Splunk

Splunk Component

Available as of Camel 2.13

The Splunk component provides access to Splunk, via the Splunk provided client Rest API, allowing you to publish and search for events in Splunk.

Maven users will need to add the following dependency to their **`pom.xml`** for this component:

```
<dependency>
  <groupId>org.apache.camel</groupId>
  <artifactId>camel-splunk</artifactId>
  <version>${camel-version}</version>
</dependency>
```

URI Format

```
splunk://[endpoint]?[options]
```

Producer Endpoints

Endpoint	Description
`stream`	Streams data to a named index, or the default index if not specified. When using stream mode be aware that Splunk has some internal buffer (about 1MB or so) before events gets to the index. If you need realtime, better use **`submit`** or **`tcp`** mode.
`submit`	Uses Splunk's Rest API to publish events to a named index, or the default if not specified.
`tcp`	Streams data to a TCP port, and requires a open receiver port in Splunk.

When publishing events the message body should contain a **`SplunkEvent`**. See later.

Example

```
from("direct:start")
  .convertBodyTo(SplunkEvent.class)
  .to("splunk://submit?
username=user&password=123&index=myindex&sourceType=someSourceType&source=m
ySource");
```

In this example a converter is required to convert to a **`SplunkEvent`** class.

Consumer Endpoints

Endpoint	Description

`normal`	Performs normal search and requires a search query in the search option.
`savedsearch`	Performs search based on a search query saved in Splunk and requires the name of the query in the **`savedSearch`** option.

Example

```
from("splunk://normal?
delay=5s&username=user&password=123&initEarliestTime=-10s&search=search
index=myindex sourcetype=someSourcetype")
  .to("direct:search-result");
```

`camel-splunk` creates a route exchange per search result with an instance of **`org.apache.camel.component.splunk.event.SplunkEvent`** in the body.

URI Options

Name	Default Value	Context	Description
`connectionTimeout`	`5000`	`Both`	Splunk server connection timeout, in milliseconds.
`count`	`0`	`Consumer`	A number that indicates the maximum number of entities to return. This is not the same as **`maxMessagesPerPoll`** option, which currently is unsupported.
`earliestTime`	`null`	`Consumer`	Earliest time of the search time window.
`eventHost`	`null`	`Producer`	**Camel 2.17:** Override the default Splunk event host field.
`host`	`localhost`	`Both`	Splunk host.
`index`	`null`	`Producer`	Splunk index to write to.
`initEarliestTime`	`null`	`Consumer`	Initial start offset of the first search. Required.
`latestTime`	`null`	`Consumer`	Latest time of the search time window.
`password`	`null`	`Both`	Splunk password.
`port`	`8089`	`Both`	Splunk port.
`raw`	`false`	`Producer`	**Camel 2.16.0** : Governs whether the body should be inserted as raw. If **`true`**, the body will be transformed to a **`java.lang.String`** before it's send to Splunk.
`savedSearch`	`null`	`Consumer`	The name of the query saved in Splunk to run.
`scheme`	`https`	`Both`	Scheme to use. Can be one of: **`http`** or **`https`**.
`search`	`null`	`Consumer`	The Splunk query to run.
`source`	`null`	`Producer`	Splunk source argument.
`sourceType`	`null`	`Producer`	Splunk sourcetype argument.

`sslProtocol`	`TLSv1.2`	`Both`	**Camel 2.16:** The SSL protocol to use. Can be one of: - **`TLSv1.2`** - **`TLSv1.1`** - **`TLSv1`** - **`SSLv3`** **Note**: this option is ignored unless the scheme is: **`https`**.
`streaming`	`false`	`Consumer`	**Camel 2.14.0** : Stream exchanges as they are received from Splunk, rather than returning all of them in one batch. This has the benefit of receiving results faster, as well as requiring less memory as exchanges aren't buffered in the component.
`tcpReceiverPort`	`0`	`Producer`	Splunk TCP receiver port when using TCP producer endpoint.
`username`	`null`	`Both`	Splunk username.
`useSunHttpsHandler`	`false`	`Both`	When **`true`** an instance of **`sun.net.www.protocol.https.Handler`** is used to establish the connection to Splunk. Can be useful when running in application servers to avoid application server HTTPS handling.

Message Body

Splunk operates on data in key/value pairs. The **`SplunkEvent`** class is a placeholder for such data, and should be in the message body for the producer. Likewise it will be returned in the body per search result for the consumer.

From **Camel 2.16.0** you can send raw data to Splunk by setting **`raw=true`** on the producer endpoint. This is useful for e.g., **`json/xml`** and other payloads where Splunk has build in support.

Use Cases

Search Twitter for tweets with music and publish events to Splunk

```
from("twitter://search?
type=polling&keywords=music&delay=10&consumerKey=abc&consumerSecret=def&acc
essToken=hij&accessTokenSecret=xxx")
  .convertBodyTo(SplunkEvent.class)
  .to("splunk://submit?username=foo&password=bar&index=camel-
tweets&sourceType=twitter&source=music-tweets");
```

To convert a Tweet to a **`SplunkEvent`** you could use a converter like:

```
@Converter
public class Tweet2SplunkEvent {
    @Converter
    public static SplunkEvent convertTweet(Status status) {
        SplunkEvent data = new SplunkEvent("twitter-message", null);

        data.addPair("from_user", status.getUser().getScreenName());
        data.addPair("in_reply_to", status.getInReplyToScreenName());
        data.addPair(SplunkEvent.COMMON_START_TIME, status.getCreatedAt());
        data.addPair(SplunkEvent.COMMON_EVENT_ID, status.getId());
        data.addPair("text", status.getText());
        data.addPair("retweet_count", status.getRetweetCount());

        if (status.getPlace() != null) {
            data.addPair("place_country", status.getPlace().getCountry());
            data.addPair("place_name", status.getPlace().getName());
            data.addPair("place_street", status.getPlace().
getStreetAddress());
        }

        if (status.getGeoLocation() != null) {
            data.addPair("geo_latitude", status.getGeoLocation().
getLatitude());
            data.addPair("geo_longitude", status.getGeoLocation().
getLongitude());
        }

        return data;
    }
}
```

Search Splunk for tweets:

```
from("splunk://normal?username=foo&password=bar&initEarliestTime=-
2m&search=search index=camel-tweets sourcetype=twitter")
  .log("${body}");
```

Comments

Splunk comes with a variety of options for leveraging machine generated data with pre-built apps for analyzing and displaying this. For example the JMX app. could be used to publish JMX attributes, e.g., route and JVM metrics to Splunk, and displaying this on a dashboard.

See Also

- Configuring Camel
- Component
- Endpoint
- Getting Started