The Essentials

With Logsene, we expose the Elasticsearch API so you can search your logs from your own application, or by configuring/adapting existing Elasticsearch UIs, such as Kibana. You can also use this API to index events or change the mapping.

When you use the API, here are the things you need to know:

- host name: **logsene-receiver.sematext.com** / **logsene-receiver-syslog.eu.sematext.com** (if using Sematext Cloud Europe)
- port: **80** (**443** for HTTPS)
- index name: your Logsene app token - note that this token should be kept secret

Searching

Logsene supports a subset of Elasticsearch APIs, with rich query language and extensive capabilities of searching through data you've sent to Logsene. The supported Search API's are:

- URI based search
- Request body based search
- Real time GET
- Multiple GET operations in a single request
- Multiple Search operations in a single request

For each of the operations you'll need your Logsene app token when calling **logsene-receiver.sematext.com** / **logsene-receiver-syslog.eu.sematext.com** (if using Sematext Cloud Europe). In the following examples we will use a "dummy token" - *cc5e9c1b-3046-4e43-998e-2a0b2c01b912* as the token. You should use your real Logsene App token, of course.

URI based search

The simplest search method to get your data out of Logsene is fully compatible with URI Search in Elasticsearch (https://www.elastic.co/guide/en/elasticsearch/reference/current/search-uri-request.html). You need to provide the query using the *q* parameter. For example, to search for the *internal* and *connection* terms you would run the following:

```
curl -XGET 'logsene-receiver.sematext.com/cc5e9c1b-3046-4e43-998e-2a0b2c01b912/_search?pretty
```

Note: To learn more about Apache Lucene query syntax, please refer to https://lucene.apache.org/core/6_6_0/queryparser/org/apache/lucene/queryparser/classic/package-summary.html

Request body based search

The request body based search lets us leverage full Elasticsearch query DSL language (https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html) along with its filtering capabilities (https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-filters.html) and aggregations (https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html). With full featured Elasticsearch query API we can search and find any data we are really looking for.

For example, to find documents that match the *internal* and *connection* terms run the following:

```
curl -XGET 'logsene-receiver.sematext.com/cc5e9c1b-3046-4e43-998e-2a0b2c01b912/_search?pretty' -d '{
 "query" : {
  "bool" : {
   "must" : [
    {
     "match" : {
      "_all" : "internal"
     }
    },
    {
     "match" : {
      "_all" : "connection"
     }
    }
   ]
  }
 }
}'
```

To analyze this data further we can add aggregations to our query (https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations.html) to find common status responses for example:

```
curl -XGET 'logsene-receiver.sematext.com/cc5e9c1b-3046-4e43-998e-2a0b2c01b912/_search?pretty' -d '{
 "query" : {
  "bool" : {
   "must" : [
    {
     "match" : {
      "_all" : "internal"
     }
    },
    {
     "match" : {
```

```
      "_all" : "connection"
     }
    }
   ]
  }
 },
 "aggs" : {
  "statuses" : {
   "terms" : {
    "field" : "status"
   }
  }
 }
}'
```

Response format

Logsene, just like Elasticsearch, talks to you using JSON. Here's an example response:

```
{
  "took" : 10,
  "timed_out" : false,
  "_shards" : {
    "total" : 3,
    "successful" : 3,
    "failed" : 0
  },
  "hits" : {
    "total" : 126149,
    "max_score" : 0.57406324,
    "hits" : [ {
     ...
    ]
  },
  "aggregations" : {
    ...
  }
}
```

As you can see the response is a JSON object with three main sections:

1. the header, which gives us information about the status of the response, like time it took to render it, if the query was timed out,
2. the *hits* object that includes information about returned results (total count and maximum score) and of course the *hits* array, which includes the returned documents (10 by default),

3. the *aggregations* object that includes aggregations results if we've used aggregations in our query

The real example of the results returned look as follows:


```
{
  "took" : 365,
  "timed_out" : false,
  "_shards" : {
    "total" : 3,
    "successful" : 3,
    "failed" : 0
  },
  "hits" : {
    "total" : 126149,
    "max_score" : 0.57406324,
    "hits" : [ {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29clTtUV2O9bWZ1ZaI",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29clTtUV2O9bWZ1ZaS",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29clTtUV2O9bWZ1ZaU",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29cgaeUV2O9bWZ1WBJ",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29cgaeUV2O9bWZ1WBL",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
```

```
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29cgaeUV2O9bWZ1WBN",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29dMMJUV2O9bWZ110r",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29dMMJUV2O9bWZ110t",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29dMMJUV2O9bWZ110v",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    }, {
      "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
      "_type" : "apache",
      "_id" : "AU29dMMJUV2O9bWZ110z",
      "_score" : 0.57406324,
      "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04
    } ]
  },
  "aggregations" : {
    "statuses" : {
      "doc_count_error_upper_bound" : 0,
      "sum_other_doc_count" : 0,
      "buckets" : [ {
        "key" : "200",
        "doc_count" : 126149
      } ]
    }
  }
}
```

Real time GET operation

The real time GET operation is very simple and lets us get a single document out of a particular Logsene index. To retrieve a document we need to provide Logsene with the following information:

- **index name** - it will be *_free* if your Logsene app trial has expired and you don't have a paid plan, or *_* (where *date* is *YYYY-MM-DD*) when you have a paid plan for the Logsene service,
- **type name** - the type of the document you want to retrieve,
- **document identifier** - the identifier of the document

For example, to retrieve a document with identifier *AU29tJz0UV2O9bWZ_KkU* and type *apache* from our example application identified by *cc5e9c1b-3046-4e43-998e-2a0b2c01b912* token (application is free, which means that we need to append the token with *_free* postfix to get the index name) we would run the following command:

```
curl -XGET 'logsene-receiver.sematext.com/cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free/apache/A
```

Multiple GET operations in a single request

In addition to supporting the real time GET functionality, Logsene lets one leverage Elasticsearch MGet API (https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-multi-get.html), which allows us to retrieve multiple document using the real time GET API in a single request. For example, to retrieve documents with identifier *AU29tJz0UV2O9bWZ_KkU* and *AU29rlOPUV2O9bWZ-Daw* which are of type *apache* from our example application identified by *cc5e9c1b-3046-4e43-998e-2a0b2c01b912* token we would run the following request:

```
curl -XGET 'logsene-receiver.sematext.com/cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free/_mget?pr
 "docs" : [
  { "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free", "_type" : "apache", "_id" : "AU2
  { "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free", "_type" : "apache", "_id" : "AU2
 ]
}'
```

As you can see, we are sending a *HTTP GET* request to the *_mget* REST endpoint of Logsene receiver and get back a JSON object that contains the *docs* array. Each entry of the *docs* array is identifying a single document by providing the index name (the *_index* property), the type name (the *_type* property) and the document identifier (the *_id* property).

The response to the above command would look as follows:

```
{
  "docs" : [ {
```

```
    "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
    "_type" : "apache",
    "_id" : "AU29tJz0UV2O9bWZ_KkU",
    "_version" : 1,
    "found" : true,
    "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04T0
  }, {
    "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
    "_type" : "apache",
    "_id" : "AU29rlOPUV2O9bWZ-Daw",
    "_version" : 1,
    "found" : true,
    "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-04T0
  } ]
}
```

Multiple Search operations in a single request

Similar to MGet, Logsene lets you run multiple search requests in a single HTTP request using Elasticsearch Multi Search API (https://www.elastic.co/guide/en/elasticsearch/reference/current/search-multi-search.html). The request needs to be run against _msearch REST end-point and each query needs to include two lines - meta line defining the index name and a line defining the query using Elasticsearch query DSL (https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html).

For example, the following example shows the usage of Multiple Search API:

```
curl -XGET 'logsene-receiver.sematext.com/_msearch?pretty' --data-binary '{ "index" : "cc5e9
{ "query" : { "match_all" : {} }, "size" : 1 }
{ "index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free" }
{ "query" : { "term" : { "status" : 200 } }, "size" : 1 }'
```

Keep in mind that Multiple Search API is using *–data-binary* switch in the *curl* command to keep the new line characters in the request. This is crucial to make the Multiple Search API working correctly.

The response includes standard search response for each of the included queries and for the above query will look as follows:

```
{
  "responses" : [ {
    "took" : 5,
    "timed_out" : false,
    "_shards" : {
      "total" : 2,
      "successful" : 2,
```

```
      "failed" : 0
    },
    "hits" : {
      "total" : 452761,
      "max_score" : 1.0,
      "hits" : [ {
        "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
        "_type" : "apache",
        "_id" : "AU293_WeUV2O9bWZGVUN",
        "_score" : 1.0,
        "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-
      } ]
    }
  }, {
    "took" : 14,
    "timed_out" : false,
    "_shards" : {
      "total" : 2,
      "successful" : 2,
      "failed" : 0
    },
    "hits" : {
      "total" : 387680,
      "max_score" : 1.1691506,
      "hits" : [ {
        "_index" : "cc5e9c1b-3046-4e43-998e-2a0b2c01b912_free",
        "_type" : "apache",
        "_id" : "AU293_WeUV2O9bWZGVUN",
        "_score" : 1.1691506,
        "_source":{"status": "200", "request": "OPTIONS * HTTP/1.0", "@timestamp": "2015-06-
      } ]
    }
  } ]
}
```