title: Sematext Logstash Integration description: Logstash can be used to send logs to Sematext monitoring and logging platform. Parse logs with grok filter, tag specific events, aggregate and index data and metrics from different sources

In order make Logstash send logs to Sematext Logs Management app, you need to configure it with the Elasticsearch plugin to output app's endpoint, while specifying:

- **logsene-receiver.sematext.com** as the host
- **80** or **443** as the **port**
- your Logs Management App token as the index name
- **http** as the **protocol**
- **ssl** as **true** for SSL/HTTPS

Tailing a File

To send the contents of a file, you'd configure it like this:

```
input {
  file {
    path => "/var/log/apache.log"
    start_position => "beginning"           # this will also send existing contents the firs
    add_field => { "source" => "apache" }   # add a source field, for easier filtering
 }
}

output {
  elasticsearch {
          hosts => "logsene-receiver.sematext.com:443"  # this is called "hosts" in Logstash
          ssl => true               # (requires Logstash 1.5+) if you do not want to use SSL
          index => "LOGSENE_APP_TOKEN_GOES_HERE"
          manage_template => false  # Logsene will manage templates for you
 }
}
```

To get started quickly, you can simply download Logstash, unpack it, save the above configuration into a file (e.g., /etc/logstash/conf.d/logsene.conf), then start Logstash:

```
bin/logstash -f /tmp/logstash.conf
```

After it starts, logs from that file will flow to Logs Management app and you should be able to start searching them.

Parsing Unstructured Data

You may want to extract metrics from your logs, to do various analysis tasks. For example, you can make a pie chart in Kibana that shows you how often clients get 200 response codes, how often 500 and so on.

With Logstash, you can parse logs by using the grok filter. For example, you can enhance the previous configuration to parse your Apache combined log format by changing it to the following:

```
input {
  file {
    path => "/var/log/apache.log"
    add_field => { "source" => "apache" }    # add a source field, for easier filtering
    start_position => "beginning"
  }
}

filter {
  if [source] == "apache" {                  # only try to parse if logs are from the "apache
    grok {
      match => [ "message", "%{COMBINEDAPACHELOG}" ]
    }
  }
}

output {
  elasticsearch {
    hosts => "logsene-receiver.sematext.com:443"
    ssl => true
    index => "LOGSENE_APP_TOKEN_GOES_HERE"
    manage_template => false
 }
}
```

Tagging Specific Logs

Logstash populates a number of special fields, such as **host** and **@timestamp**. You can also configure it 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 from the apache log and contain the word "error" as "apache errors".

To achieve this, you can use the mutate filter to add the tags, and wrap it in a conditional that matches those specific logs:

```
filter {
  if [source] == "apache" and [message] =~ "error" {
```

```
    mutate {
      add_field => {
        "tags" => ["apache errors" ]
      }
    }
  }
}
```