Yelp / **elastalert**

Easy & Flexible Alerting With ElasticSearch https://elastalert.readthedocs.org

1,302 commits **16** branches **96** releases **108** contributors Apache-2.0

Qmando committed on **GitHub** Merge pull request #1126 from micahhausler/rule-import-absolute-path ... Latest commit `1134c9e` 6 days ago

docs	Merge pull request #1083 from pralabhkumar/develop	8 days ago
elastalert	Merge pull request #1126 from micahhausler/rule-import-absolute-path	6 days ago
example_rules	Added functionality to have multiple keys in compare_keys in Change rule	12 days ago
tests	Merge pull request #1126 from micahhausler/rule-import-absolute-path	6 days ago
.gitignore	Upgraded to Boto3	2 months ago
.pre-commit-config.yaml	Add support for referencing a top-level rule in any arbitrary alert f…	a year ago
.travis.yml	Removed Python 2.6 support	a month ago
Dockerfile-test	Update Dockerfile-test to install pip lib deps	a month ago
LICENSE	Added Apache license	2 years ago
Makefile	Add framework for running tests in a docker container:	8 months ago
README.md	Document how to build the docs	3 months ago
changelog.md	Version 0.1.11	a month ago
config.yaml.example	Upgraded to Boto3	2 months ago
docker-compose.yml	Add framework for running tests in a docker container:	8 months ago
requirements-dev.txt	Removed Python 2.6 support	a month ago
requirements.txt	Removed pin on elasticsearch library, added back requirements.txt	26 days ago
setup.cfg	Removed Python 2.6 support	a month ago
setup.py	Bump version number in setup.py (nobody ever forgets to do that...)	20 days ago
supervisord.conf.example	documentation and examples for entry_point	2 years ago
tox.ini	Removed Python 2.6 support	a month ago

README.md

ElastAlert - Read the Docs.

Easy & Flexible Alerting With Elasticsearch

ElastAlert is a simple framework for alerting on anomalies, spikes, or other patterns of interest from data in Elasticsearch.

ElastAlert works with all versions of Elasticsearch.

At Yelp, we use Elasticsearch, Logstash and Kibana for managing our ever increasing amount of data and logs. Kibana is great for visualizing and querying data, but we quickly realized that it needed a companion tool for alerting on inconsistencies in our data. Out of this need, ElastAlert was created.

If you have data being written into Elasticsearch in near real time and want to be alerted when that data matches certain patterns, ElastAlert is the tool for you. If you can see it in Kibana, ElastAlert can alert on it.

Overview

We designed ElastAlert to be reliable, highly modular, and easy to set up and configure.

It works by combining Elasticsearch with two types of components, rule types and alerts. Elasticsearch is periodically queried and the data is passed to the rule type, which determines when a match is found. When a match occurs, it is given to one or more alerts, which take action based on the match.

This is configured by a set of rules, each of which defines a query, a rule type, and a set of alerts.

Several rule types with common monitoring paradigms are included with ElastAlert:

- "Match where there are X events in Y time" (`frequency` type)
- "Match when the rate of events increases or decreases" (`spike` type)
- "Match when there are less than X events in Y time" (`flatline` type)
- "Match when a certain field matches a blacklist/whitelist" (`blacklist` and `whitelist` type)
- "Match on any event matching a given filter" (`any` type)
- "Match when a field has two different values within some time" (`change` type)
- "Match when a never before seen term appears in a field" (`new_term` type)
- "Match when the number of unique values for a field is above or below a threshold (`cardinality` type)

Currently, we have support built in for the following alert types:

- Email
- JIRA
- OpsGenie
- Commands
- HipChat
- Slack
- Telegram
- AWS SNS
- VictorOps
- PagerDuty
- Exotel
- Twilio
- Gitter

Additional rule types and alerts can be easily imported or written.

In addition to this basic usage, there are many other features that make alerts more useful:

- Alerts link to Kibana dashboards
- Aggregate counts for arbitrary fields
- Combine alerts into periodic reports
- Separate alerts by using a unique key field
- Intercept and enhance match data

To get started, check out `Running ElastAlert For The First Time` in the documentation.

Running ElastAlert

```
$ python elastalert/elastalert.py [--debug] [--verbose] [--start <timestamp>] [--end <timestamp>] [--rule <filename.yaml>] [--config <filename.yaml>]
```

`--debug` will print additional information to the screen as well as suppresses alerts and instead prints the alert body.

`--verbose` will print additional information without supressing alerts.

`--start` will begin querying at the given timestamp. By default, ElastAlert will begin querying from the present. Timestamp format is `YYYY-MM-DDTHH-MM-SS[-/+HH:MM]` (Note the T between date and hour). Eg: `--start 2014-09-26T12:00:00` (UTC) or `--start 2014-10-01T07:30:00-05:00`

`--end` will cause ElastAlert to stop querying at the given timestamp. By default, ElastAlert will continue to query indefinitely.

`--rule` will allow you to run only one rule. It must still be in the rules folder. Eg: `--rule this_rule.yaml`

`--config` allows you to specify the location of the configuration. By default, it is will look for config.yaml in the current directory.

Third Party Tools And Extras

Bitsensor Kibana plugin

Configure and test rules via a Kibana plugin

Documentation

Read the documentation at Read the Docs.

To build a html version of the docs locally

```
pip install sphinx_rtd_theme sphinx
cd docs
make html
```

View in browser at build/html/index.html

Configuration

See config.yaml.example for details on configuration.

Example rules

Examples of different types of rules can be found in example_rules/.

- `example_spike.yaml` is an example of the "spike" rule type, which allows you to alert when the rate of events, averaged over a time period, increases by a given factor. This example will send an email alert when there are 3 times more events matching a filter occurring within the last 2 hours than the number of events in the previous 2 hours.
- `example_frequency.yaml` is an example of the "frequency" rule type, which will alert when there are a given number of events occuring within a time period. This example will send an email when 50 documents matching a given filter occur within a 4 hour timeframe.
- `example_change.yaml` is an example of the "change" rule type, which will alert when a certain field in two documents changes. In this example, the alert email is sent when two documents with the same 'username' field but a different value of the 'country_name' field occur within 24 hours of each other.
- `example_new_term.yaml` is an example of the "new term" rule type, which alerts when a new value appears in a field or fields. In this example, an email is sent when a new value of ("username", "computer") is encountered in example login logs.

Frequently Asked Questions

My rule is not getting any hits?

So you've managed to set up ElastAlert, write a rule, and run it, but nothing happens, or it says `0 query hits` . First of all, we recommend using the command `elastalert-test-rule rule.yaml` to debug. It will show you how many documents match your filters for the last 24 hours (or more, see `--help`), and then shows you if any alerts would have fired. If you have a filter in your rule, remove it and try again. This will show you if the index is correct and that you have at least some documents. If you have a filter in Kibana and want to recreate it in ElastAlert, you probably want to use a query string. Your filter will look like

```
filter:
- query:
    query_string:
      query: "foo: bar AND baz: abc*"
```

If you receive an error that Elasticsearch is unable to parse it, it's likely the YAML is not spaced correctly, and the filter is not in the right format. If you are using other types of filters, like `term`, a common pitfall is not realizing that you may need to use the analyzed token. This is the default if you are using Logstash. For example,

```
filter:
- term:
    foo: "Test Document"
```

will not match even if the original value for `foo` was exactly "Test Document". Instead, you want to use `foo.raw`. If you are still having trouble troubleshooting why your documents do not match, try running ElastAlert with `--es_debug_trace /path/to/file.log`. This will log the queries made to Elasticsearch in full so that you can see exactly what is happening.

I got hits, why didn't I get an alert?

If you got logs that had `X query hits, 0 matches, 0 alerts sent`, it depends on the `type` why you didn't get any alerts. If `type: any`, a match will occur for every hit. If you are using `type: frequency`, `num_events` must occur within `timeframe` of each other for a match to occur. Different rules apply for different rule types.

If you see `X matches, 0 alerts sent`, this may occur for several reasons. If you set `aggregation`, the alert will not be sent until after that time has elapsed. If you have gotten an alert for this same rule before, that rule may be silenced for a period of time. The default is one minute between alerts. If a rule is silenced, you will see `Ignoring match for silenced rule` in the logs.

If you see `X alerts sent` but didn't get any alert, it's probably related to the alert configuration. If you are using the `--debug` flag, you will not receive any alerts. Instead, the alert text will be written to the console. Use `--verbose` to achieve the same affects without preventing alerts. If you are using email alert, make sure you have it configured for an SMTP server. By default, it will connect to localhost on port 25. It will also use the word "elastalert" as the "From:" address. Some SMTP servers will reject this because it does not have a domain while others will add their own domain automatically. See the email section in the documentation for how to configure this.

Why did I only get one alert when I expected to get several?

There is a setting called `realert` which is the minimum time between two alerts for the same rule. Any alert that occurs within this time will simply be dropped. The default value for this is one minute. If you want to receive an alert for every single match, even if they occur right after each other, use

```
realert:
  minutes: 0
```

You can of course set it higher as well.

How can I prevent duplicate alerts?

By setting `realert`, you will prevent the same rule from alerting twice in an amount of time.

```
realert:
  days: 1
```

You can also prevent duplicates based on a certain field by using `query_key`. For example, to prevent multiple alerts for the same user, you might use

```
realert:
  hours: 8
query_key: user
```

Note that this will also affect the way many rule types work. If you are using `type: frequency` for example, `num_events` for a single value of `query_key` must occur before an alert will be sent. You can also use a compound of multiple fields for

this key. For example, if you only wanted to receieve an alert once for a specific error and hostname, you could use

```
query_key: [error, hostname]
```

Internally, this works by creating a new field for each document called `field1,field2` with a value of `value1,value2` and using that as the `query_key`.

The data for when an alert will fire again is stored in Elasticsearch in the `elastalert_status` index, with a `_type` of `silence` and also cached in memory.

How can I change what's in the alert?

You can use the field `alert_text` to add custom text to an alert. By setting `alert_text_type: alert_text_only`, it will be the entirety of the alert. You can also add different fields from the alert by using Python style string formatting and `alert_text_args`. For example

```
alert_text: "Something happened with {0} at {1}"
alert_text_type: alert_text_only
alert_text_args: ["username", "@timestamp"]
```

You can also limit the alert to only containing certain fields from the document by using `include`.

```
include: ["ip_address", "hostname", "status"]
```

My alert only contains data for one event, how can I see more?

If you are using `type: frequency`, you can set the option `attach_related: true` and every document will be included in the alert. An alternative, which works for every type, is `top_count_keys`. This will show the top counts for each value for certain fields. For example, if you have

```
top_count_keys: ["ip_address", "status"]
```

and 10 documents matched your alert, it may contain something like

```
ip_address:
127.0.0.1: 7
10.0.0.1: 2
192.168.0.1: 1

status:
200: 9
500: 1
```

How can I make the alert come at a certain time?

The `aggregation` feature will take every alert that has occured over a period of time and send them together in one alert. You can use cron style syntax to send all alerts that have occured since the last once by using

```
aggregation:
  schedule: '2 4 * * mon,fri'
```

I have lots of documents and it's really slow, how can I speed it up?

There are several ways to potentially speed up queries. If you are using `index: logstash-*`, Elasticsearch will query all shards, even if they do not possibly contain data with the correct timestamp. Instead, you can use Python time format strings and set `use_strftime_index`

```
index: logstash-%Y.%m
use_strftime_index: true
```

Another thing you could change is `buffer_time` . By default, ElastAlert will query large overlapping windows in order to ensure that it does not miss any events, even if they are indexed in real time. In config.yaml, you can adjust `buffer_time` to a smaller number to only query the most recent few minutes.

```
buffer_time:
  minutes: 5
```

By default, ElastAlert will download every document in full before processing them. Instead, you can have ElastAlert simply get a count of the number of documents that have occured in between each query. To do this, set `use_count_query: true` . This cannot be used if you use `query_key` , because ElastAlert will not know the contents of each documents, just the total number of them. This also reduces the precision of alerts, because all events that occur between each query will be rounded to a single timestamp.

If you are using `query_key` (a single key, not multiple keys) you can use `use_terms_query` . This will make ElastAlert perform a terms aggregation to get the counts for each value of a certain field. Both `use_terms_query` and `use_count_query` also require `doc_type` to be set to the `_type` of the documents. They may not be compatible with all rule types.

Can I perform aggregations?

The only aggregation supported currently is a terms aggregation, by setting `use_terms_query` .

I'm not using @timestamp, what do I do?

You can use `timestamp_field` to change which field ElastAlert will use as the timestamp. You can use `timestamp_type` to change it between ISO 8601 and unix timestamps. You must have some kind of timestamp for ElastAlert to work. If your events are not in real time, you can use `query_delay` and `buffer_time` to adjust when ElastAlert will look for documents.

I'm using flatline but I don't see any alerts

When using `type: flatline` , ElastAlert must see at least one document before it will alert you that it has stopped seeing them.

How can I get a "resolve" event?

ElastAlert does not currently support stateful alerts or resolve events.

Can I set a warning threshold?

Currently, the only way to set a warning threshold is by creating a second rule with a lower threshold.

License

Read the documentation at Read the Docs.

Questions? Drop by #elastalert on Freenode IRC.