title: Sematext Rsyslog Integration description: Forward logs to Sematext logging management service with rsyslog. Query and visualise logs, create alerts, narrow down events of interest and locate system or app errors

error_outline

IMPORTANT

Although the latest version of `rsyslog` is `8.x.`, the most popular Linux distributions such as Ubuntu, Debian, RHEL, CentOS, etc. come with older versions of rsyslog (e.g. 5.x and 7.x). To avoid issues with old versions of `rsyslog`, we strongly recommend that you upgrade it to `8.x.` Maintainers of `rsyslog` provide packages and install instructions at http://www.rsyslog.com/downloads/download-other/.

Overview

rsyslog is a modern syslog daemon focused on performance. Logsene gives you lots of ways to forward your logs with rsyslog:

- UDP
- TCP (you can also encrypt logs with TLS)
- RELP (uses application-level acknowledgement for increased reliability over plain TCP)
- HTTP / HTTPS over the Elasticsearch API

You can also send JSON over syslog if you need support for structured data.

There are 3 steps for configuring your rsyslog for Logsene:

1. Configure one or more inputs. For example, configure rsyslog to send local logs, remote logs over TCP and so on
2. Choose a protocol and an authentication method. For UDP, TCP, TLS and RELP you can authorize your public IP. However, **we strongly recommend using the Logsene app's token**, which works with all supported protocols
3. Configure the output. Based on the chosen protocol and authentication method, you'll have to configure the appropriate output plugin to send logs in the desired format

Configuring Inputs

First, configure rsyslog to receive logs that you'd like to send to Logsene. The most common input module is imuxsock, which will take logs from your local /dev/log socket. To start listening on local logs, add this line at the beginning of your **/etc/rsyslog.conf**:

```
$ModLoad imuxsock
```

Tailing Files

You can also have rsyslog tail files, listen for syslog messages over TCP, UDP, RELP, pick up messages from the journal and more.

To tail a file, load the file input module and optionally decide how often to pool for changes. Then, for every file, specify its path and related parameters, like this:

Tailing Files by Polling; Old Config Format

```
# add once
$ModLoad imfile                    # load the file input module
$InputFilePollInterval 1           # how often to check files for changes

# for every file you want to monitor
$InputFileName /var/log/jetty.log  # the file to monitor
$InputFileTag jetty:               # syslog tag attributed to those events. Yes, the trail
$InputFileStateFile jetty          # state file to remember where it left between restarts
$InputFileReadMode 2               # support indented multiline logs (requires rsyslog 5.7
$InputRunFileMonitor               # start monitoring this file
```

If you have issues with `logrotate` or other utilities that truncate or organize the files you monitor, upgrade rsyslog to version 8.1.5 or later and the problems should go away. The file input module gets inotifysupport and you also have a new configuration format at your disposal, which is easier to maintain:

Tailing Files via Inotify; New Config Format

```
# add once
module(load="imfile")

# for every file
input(type="imfile"
  File="/var/log/jetty.log"        # the file to monitor
  Tag="jetty:"                      # syslog tag attributed to those events
  ReadMode="2"                     # support indented multiline logs (requires rsyslog 5.7
)
```

Protocol and Authentication

To forward logs, you can use HTTP/HTTPS and authenticate by using your Logsene application token (recommended!). Alternatively, you can use UDP, TCP/TLS or RELP and authenticate either by using the Logsene application token, or by authorizing your public IP in the Logsene application settings.

HTTP/HTTPS via the Elasticsearch API

The recommended method is to use the Elasticsearch API to send logs over HTTP or HTTPS. This will give you maximum flexibility, reliability and encryption, if you need it. This requires rsyslog 6.4.0 or later, and the installation of the Elasticsearch output module. HTTPS is supported in rsyslog 8.2.0 or later (see info about rsyslog update above). To enable the Elasticsearch output module, install the **rsyslog-elasticsearch** package or use **–enable-elasticsearch** when compiling from sources.

Configuration

Before forwarding logs via the Elasticsearch API, define a template in **/etc/rsyslog.conf** that gives structure to your messages by formatting them as JSON:

Configuring Log Template

```
# define a template to specify which fields we send
template(name="LogseneFormat" type="list" option.json="on") {
  constant(value="{")
  constant(value="\"@timestamp\":\"")
  property(name="timereported" dateFormat="rfc3339")
  constant(value="\",\"message\":\"")
  property(name="msg")
  constant(value="\",\"host\":\"")
  property(name="hostname")
  constant(value="\",\"severity\":\"")
  property(name="syslogseverity-text")
  constant(value="\",\"facility\":\"")
  property(name="syslogfacility-text")
  constant(value="\",\"syslog-tag\":\"")
  property(name="syslogtag")
  constant(value="\",\"source\":\"")
  property(name="programname")
  constant(value="\"}")
}
```

UDP, TCP, TLS or RELP

You can send RFC-3164 and RFC-5424 compatible syslog to Logsene via any of the following protocols:

- UDP: fire-and-forget protocol that doesn't guarantee any reliability but is also the fastest and simplest implementation
- TCP: provides better reliability than TCP, though messages might still be lost from the system buffers if the connection breaks and rsyslog is restarted
- TLS: RFC-5425 syslog over TLS is supported
- RELP: this is a reliable protocol, because it supports application-level acknowledgments

Again, with all these protocols, you can either authenticate with your Logsene application token, or by registering your public IP.

Requirements

UDP and TCP support is built into `rsyslog`. For TLS, you need to install the **gtls driver**, typically provided by the **rsyslog-gnutls** package (or by adding –**enable-gnutls** when compiling rsyslog from source). For RELP, you'll need the RELP output module, usually provided by the **rsyslog-relp** package (or by adding –**enable-relp** when compiling from source).

Configuration

If you chose to authorize using static IP address, instead of authenticating using Logsene application token (which is the recommended option), you don't need to make any configuration changes in this step. Instead, go to the Logsene web application and authorize the public IP (or multiple IPs) of the server(s) from where you send your logs.

To use the Logsene application token, you'll first have to obtain it from your list of Logsene applications. Then, in **/etc/rsyslog.conf**, define a template that forwards your messages in CEE-formatted JSON over syslog, where you should put your token in the `logsene-app-token` field:

```
$template LogseneFormat,"<%PRI%>%TIMEREPORTED:::date-rfc3339% %HOSTNAME% %syslogtag%@cee: {\
```

If you are using rsyslog version 7 or later, you can use the new configuration format to define the template. It's more verbose, but easier to maintain (e.g. add new fields, reformat messages):

Configuring Log Template

```
 template(
  name="LogseneFormat"
  type="list"
) {
  # standard syslog fields
  constant(value="<")
    property(name="pri")
  constant(value=">")
  property(name="timereported" dateFormat="rfc3339")
  constant(value=" ")
    property(name="hostname")
  constant(value=" ")
  property(name="syslogtag")
  # CEE-formatted JSON message over syslog
  constant(value="@cee: {\"logsene-app-token\": \"LOGSENE_APP_TOKEN_GOES_HERE\", \"message\"
  property(name="msg" format="json")                        # original syslog message goes
  constant(value="\"}\n")
}
```

Configuring Outputs

The last step is to configure an output module for sending your logs to Logsene. This depends on your chosen protocol.

HTTP / HTTPS via Omelasticsearch

To send your logs over HTTP, load the Elasticsearch output module and point it to **logsene-receiver.sematext.com** (or **logsene-receiver.sematext.com** is using Sematext Cloud Europe) on port **80**. Make sure you replace LOGSENE_APP_TOKEN_GOES_HERE with your actual token:

Configuring Elasticsearch Output

```
module(load="omelasticsearch")
action(type="omelasticsearch"
    template="LogseneFormat"          # the template that you defined earlier
    searchIndex="LOGSENE_APP_TOKEN_GOES_HERE"
    server="logsene-receiver.sematext.com"
    serverport="80"
    bulkmode="on"
    queue.dequeuebatchsize="100"      # how many messages to send at once
    action.resumeretrycount="-1")     # buffer messages if connection fails
```

If you want to encrypt logs, by sending them over HTTPS, you'll have to change `serverport` to `"443"` and add `usehttps="on"`.

UDP

If you're using Logsene application token for authentication, specify the LogseneFormat template in your *action* line. The host you'll connect to is **logsene-syslog-receiver.sematext.com** or **logsene-syslog-receiver.eu.sematext.com** (if using Sematext Cloud Europe):

```
*.* @logsene-syslog-receiver.sematext.com;LogseneFormat
```

If you've authorized your public IPs, the RFC-5424 predefined template will do:

```
*.* @logsene-syslog-receiver.sematext.com;RSYSLOG_SyslogProtocol23Format
```

TCP

With TCP, the *action* line looks similar to UDP, except you'll have two @ signs.

If you're authorizing using Logsene application token:

```
*.* @@logsene-syslog-receiver.sematext.com;LogseneFormat
```

If you use IP address for authorization:

```
*.* @@logsene-syslog-receiver.sematext.com;RSYSLOG_SyslogProtocol23Format
```

TLS

To set up syslog over TLS, you need to configure the TLS driver like this:

Configure TLS; Old Config Format

```
$DefaultNetstreamDriver gtls
$DefaultNetstreamDriverCAFile /etc/ssl/certs/ca-certificates.crt

$ActionSendStreamDriverAuthMode x509/name
$ActionSendStreamDriverPermittedPeer *.sematext.com
$ActionSendStreamDriverMode 1
```

Finally, add the *rsyslog action*, which is the same as the TCP one, except you'll use **port 10514**.

Again, there are two options, for authorizing with the Logsene application token:

```
*.* @@logsene-syslog-receiver.sematext.com:10514;LogseneFormat
```

And for authorizing using the IP address(es):

```
*.* @@logsene-syslog-receiver.sematext.com:10514;RSYSLOG_SyslogProtocol23Format
```

If you prefer the new configuration format, you can find the complete TLS configuration below:

Configure TLS; New Config Format

```
 global (
 defaultNetstreamDriver="gtls"
 defaultNetstreamDriverCAFile="/etc/ssl/certs/ca-certificates.crt"
)

action(
  type="omfwd"
  target="logsene-syslog-receiver.sematext.com"
  port="10514"
  protocol="tcp"
  template="LogseneFormat"               # use "RSYSLOG_SyslogProtocol23Format" for IP auth
  StreamDriverMode="1"
  StreamDriverPermittedPeers="*.sematext.com"
  StreamDriverAuthMode="x509/name"
)
```

RELP

To forward via RELP, load the RELP output module and then point it to **logsene-syslog-receiver.sematext.com** (or **logsene-syslog-receiver.eu.sematext.com** if using Sematext Cloud Europe) on **port 20514**.

As with TCP or UDP, specify the LogseneFormat template for authorizing with your Logsene application token:

```
$ModLoad omrelp
*.* :omrelp:logsene-syslog-receiver.sematext.com:20514;LogseneFormat
```

And the RFC-5424 template for IP-based authorization:

```
$ModLoad omrelp
*.* :omrelp:logsene-syslog-receiver.sematext.com:20514;RSYSLOG_SyslogProtocol23Format
```

Tag Your Logs

From your syslog messages, Logsene will populate a number of special fields, such as the **source** and **host**. You can also configure rsyslog to add one or more tags to logs matching certain criteria. This is useful when you want to quickly identify a special kind of logs. For example, you could tag events that come to the "kernel" facility with a severity/level of "error" as both "kernel errors" and "urgent issues". Tagging information will be visible in the Logsene web application, which you can then use for filtering, sorting, …

To achieve this, define a template similar to the ones described above, where you'd add a **tags** field. Then, you'd use a conditional to match those messages and send them to Logsene using the newly defined template:

Notice the **&~** statement - this prevents rsyslog from sending matched events twice (once with tags and once without). Make sure you place these conditionals before your main Logsene action (the one starting with ***.***).