Application Logging

Fluentd is an open source data collector, which lets you unify data collection and consumption for a better use and understanding of data generated on your Oak device.
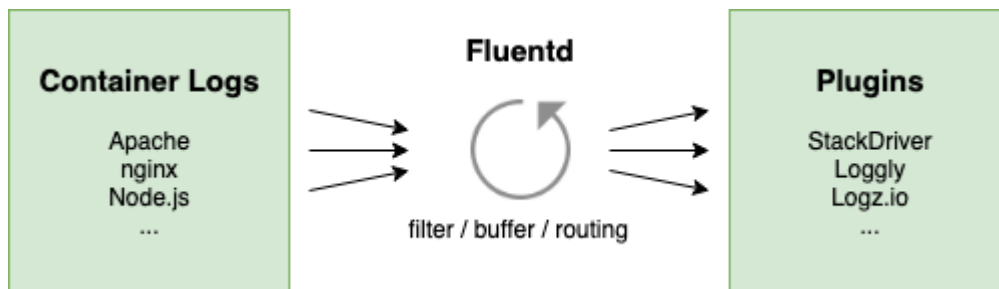

Figure 1.1 - Your application container logs are forwarded to `platform-fluentd`

Logging with Fluentd

Fluentd structures data into JSON, allowing for filtering, buffering and routing as seen in Figure 1.1.

Fluentd utilizes a unified logging layer to allow for easier processing and analysis. There are countless plugins available to handle your logs.

In order to utilize a logging provider, you must install component-fluentd (GitHub) :fa-external-link-square: onto your Oak Device.

The master branch of component-fluentd uses Google Cloud Plugin (Tutorial). Some providers require mounting or providing credentials in your Docker image for `component-fluentd`. If that is the case, as it is with Google Cloud Plugin, you will need to clone the GitHub repository for component-fluentd to mount your secrets and then create a private Docker container for you to install on your Oak devices.

This same component-fluentd has two other branches for the providers: Logz.io (Plugin Tutorial) and Loggly (Plugin Tutorial).
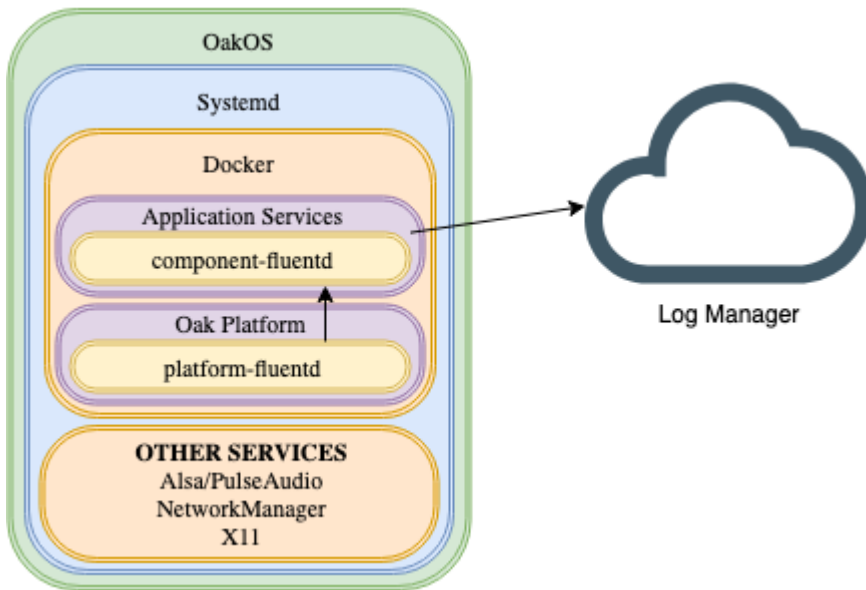
Figure 1.2 - `platform-fluentd` captures all logs and `component-fluentd` sends them to your provider."

OakOS uses a system component called `platform-fluentd` (Figure 1.2) that provides a stream for logs. In order to send logs to a provider as shown in Figure 1.1, you will need to install a plugin using `component-fluentd`. The plugin is rather simple, the primary facets of this repository are the Dockerfile and the fluentd.conf configuration.

```
FROM fluent/fluentd:stable
USER root
ENTRYPOINT []

RUN apk add --no-cache --virtual .build-deps build-base ruby-dev \\
    && gem install fluent-plugin-google-cloud \\
    && apk del .build-deps

ENV GOOGLE_APPLICATION_CREDENTIALS=/gcp-credentials.json

CMD [\"fluentd\", \"-p\", \"/fluentd/plugins\", \"-c\", \"/fluentd.conf\"]
```

In the Dockerfile, you may specify any environment variables your provider requires such as GOOGLE_APPLICATION_CREDENTIALS.

The fluentd.conf (Stackdriver for Google Cloud Plugin) configuration provides the options for your plugin.

```
# Accept logs from OakOS forwarding
<source>
  @type forward
  port 24224
</source>

# Filter out FluentD logs
<match fluent.**>
  @type null
</match>

# Dump logs to stdout (do not use in production)
# <match **>
#   @type stdout
# </match>
```

```
# Send logs to Stackdriver (Google Cloud Platform)
<match **>
  @type google_cloud
  use_metadata_service false
  use_grpc true
  # This is used by Stackdriver to group logs.
  # It is recommended that the same value is used on all units for an org.
  zone \"#{ENV['GOOGLE_ZONE']}\"
  vm_id \"#{Socket.gethostname}\"
  # Set the chunk limit conservatively to avoid exceeding the limit
  # of 10MB per write request.
  buffer_chunk_limit 2M
  flush_interval 15s
  retry_wait 30s
  max_retry_wait 4h
  disable_retry_limit
  num_threads 4
</match>
```

This fluentd.conf file specifies how you want to handle your logs. This file will be customized for each provider. See more examples in our other plugin tutorials: Logz.io Plugin and Loggly Plugin.

Fluentd Components for your platform application"

Our Docker Hub has three different logging provider component plugins, using Fluentd, for your use:

GitHub Component	Provider	Howto
component-fluentd:logzio	Logz.io	Logz.io Plugin
component-fluentd:loggly	loggly	Loggly Plugin
component-fluentd:master	Google Stackdriver	Google Cloud Plugin