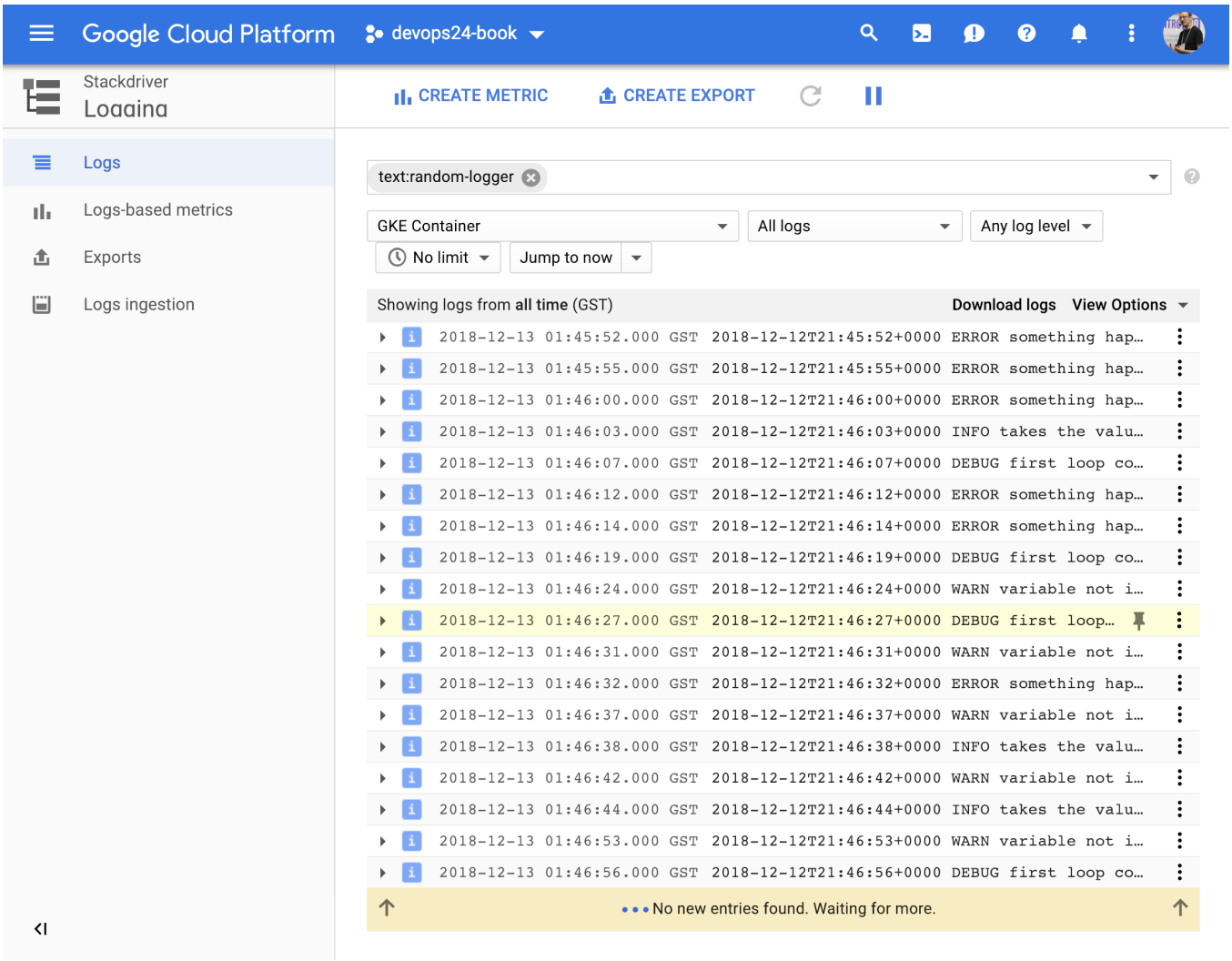
Wait until the action propagates

Now that we enabled *Stackdriver Logging API*, **Fluentd** will be able to ship log entries there. All we have to do is wait for a minute or two until the action propagates.

Let's see the Stackdriver UI.

```
open "https://console.cloud.google.com/logs/viewer"
```

Please type *random-logger* in the *Filter by label or text search* field and select *GKE Container* from the drop-down list. The **output** should display all the logs that contain *random-logger* text.



GCP StackDriver logs screen

Q

By default, every GKE cluster comes with a **Fluentd DaemonSet** that is configured to forward logs to **GCP StackDriver**.

We won't go into details about how to use **Stackdriver**. It is easy and, hopefully, intuitive. So, I'll leave it to you to explore it in more detail. What matters is that it is very similar to what we experienced with **Papertrail**. Most of the differences are cosmetic.

If you are using GCP, **Stackdriver** is ready and waiting for you. As such, it probably makes sense to use it over any other third-party solution. **Stackdriver** contains not only the logs coming from the cluster but also logs of all GCP services (e.g., load balancers). That is probably a significant difference between the two solutions. It is a massive bonus in favor of **Stackdriver**. Still, check the pricing before making a decision.

In the next lesson, we will combine **AWS CloudWatch** with an EKS cluster.