Okta Source

The Okta Source provides a secure endpoint to receive event data from the Okta System Log API. It securely stores the required authentication, scheduling, and state tracking information.

**Okta API rate limits**

During a polling interval, an Okta Source will make a request for every 1,000 logs available. The Okta API uses paging and only 1,000 logs are returned at a time.

**States**

An Okta Source tracks errors, reports its health, and start-up progress. You’re informed, in real-time, if the Source is having trouble connecting, if there’s an error requiring user action, or if it is healthy and collecting by utilizing Health Events.

An Okta Source goes through the following states when created:

1. **Pending**: Once the Source is submitted it is validated, stored, and placed in a Pending state.
2. **Started**: A collection task is created on the Hosted Collector.
3. **Initialized**: The task configuration is complete in Sumo Logic.
4. **Authenticated**: The Source successfully authenticated with Okta.
5. **Collecting**: The Source is actively collecting data from Okta.

If the Source has any issues during any one of these states it is placed in an Error state.

When you delete the Source it is placed in a Stopping state, when it has successfully stopped it is deleted from your Hosted Collector.

On the Collection page, the Health and Status for Sources is displayed. Use Health Events to investigate issues with collection.

Hover your mouse over the status icon to view a tooltip with details on the detected issue.
Generate the Okta API token

Create an Okta API token, following instructions in Okta help. You will add the token to the Okta source, later in this procedure.

Create an Okta Source

When you create an Okta Source, you add it to a Hosted Collector. Before creating the Source, identify the Hosted Collector you want to use or create a new Hosted Collector. For instructions, see Configure a Hosted Collector.

To configure an Okta Source:

1. In the Sumo Logic web app, select Manage Data > Collection > Collection.
2. On the Collectors page, click Add Source next to a Hosted Collector.
4. Enter a Name to display for the Source in the Sumo web application. The description is optional.
5. (Optional) For Source Category, enter any string to tag the output collected from the Source. Category metadata is stored in a searchable field called _sourceCategory.

6. Forward to SIEM. Check the checkbox to forward your data to Cloud SIEM Enterprise. When configured with the Forward to SIEM option the following metadata fields are set:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>_siemVendor</td>
<td>Okta</td>
</tr>
<tr>
<td>_siemProduct</td>
<td>SSO</td>
</tr>
</tbody>
</table>
Field Name          Value

_siemFormat          JSON

_siemEventID         <eventType>

Where <eventType> is the value of the field from the JSON event, such as user.session.start. Possible event types.

7. Fields. Click the +Add link to add custom log metadata Fields.
   - Define the fields you want to associate, each field needs a name (key) and value.
     - A green circle with a check mark is shown when the field exists and is enabled in the Fields table schema.
     - An orange triangle with an exclamation point is shown when the field doesn't exist, or is disabled, in the Fields table schema. In this case, an option to automatically add or enable the nonexistent fields to the Fields table schema is provided. If a field is sent to Sumo that does not exist in the Fields schema or is disabled it is ignored, known as dropped.

8. Okta API Key. Provide the Okta API key you want to use to authenticate collection requests.

9. Okta Domain. Provide your specific Okta domain, such as mydomain.okta.com.

10. Okta Event Types to Request. By default, the Source will ingest all Okta events. You can instead configure a subset of events to collect. Click Select Events to specify the events you want to collect.

11. (Optional) The Polling Interval is set for 300 seconds by default, you can adjust it based on your needs. This sets how often the Source checks for new data.

12. When you are finished configuring the Source click Submit.

Error types
When Sumo Logic detects an issue it is tracked by Health Events. The following table shows the three possible error types, the reason the error would occur, if the Source attempts to retry, and the name of the event log in the Health Event Index.
<table>
<thead>
<tr>
<th>Type</th>
<th>Reason</th>
<th>Retries</th>
<th>Retry Behavior</th>
<th>Health Event Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ThirdPartyConfig</td>
<td>Normally due to an invalid configuration. You'll need to review your Source configuration and make an update.</td>
<td>No retries are attempted until the Source is updated.</td>
<td>Not applicable</td>
<td>ThirdPartyConfigError</td>
</tr>
<tr>
<td>ThirdPartyGeneric</td>
<td>Normally due to an error communicating with the third party service APIs.</td>
<td>Yes</td>
<td>The Source will retry for up to 90 minutes, after which retries will be attempted every 60 minutes.</td>
<td>ThirdPartyGenericError</td>
</tr>
<tr>
<td>FirstPartyGeneric</td>
<td>Normally due to an error communicating with the internal Sumo Logic APIs.</td>
<td>Yes</td>
<td>The Source will retry for up to 90 minutes, after which retries will be attempted every 60 minutes.</td>
<td>FirstPartyGenericError</td>
</tr>
</tbody>
</table>

**JSON configuration**

Sources can be configured using UTF-8 encoded JSON files with the Collector Management API. See [how to use JSON to configure Sources](#) for details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>JSON Object</td>
<td>Yes</td>
<td>Contains the configuration parameters for the Source.</td>
<td>Access</td>
</tr>
<tr>
<td>schemaRef</td>
<td>JSON Object</td>
<td>Yes</td>
<td>Use {&quot;type&quot;:&quot;Okta&quot;} for an Okta Source.</td>
<td>not modifiable</td>
</tr>
<tr>
<td>sourceType</td>
<td>String</td>
<td>Yes</td>
<td>Use Universal for an Okta Source.</td>
<td>not modifiable</td>
</tr>
</tbody>
</table>

The following table shows the config parameters for an Okta Source.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Required</th>
<th>Default</th>
<th>Description</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Type a desired name of the Source. The name must be unique per Collector. This value is assigned to the metadata field _source.</td>
<td>modificable</td>
</tr>
<tr>
<td>description</td>
<td>String</td>
<td>No</td>
<td>null</td>
<td>Type a description of the Source.</td>
<td>modificable</td>
</tr>
<tr>
<td>category</td>
<td>String</td>
<td>No</td>
<td>null</td>
<td>Type a category of the source. This value is</td>
<td>modificable</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Required</td>
<td>Default</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>---------</td>
<td>----------</td>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>apiKey</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>The Okta API key you want to use to authenticate collection requests.</td>
<td></td>
</tr>
<tr>
<td>domain</td>
<td>String</td>
<td>Yes</td>
<td></td>
<td>Provide your specific Okta domain, such as mydomain.okta.com.</td>
<td></td>
</tr>
<tr>
<td>collectAll</td>
<td>Boolean</td>
<td>No</td>
<td>True</td>
<td>By default, the Source will ingest all Okta events. If false, <code>eventTypes</code> is required.</td>
<td></td>
</tr>
<tr>
<td>eventTypes</td>
<td>String</td>
<td>No</td>
<td></td>
<td>Comma separated list of events to collect. Required if <code>collectAll</code> is false.</td>
<td></td>
</tr>
<tr>
<td>pollingInterval</td>
<td>Integer</td>
<td>No</td>
<td>300</td>
<td>This sets how often the Source checks for new data.</td>
<td></td>
</tr>
</tbody>
</table>

Okta Source JSON example:

```
{
    "api.version":"v1",
    "source":{
        "schemaRef":{
            "type":"Okta"
        },
        "config":{
            "name":"Okta",
            "description":"East field",
            "domain":"mydomain.okta.com",
            "collectAll":true,
            "apiKey":"******",
            "fields":{
                "_siemForward":false
            }
        }
    }
}
```
"category":"eastTeamF",
"pollingInterval":300
},
"sourceType":"Universal"
}