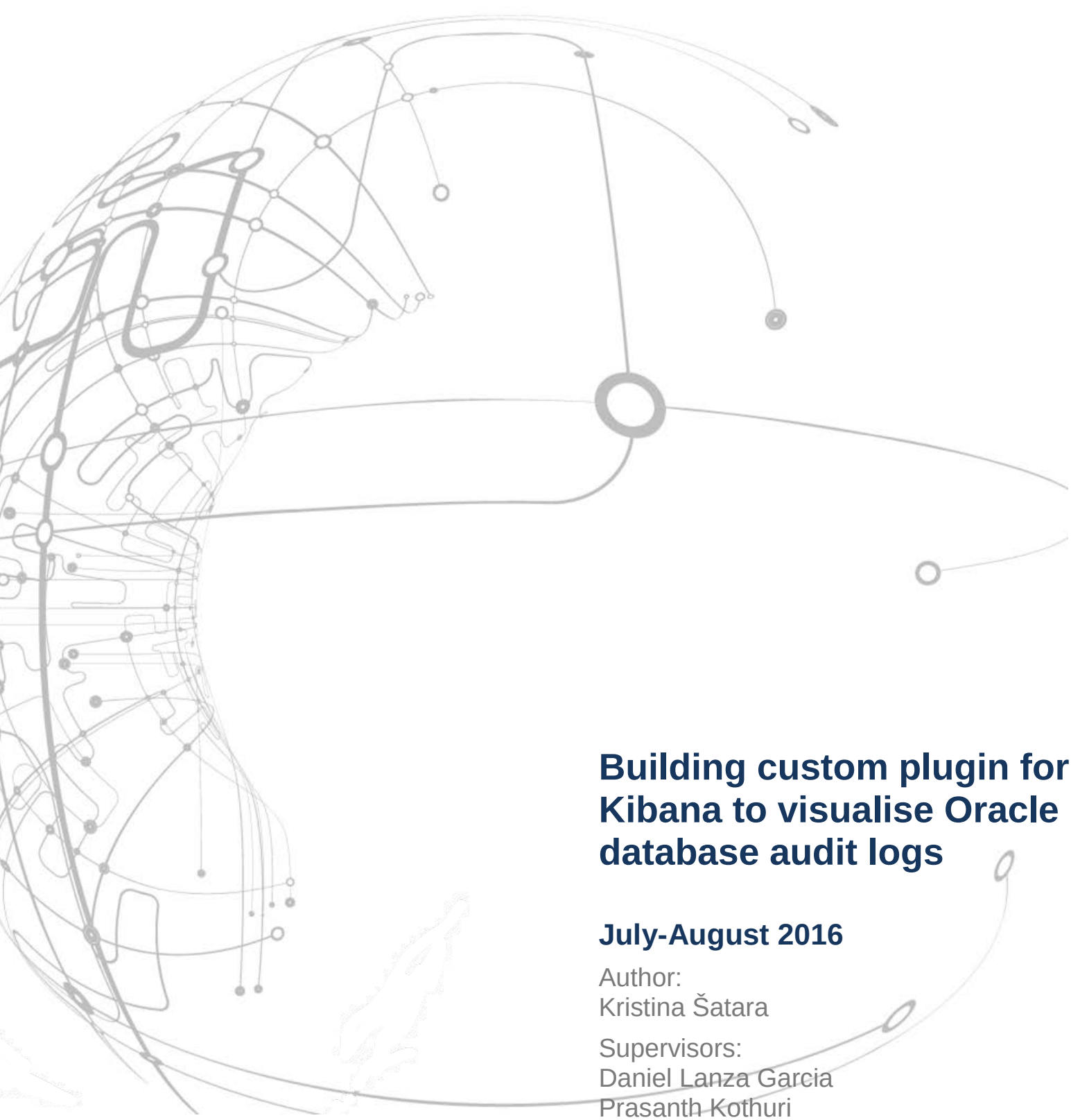

Building custom plugin for Kibana to visualise Oracle database audit logs

July-August 2016

Author:
Kristina Šatara

Supervisors:
Daniel Lanza Garcia
Prasanth Kothuri

CERN openlab Summer Student Report 2016

15 years CERN openlab

Project Specification

A central platform which provides a highly scalable, secure and central repository that stores consolidated audit data from Oracle databases is being implemented at CERN. Among other purposes, this central platform will be used for reporting. The reports will provide a holistic view of activity across all oracle databases and will include compliance reports, activity reports and privilege reports.

With guidance and support from the supervisor, the candidate will customize the mentioned reports in order to improve the visualizations and graphs contained in it using a flexible analytics and visualization platform called Kibana.

The candidate will gain experience in building custom visualizations, aggregations and styling using Kibana along with mastering real-time summaries and charting of streaming data.

Abstract

Report provides more details about Openlab Summer Student Programme project. Project was Building custom plugin for Kibana to visualise Oracle database audit logs. Chapter one is explaining bigger picture about the project. Chapter two is giving more details about the project architecture. It is also explaining more about used technologies – about Elasticsearch and Kibana. This is important, because it explains problems with current technologies and motivation for this project.

Chapter three is explaining student's project. It is giving information about used technologies, plugin structure and current state of the plugin. It also explains which difficulties had student during the work.

Considering the fact that some of readers may be interested to try using the plugin, to extend it, or even to create own plugin - chapter three is providing readers with the info how they can do it. In the same chapter can be found ideas for further work on the project.

Table of Contents

1 Introduction

Nowadays, there are around 10000 people working at CERN. Regardless are they summer students, technical students, physicists, programmers or other staff, almost all of them have computer accounts. CERN offers numerous services to its employees (EDH, mails...). Various database storage systems are needed to store this amount of data. Some of this data comes from physics experiments, some from employees.

Because of that CERN has different teams with different tasks. For example, there is team whose task is to ensure security in these services, or team for optimization of the code which is used for simulation or in experiments.

I had opportunity to work in IT-DB group. They have different projects, from which one is working on building central repository for Oracle database audit logs. Also, they are providing that linear and alert logs are parsed and stored in this central repository. Beside this, in this project are present performance metrics for troubleshooting and capacity planning.

My task was to develop a custom plugin for Kibana. This plugin should visualise Oracle database audit logs.

2 Architecture

As previously mentioned, in IT-DB group is being built central repository for Oracle database audit logs. It is important to mention that all the data are saved in more Oracle databases.

Flume presents distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of streaming event data. Flume is used in CERN to collect the data and to forward it to HDFS or Elastic Stack.

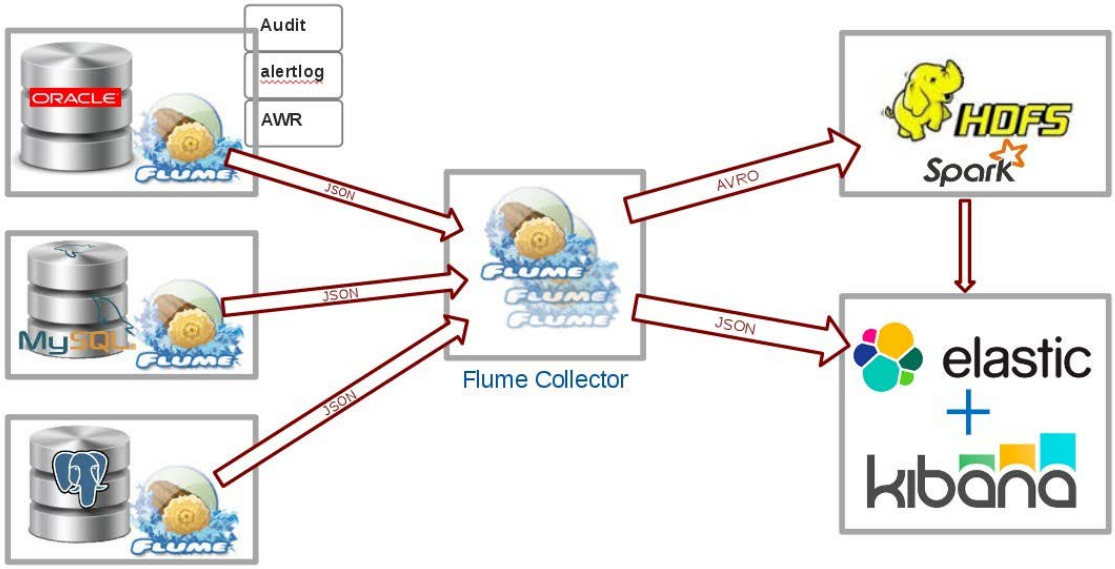

Image 1.

Elastic Stack

Elastic Stack (previously ELK Stack) consists of Elasticsearch, Logstash, Kibana and Beats. All together, they present powerful tool for storing, searching and visualizing data. Each one of them has its own function which will be described in more details.

1.1 Elasticsearch

Elasticsearch is a distributed, open source search and analytics engine, designed for horizontal scalability, reliability, and easy management. It provides real-time search and analytics capabilities, scaling horizontally if your data grows by adding more nodes.

Elasticsearch clusters detect new or failed nodes and they reorganize data automatically. Developer-friendly query API supports multilingual search, geolocation, contextual did-you-mean suggestions, autocomplete, and result snippets.

Almost any action can be performed using RESTful API using JSON over HTTP.

Data organisation in Elastic

Elasticsearch organises data in indices (concept of index is similar to concept of database in relational database). Each index consists of one or more types (concept of type is similar to concept of table in relational database). And each type consists of more specific entities. Each entity has its own id or primary key and it presents one specific document.

Image 2. shows some of the indices in Elasticsearch cluster.

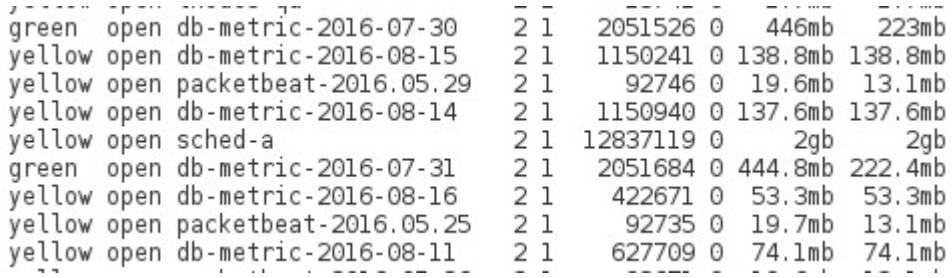

Image 2.

Querying the data in Elasticsearch

Elasticsearch has great possibilities for searching the data. It provides you to get a really quick response using Query DSL. From my personal experience, querying in Elasticsearch is very useful once you understand how it works.

Aggregations

There are many types of the aggregations. Main types are: metric, bucketing, pipeline and matrix.

Bucketing – builds buckets from which each has its own key and document criterion. At the end, we get result which is list of buckets, where every bucket consists of more documents

Metric – a type of aggregation that computes metrics over a set of documents

Pipeline – aggregates the output of some other aggregations and their associated metrics.

Matrix – aggregations that operate on multiple fields and produce a matrix result based on the values extracted from the requested document fields.

Image 3. shows query for getting all the unique metric names in db-metric-2016-08-14 index and image N result of this query.

Image 3.

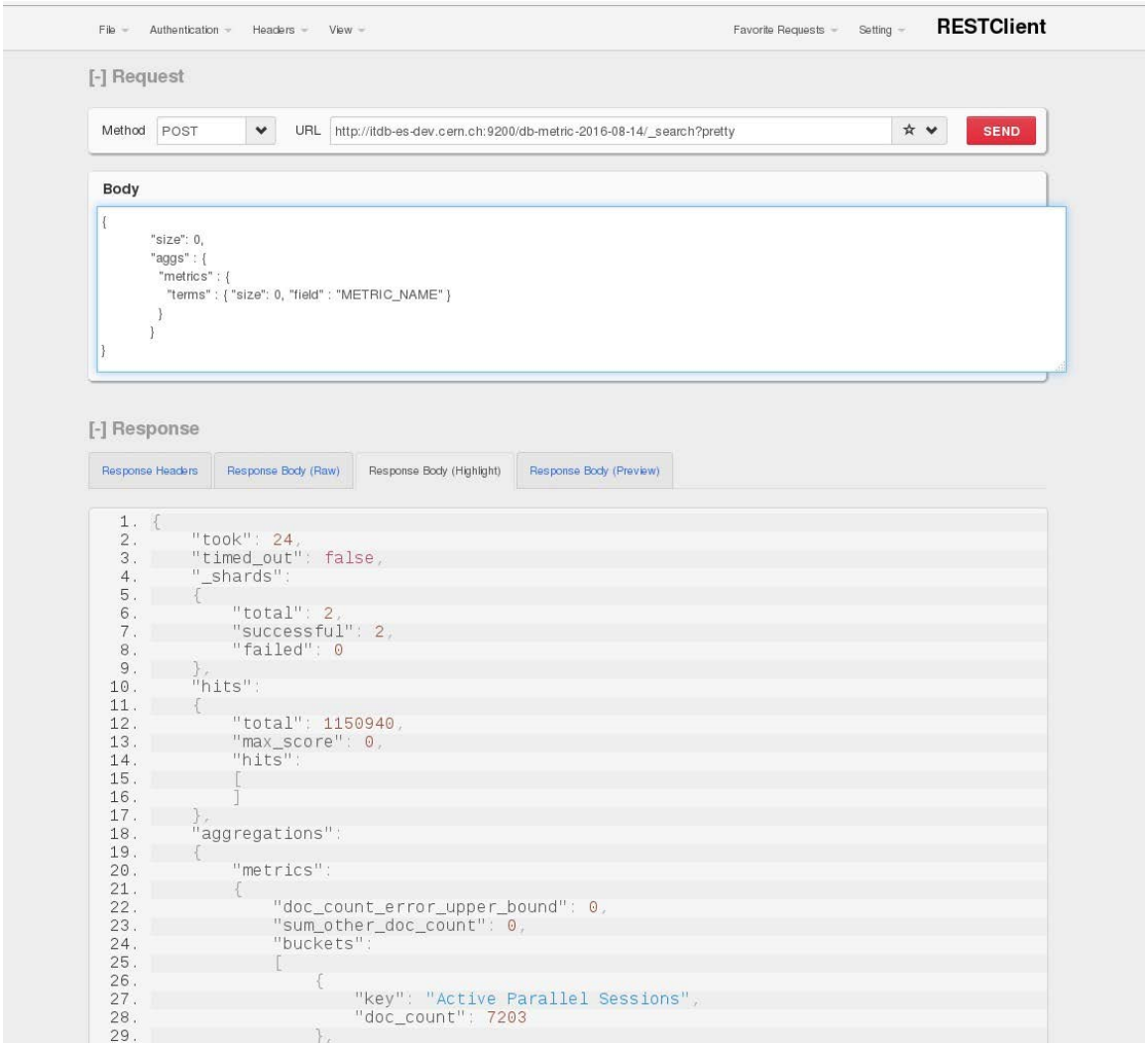

Image 4.

1.2 Kibana

Kibana is an open source data visualization platform that allows you to interact with your data through powerful graphics. It is designed to work with Elasticsearch and it also benefits from Elasticsearch's analytics and search capabilities. Easily visualise data pushed

into Elasticsearch from Logstash, ES-Hadoop, Beats, or third-party technologies like Apache Flume, Fluentd and many others.

Also, it provides real-time summary and charting of streaming data, instant sharing and embedding of dashboards, simple data exporting and merging them with other data sets. All this is really intuitive for users.

So, basically, it is interacting with REST API from Elasticsearch and visualising big amount of data in intuitive way. Users can choose whether they want bar charts, line and scatter plots, histograms, pie charts, or maps. They can organize more charts into a dashboard.

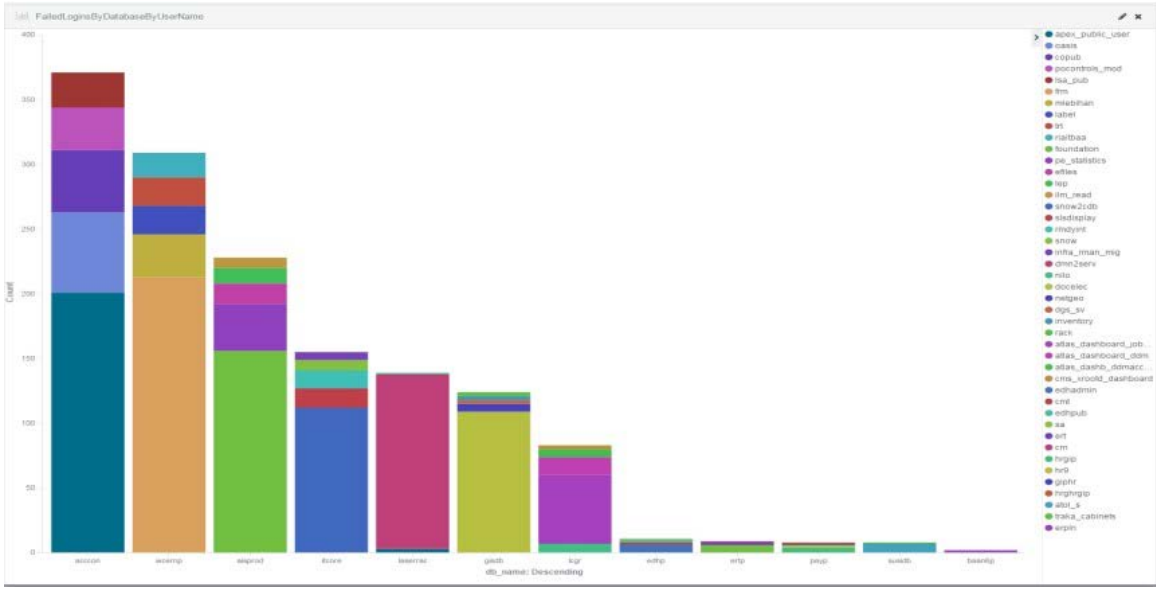

Image 5.

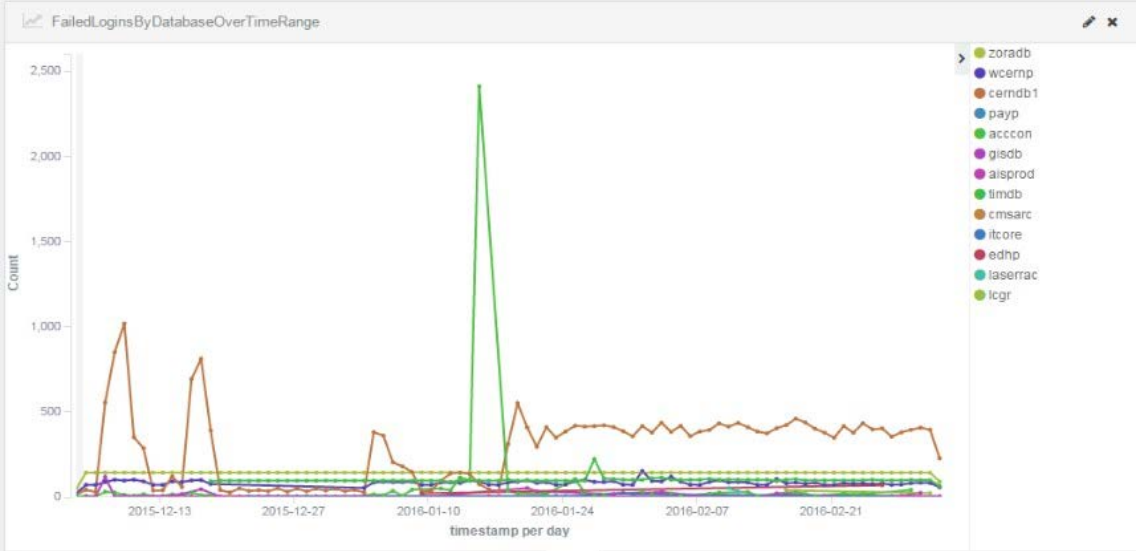

Image 6.

When user wants to create a visualisation in Kibana, he needs to know more details about search in Elasticsearch. He needs to know more about aggregations – metrics, bucketing in order to define which data he wants to show in which axis. For example, he can group data on x axis in buckets (by date, some fields, etc..).

We can conclude that it is obvious that it would be good if some details of the implementation were hidden from the users. More about this in next chapter, because this is the part of the motivation for my project.

3 Kibana plugin

Motivation

As previously mentioned, motivation for this project is to hide the implementation details of search in Elasticsearch for user that uses Kibana plugin. We want to provide user with the possibility to choose parameters from select list, and then the chart is shown. This would make users job simpler and easier. Also, motivation was to research more about building plugins in Kibana in order to visualise Oracle database audit logs.

Technologies

From technologies, there were used AngularJS (JavaScript framework, which is great for web development, it is MVW framework; Model-View-Whatever), D3 (JavaScript library for visualisations), Jira (for tracking problems), Gitlab (for versions of the project). Also, npm package manager was used as each plugin presents different npm module.

Project organisation

I used Kibana plugin generator called Yeoman generator, in order to create simple Kibana plugin. This generator is great, because it creates basic structure for the plugin. As already mentioned, each plugin is npm module. That means that every plugin has its own package.json file for list of dependencies for the plugin and node_modules folder which consists of dependency packages.

Image 7. shows current state of the project. As we can see, it consists of node_modules folder, public folder and server folder. Beside this, there are index.js and package.json files. Index.js is the entering point for the plugin.

Public folder consists of templates folder, which further consists of html and js files. Html files present web pages which are served to the users, and js files present controllers for those pages. Those js files are actually angular modules.

Server folder consists of routes folder. In routes folder there are js files, from every file present different route. These routes provide controllers from /public/templates folder to

get the data. For example, you can get list of all metric names, or you can send query parameters and get documents that correspond to these querying parameters.

Image 7.

Practical problems

First idea was to use Kibana's existing visualisations for plotting the chart. But during the time, it turned out that we couldn't manage to do that easily. When we searched how other people solved this problem, we found out it is theoretically possible, but practically we should find some other way to visualise the data. At this point, it was decided to use D3

library (JavaScript library for creating different types of visualisations). Kibana itself is using D3 library for creating the visualisations.

Also, when working with new technologies, it's normal to face with lack of the documentation and some other learning materials. All this was too superficial for me and could help only in the first stages of the project, when I wanted to implement some simple functionalities.

Current state of the project

Current state of the project can be seen in image 8. There is possibility to choose parameters for visualisation. These parameters are beginning and end date, metric name and id. After choosing the parameters, visualisation for Oracle database audit logs is shown.

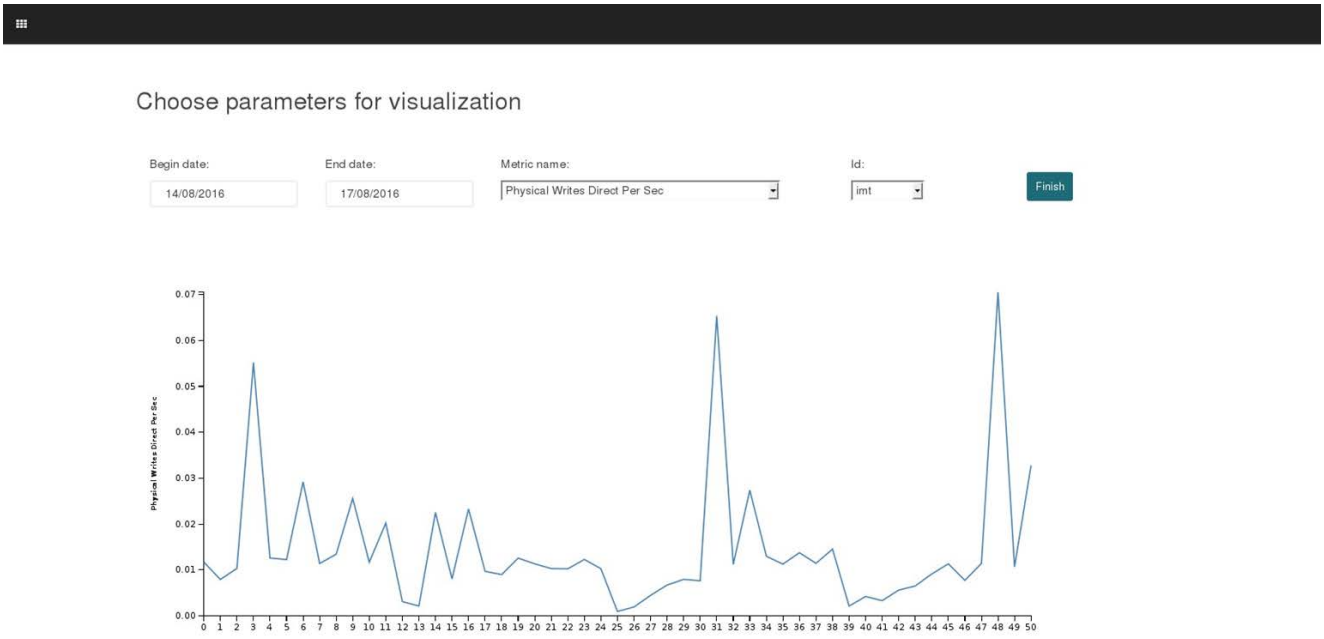

Image 8.

After user chooses beginning date, query is sent to ElasticSearch to get list of all metric names from this date. After choosing other parameters and clicking on Finish button, query is created from these parameters and also sent to Elasticsearch. What we need to know when we're creating a query are next parameters: metric name, database id, beginning and ending date.

In image 9. is shown query which is sent to Elasticsearch. As we can see, we also need to specify interval for each bucket. Interval is computed from beginning and ending date, and it is expressed in hours.

```
body:
    {
      "size": 0,
      "aggs": {
        "aggQueryData": {
          "date_histogram": {
            "field": "END_TIME",
            "interval": interval,
            "time_zone": "Europe/Berlin",
            "min_doc_count": 1
          },
          "aggs": {
            "avg_value": {
              "avg": {
                "field": "VALUE"
              }
            }
          }
        }
      },
      "query": {
        "filtered": {
          "query": {
            "query_string": {
              "query": query,
              "analyze_wildcard": true
            }
          },
          "filter": {
            "bool": {
              "must": [
                {
                  "range": {
                    "END_TIME": {
                      "gte": startDate,
                      "lte": endDate,
                      "format": "yyyy-MM-dd"
                    }
                  }
                },
                {
                  "term": {
                    "METRIC_NAME": metricName
                  }
                },
                {
                  "term": {
                    "oracle_sid": id
                  }
                }
              ]
            }
          }
        }
      }
    }
```

Image 9.

After getting data from Elasticsearch, D3 library is being used for creating line chart. X axis shows time buckets – data is group by computed interval. Y axis shows average value of “value” field. This field “value” semantic shows average value of time that was needed for execution of some event (presented by metric name field) for matching bucket, or it can show average disk space that was used during the event.

How to use the plugin

Firstly, all plugins in Kibana should be places in installedPlugins folder (for Kibana 4.3 version, for other versions this could be different).

As plugin is one npm module, it has package.json file. This file has list of all dependencies important for the project. Running npm install in your terminal, all dependencies listed in package.json will be installed in plugin in node_modules folder.

As plugin is using data from db-metric index from Elasticsearch, this index should exist. In order to use to plugin, user needs to follow few steps:

1) switch to Kibana plugin directory

   cd /opt/kibana/installedPlugins

2) clone the repository (install the plugin)

   git clone https://github.com/MsSquirrel/OracleLogs.git

3) install dependencies mentioned in package.json

   npm install

4) restart Kibana

   service kibana restart

How to extend the plugin

Structure of the plugin is already explained. If user wants to extend the plugin, he can add new html files and add new links in index.html to his html pages. Also, for this new views, user needs to add new controllers in /public folder. This controller concerns about model and the data. Controller uses routes which are defined in /server/routes folder.

These routes provide controller with different data. For example, if user wants all metric names in Oracle database, he can use already existing route /api/proba/getAllMetricNames/{beginDate}. Begin date is parameter, so it can be changed. You can look up to already existing routes, and create your own route. Routes are communicating with Elasticsearch and getting the data from it.

To conclude, if you want to extend the plugin - you need to add new html page, controller for it, and routes (if you need some data from Elasticsearch – or from Oracle database).

Further work

Currently, plugin is used only for db-metric index visualisation. This could be extended to more indices, instead of just one index.

Further work could be extending the plugin to include visualisation for Oracle database alert and listener logs. As each index has its own fields, we could provide user to choose which index to visualise, and from that index, which exactly fields. Also, machine learning could be used for learning some patterns in user's behaviour. This could help in improving security or in analysing the user's behaviour.

As already mentioned, there were some difficulties to use Kibana's existing visualisations and because of the lack of time, it was decided to use D3 library. So, further work could be exploring Kibana's source code in more details than I have done it. After all, maybe it isn't so complicated to use Kibana's internal libraries and to use its visualisations. And for surely, they look much nicer than simple charts from D3 which were used.

4 Conclusion

Functionalities that are already described are available and can be used already. Hopefully, it will help users to save their time. Instead of learning details of search in Elasticsearch, they can concentrate on other tasks. They will get the chart they wanted, and they can use it in their own purpose. For example, it can be pattern recognition, or other way of analysing the data.

It's also proposed possible further work which can be done in some next period.

The plugin code is available in CERN's git repository, where can be found details about how to use plugin, versions of Kibana, Elasticsearch, etc.

References

- www.elastic.co

- www.timroes.de

- http://logz.io/blog/kibana-visualizations/

- www.npmjs.com/package/generator-kibana-plugin

- https://github.com/elastic/kibana

- https://github.com/elastic/timelion