



<https://verifone.cloud/docs/oakos/writing-application/application-logging/loggly-plugin>

Updated: 23-Oct-2020

Loggly Plugin

We will set up the [example application](#), using [Fluentd with Loggly](#), for handling logs.

WARNING You will need an account at [Loggly](#) to continue.

Setting up Loggly

For testing this tutorial, if you do not already have an account, you may create a free trial at Loggly.

Once you have created your account, you must get an API token to set in the environment variable:
LOGGLY_TOKEN

Go to the Loggly token endpoint for your organization:

- `https://{your-organization}.loggly.com/account/users/api/tokens`

The screenshot shows the Loggly web interface. At the top is a dark navigation bar with the 'solarwinds loggly' logo and several icons. Below this is a light-colored header with tabs: 'Overview', 'Users', 'API Tokens' (which is highlighted), 'Archiving', 'CloudTrail', 'Subscription', and 'User Settings'. A light blue informational banner states: 'Please note: This API token is used to retrieve your logs from Loggly. The Customer Token required to your logs to Loggly is available [here](#).' Below the banner, the 'API Tokens' section is shown with an 'Add New' button. Underneath, it says 'Active (1)'. A table lists one active API token with the label 'API Token' and a value '12345678-abcd-1234-efgh-098765432100'.

API Token
12345678-abcd-1234-efgh-098765432100

Figure 1.1 - API Tokens List at Loggly.

To create a token, click the Add New button (see Figure 1.1). Once created, the token will appear in the active tokens list, such as the example above. That value will need to be set, in the next step, in a Postman environment variable: `{{logglyToken}}`

Install the example application and configure Fluentd"

Oak Platform (API): Install

```
{
  "services": [ {
    "image": "index.docker.io/oaklabs/app-example:release-1.0.1",
    "environment": {
      "TZ": "America/Phoenix"
    }
  },
  {
    "image": "index.docker.io/oaklabs/component-fluentd:loggly",
    "environment": {
      "LOGGLY_TAG": "{{logglyTag}}",
      "LOGGLY_TOKEN": "{{logglyToken}}"
    }
  }
]
```

Viewing the logs

The logs being sent to Loggly are the same that were sent in the [Logz.io Plugin](#) tutorial.

The screenshot displays the Loggly web interface. At the top, the browser address bar shows the URL `https://(your-organization).loggly.com/search#terms=REMOTE_URL&from=2019-08-08T23:00`. The navigation bar includes the 'solarwinds loggly' logo and links for 'Search', 'Charts', 'Dashboard', 'Alerts', 'Derived Fields', and 'Source Setup'. Below the navigation bar, there's a search bar with 'REMOTE_URL' entered. A filter 'tag : my-loggly-tag' is applied. The 'Field Explorer' on the left shows 'logtype' and 'tag' fields. The 'Event Timeline' on the right shows 2 events. The 'Event View' on the right shows the expanded log entry for the first event, including fields like 'hostname', 'message', 'pid', and 'service'.

Figure 1.2 - Expanded log entry for REMOTE_URL at Loggly. This screenshot shows an expanded example log message with the REMOTE_URL environment variable being logged in the example application.

The search results are filtering by the example tag that was set in `{{logglyTag}}: my-loggly-tag`