Explore Centralized Logging Through Papertrail

In this lesson, we will explore centralized logging through Papertrail.

WE'LL COVER THE FOLLOWING

Papertrail

The first centralized logging solution we'll explore is Papertrail. We'll use it as a representative of a logging-as-a-service solution that can save us from installing and, more importantly, maintaining a self-hosted alternative.

Papertrail features live trailing, filtering by timestamps, powerful search queries, pretty colors, and quite a few other things that might (or might not) be essential when skimming through logs produced inside our clusters.

Register at PaperTrail

The first thing we need to do is to register or, if this is not the first time you tried **Papertrail**, log in.

```
open "https://papertrailapp.com/"
```

Please follow the instructions to register, or log in if you already have a user in their system.

You will be glad to find out that **Papertrail** provides a free plan that allows storage of 50 MB of logs searchable for one day, as well as a full year of downloadable archives. That should be more than enough for running the examples we are about to explore. If you have a relatively small cluster, that should keep you going indefinitely. Their prices are reasonable, even if your cluster is bigger and you have more monthly logs than 50 MB. Arguably, they are so cheap that we can say it provides a better return on investment than if we'd run an alternative solution inside our own cluster. After all, nothing is free. Even self-hosted solutions based on open source create costs in maintenance time as well as in computing power.

For now, what matters is that the examples we'll run with **Papertrail** will be well within their free plan.

If you have a small operation, **Papertrail** will work well. But, if you have many applications and a bigger cluster, you might be wondering whether **Papertrail** scales to suit your needs. Worry not. One of their customers is *GitHub*, and they are likely bigger than you are. **Papertrail** can handle (almost) any load. However, whether it is a good solution for you is yet to be discovered. Read on.

Start using PaperTrail

Let's go to the Start screen unless you are already there.

```
open "https://papertrailapp.com/start"
```

If you were redirected to the welcome screen, you are not authenticated (your session might have expired). Login and repeat the previous command to get to the start screen.

Click the *Add systems* button.

If you read the instructions, you'll probably think that the setup is relatively easy. It is. However, Kubernetes is not available as one of the options. If you change the value of the *from* drop-down list to *something else*..., you'll see a fairly big list of log sources that can be plugged into **Papertrail**. Still, there is no sign of Kubernetes. The closest one on that list is *Docker*. Even that one will not do. Don't worry. I prepared instructions for you or, to be more precise, I extracted them from the documentation buried in **Papertrail's** site.

Please note the *Your logs will go to logsN.papertrailapp.com:NNNNN and appear in Events* message at the top of the screen. We'll need that address soon, so we better store the values in environment variables.

```
PT_HOST=[...]

PT_PORT=[...]
```

Please replace the first `[...]` with the host. It should be something like `logsN.papertrailapp.com`, where `N` is the number assigned to you by **Papertrail**. The second `[...]` should be replaced with the port from the before mentioned message.

Now that we have the host and the port stored in environment variables, we can explore the mechanism we'll use to collect and ship the logs to **Papertrail**.

Mechanism to collect and ship the logs

Since I already claimed that most vendors adopted **Fluentd** for collecting and shipping logs to their solutions, it should come as no surprise that **Papertrail** recommends it as well. Folks from *SolarWinds* (**Papertrail's** parent company) created an image with customized **Fluentd** that we can use. In turn, I created a YAML file with all the resources we'll need to run their image.

```
cat logging/fluentd-papertrail.yml
```

As you can see, the YAML defines a DaemonSet with ServiceAccount, *SolarWind's* **Fluentd**, and a ConfigMap that uses a few environment variables to specify the host and the port where logs should be shipped.

We'll have to change the `logsN.papertrailapp.com` and `NNNNN` entries in that YAML before we apply it. Also, I prefer running all logs-related resources in `logging` Namespace, so we'll need to change that as well.

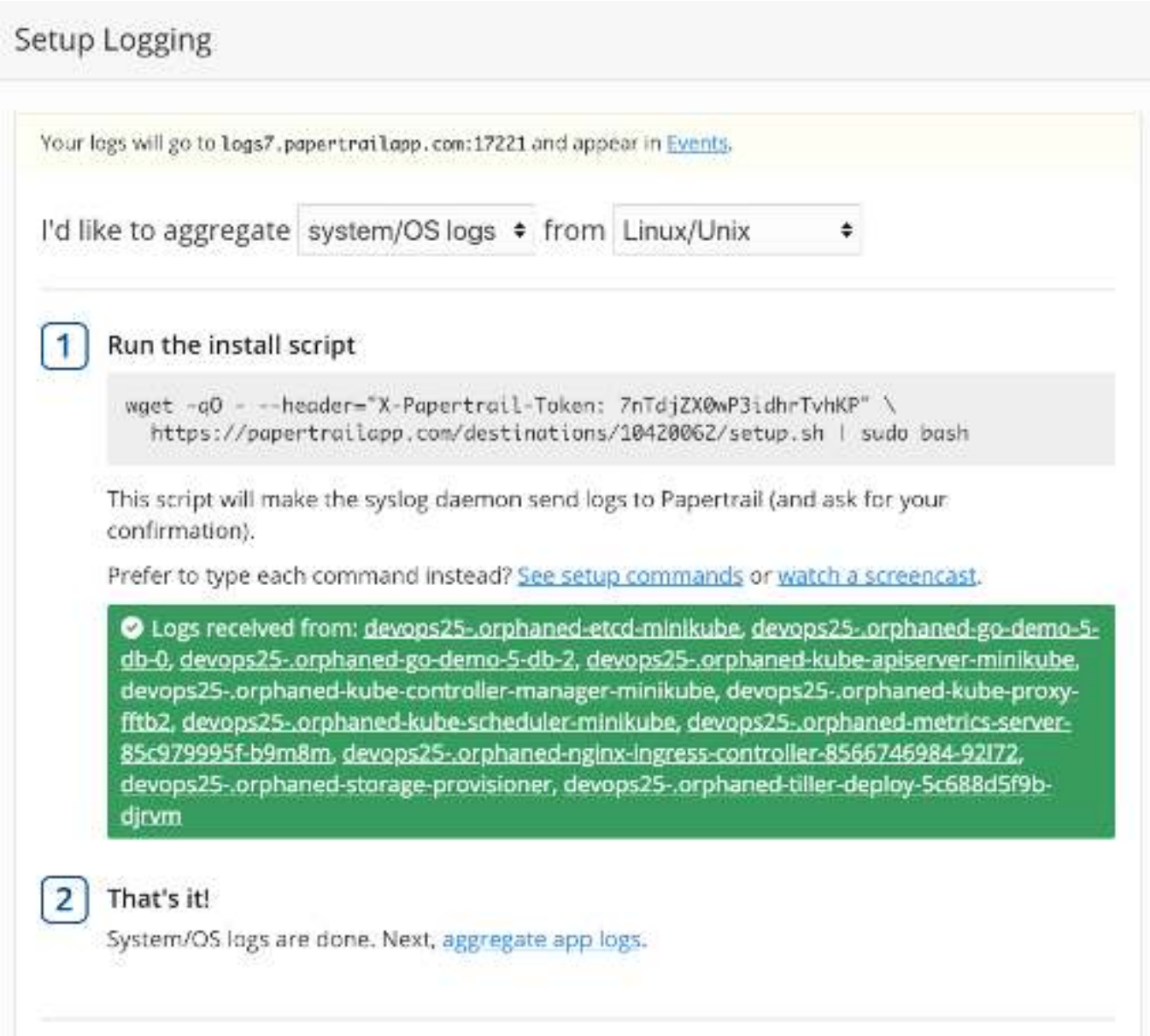
```
cat logging/fluentd-papertrail.yml \
    | sed -e \
    "s@logsN.papertrailapp.com@$PT_HOST@g" \
    | sed -e \
    "s@NNNNN@$PT_PORT@g" \
```

```
    | kubectl apply -f - --record


kubectl -n logging \
  rollout status ds fluentd-papertrail
```

Now that we're running **Fluentd** in our cluster and that it is configured to forward logs to our **Papertrail** account, we should turn back to its UI.

Please switch back to **Papertrail** console in your browser. You should see a green box stating that logs were received. Click the *Events* link.



Papertrail's Setup Logging screen

Q

Folks from *SolarWinds* (**Papertrail's** parent company) created an image with customized **Fluentd** that we can use.

COMPLETED 0%

1 of 1

Explore logs in Papertrail

Next, we'll produce a few logs and explore how they appear in Papertrail.

```
cat logging/logger.yml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: random-logger
spec:
  containers:
  - name: random-logger
    image: chentex/random-logger
```

That Pod uses `chentex/random-logger` image which has a single purpose. It periodically outputs random log entries.

Let's create `random-logger`.

```
kubectl create -f logging/logger.yml
```

Please wait for a minute or two to accumulate a few logs entries.

```
kubectl logs random-logger
```

The **output** should be similar to the one that follows.

```
...
2018-12-06T17:21:15+0000 ERROR something happened in this execution.
2018-12-06T17:21:20+0000 DEBUG first loop completed.
2018-12-06T17:21:24+0000 ERROR something happened in this execution.
2018-12-06T17:21:27+0000 ERROR something happened in this execution.
2018-12-06T17:21:29+0000 WARN variable not in use.
```

```
2018-12-06T17:21:31+0000 ERROR something happened in this execution.
2018-12-06T17:21:33+0000 DEBUG first loop completed.

2018-12-06T17:21:35+0000 WARN variable not in use.
2018-12-06T17:21:40+0000 WARN variable not in use.
2018-12-06T17:21:43+0000 INFO takes the value and converts it to string.
2018-12-06T17:21:44+0000 INFO takes the value and converts it to string.
2018-12-06T17:21:47+0000 DEBUG first loop completed.
```

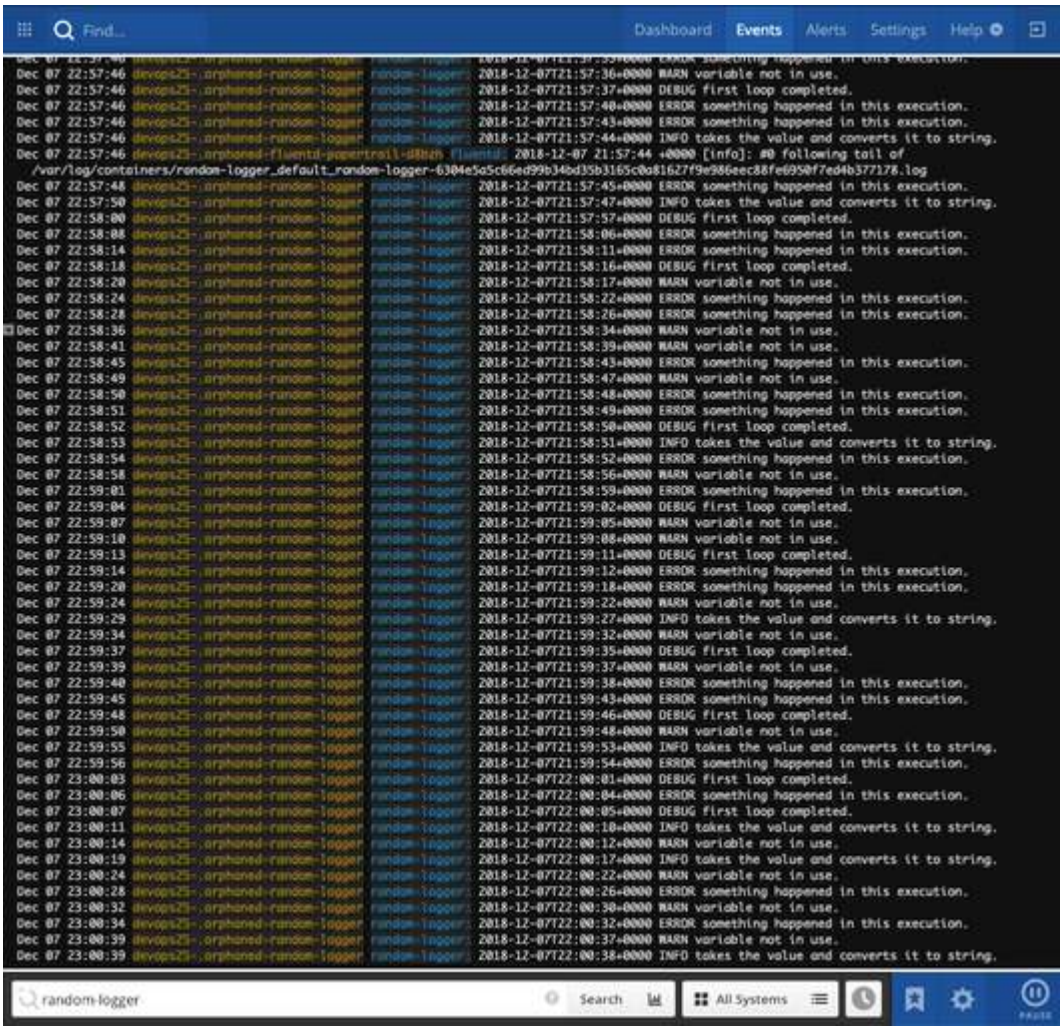
As you can see, the container is outputting random entries, some of them as `ERROR`, and others as `DEBUG`, `WARN`, and `INFO`. Messages are random as well. After all, that is not a real application, but a simple image that produces log entries we can use to explore our logging solution.

Please go back to **Papertrail** UI.

You should notice that all the logs from our system are available. Some are coming from Kubernetes, while others are from system-level services. Those from `go-demo-5` are also there, together with the `random-logger` we just installed. We'll focus on the latter.

Let's imagine that we found out through alerts that there is an issue and that we limited the scope to the `random-logger` application. Alerts helped us detect the problem and we narrowed it down to a single application by digging through metrics. We still need to consult logs to find the cause. Given what we know (or invented), the logical next step would be to retrieve only the log entries related to the `random-logger`.

Please type *random-logger* in the *Search* field at the bottom of the screen, and press the enter key.

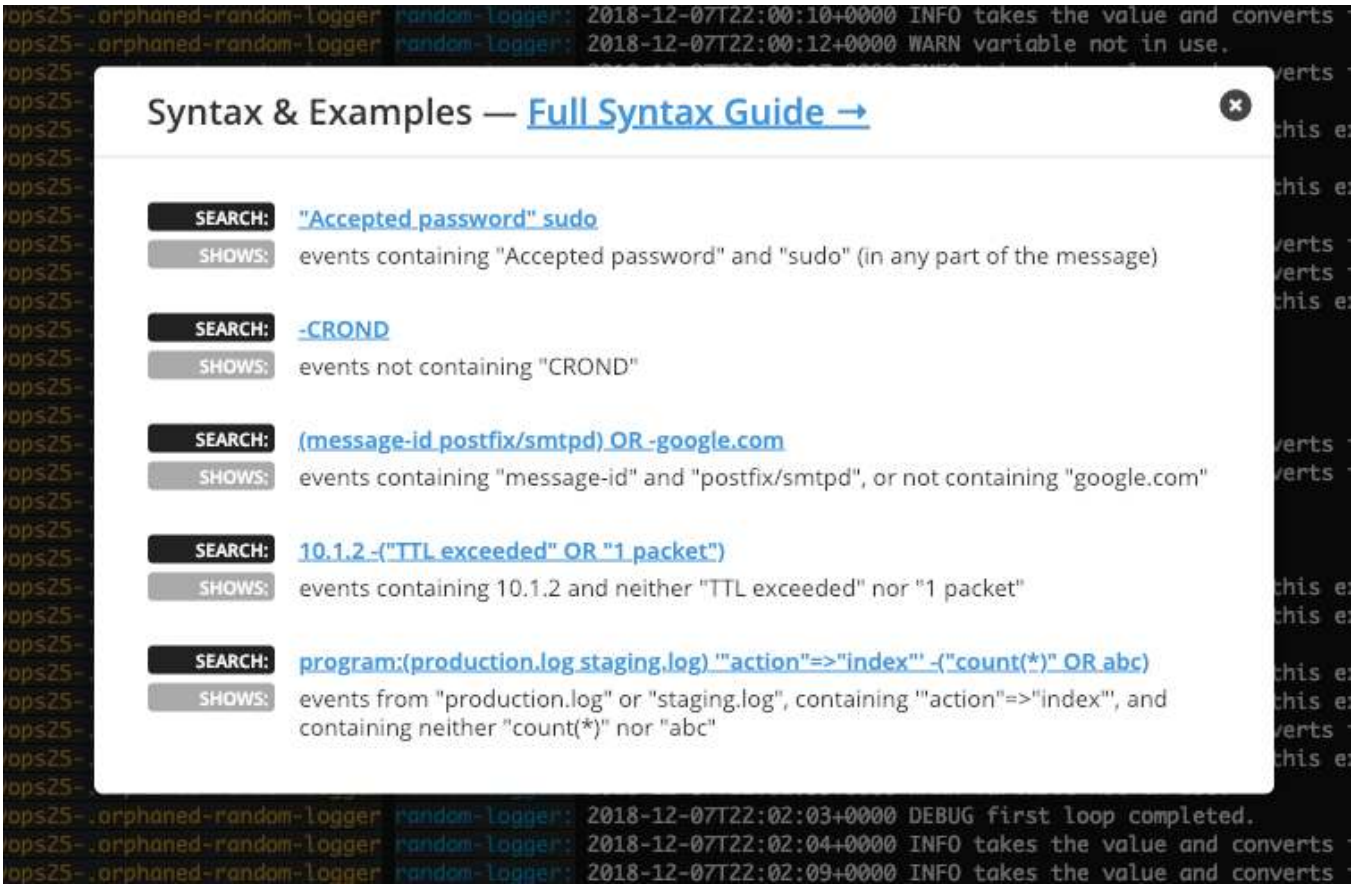


Papertrail's Events screen

Log containing the word `random-logger`

From now on, we'll see only log entries that contain the word `random-logger`. That does not necessarily mean that only the log entries from that application are displayed. Instead, any mention of that word is shown on the screen. What we did was to instruct **Papertrail** to perform a free-text search inside all the log entries and retrieve only those that contain the beforementioned word.

While free-text search across all the records is probably the most commonly used query, there are a few other ways we could filter logs. We won't go through all of them. Instead, click the *Search tips* button in the right-hand side of the *Search* field and explore the syntax yourself. If those few examples are not enough, click the *Full Syntax Guide* link.



Papertrail's Syntax & Examples screen

Remove DaemonSet and ConfigMap

There's probably no need to explore **Papertrail** in more detail. It is intuitive, easy to use, and well-documented service. I'm sure you'll figure out the details if you choose to use it. For now, we'll remove the DaemonSet and the ConfigMap before we move into exploring alternatives.

```
kubectl delete \
  -f logging/fluentd-papertrail.yml
```

Next, we'll explore logging solutions available in Cloud providers. Feel free to jump directly to Combine GCP StackDriver With A GKE Cluster, Combine AWS CloudWatch with an EKS Cluster, or Combine Azure Log Analytics with an AKS Cluster. If you do not use any of the three providers, you can skip them altogether and go directly to the Explore Centralized Logging sub-chapter.