Log Analysis and Visualization using ELK Stack

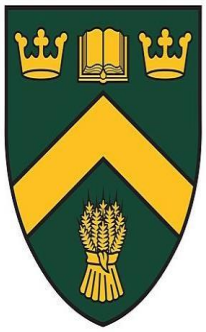
Kiranmai Arva

Graduate Co-op Work Term-2 Report

Department of Computer Science

Supervisor

Dr. Lisa Fan



University of Regina

Regina, Saskatchewan

72 Banchory Crescent,

Kanata, ON

Date:15th March 2020

Dr. Lisa Fan

Computer Science Department,

University of Regina,

Saskatchewan, S4S0A2

Dear Dr. Fan,

The co-op work report titled ‘Log Analysis and visualization using Elastic search Logstash and Kibana (ELK Stack)’ is an abstract of my second work term at Nokia. The main motive of this report is to present the knowledge and experience gained on log collection, network traffic analysis and visualization using Elastic search Logstash Kibana (ELK).

My role at Nokia is Software and Automation Tools Developer. My area of focus is on network traffic and log analysis. I was assigned to develop a real-time network traffic monitoring system using ELK. As a part of this project, I need to collect the network logs using gRPC (google Remote procedure call) and transfer them to Elastic search by using Logstash. Elastic search is a search engine database especially used for log analysis. Finally, visualizations are built using Kibana. Monitoring these real-time visualizations helps to predict anomalies in traffic and other information.

For a telecom company like Nokia, it is important to monitor the network traffic to avoid outages, detect malicious traffic, identify security threats and many other scenarios.

The report is original and solely created by me based on the experience I gained at Nokia. I have also acknowledged all the referred sources and materials.

Sincerely,

Kiranmai Arva

Student ID: 200416487

Executive summary

Telecom companies need to analyze the logs generated by different servers, routers, switches, and many other devices. Every device connected to the network will generate logs. Logs contain intelligence information like events associated, timestamp, Operating system information, and other information related to the event. All the logs are auto generated by the respective devices. The process of analyzing and extracting useful information to predict some forecasting events or outages in the network is called log analysis. Moreover, the performance and health of a device can be monitored using these Logs.

Logs generated by each device and application are in different formats. For each event, numerous logs files are generated, and It is difficult for humans to directly read and understand the logs files. Many tools are being developed by industries, one among them is Elastic search Logstash Kibana. These three are open source tools. These three tools together made log analysis, monitoring, and visualization simple. All these three applications together called Elastic stack.

Elastic search is an open-source search engine analytics built on the top of Apache Lucene. One can query and aggregate the data present in the elastic search. Data from various sources can be loaded into this search engine. **Logstash** is a powerful tool as it can import data from various sources and loaded them to multiple destinations at once. It can also parse the data, filter the data before transforming the data to the destination. **Kibana** is a visualization tool, powerful interactive visualizations can be built based on the data present in the Elasticsearch.

This report focuses on the introduction of ELK and different plugins, installation of ELK on the Centos operating system, importing the logs from source using Logstash, and transferring them to Elastic search and visualizing the useful metrics using Kibana.

Table of Contents

Elastic Search

1 Introduction

Elastic search is an open-source search engine built on the top of Apache Lucene. Data can be loaded to elastic search from multiple sources by creating an Index. Before data is indexed in elastic search the raw data needs to be parsed and normalized. The data is stored in the form of JSON documents. Each document in return will have fields. Full-text searches can be performed on the elastic search as it uses a data structure called an inverted index. To analyze the data present in the elastic search aggregations can be performed and can gain valuable insights about the metrics and time-series events. In general, Elastic search is a NoSQL DB.

1.1 Components of Elastic Search

Below are the backend components elastic search comprised with

- Node
- Cluster
- Index
- Document

Node

A single instance of an elastic search is called a node. To have a system with high availability multi-node structure is preferred. There are four types of nodes available [1]

Master node: Master node acts as a supervisor for all other nodes under one cluster.

Data node: The data is stored in the data node. Data related operations like ‘CURD’ are performed under this node

Ingest node: The data preprocessing is done on Ingest node. This action is performed even before creating an Index

Tribe node: This node is responsible for communication among multiple clusters.

Cluster

Cluster is a collection of one or more nodes together. All the instances of elastic search are connected under one single instance and that single instance is called a node. When data is huge, data can be stored among many nodes and it is called a multi-node cluster. The failure of one node does not affect the process as data is distributed among multiple nodes.

Index

Each index contains shards, shards stores data and they are the instances of apache Lucene. Under each index, there can be multiple types, documents, and fields. When compared to the relational database, a database is equivalent to an index in elastic search.

Documents

Data is stored in the form of JSON documents under each Index. Each document contains key, value pairs.

When compare to Sql database the terms are as follows.

MSQL => Databases => Tables => Columns/Rows

Elasticsearch => Indices => Types => Documents

Below is the JSON format of data storage in Elastic search.

```
{
“_index”: “Hr Info”,   #Index name Hr Info
“_type1” : “doc”,       #Type refers to table in Sql
“_id” :”101”,
```

“_score”:1,

“_source” : {

“Employee id”: 1234, # Key value field of data

“Age” : “25”

“Name”:”kiran”

}}

1.2 Commands and Operations on Elastic Search

Elastic search is a server that can process JSON requests and responses back with JSON results. To search an object or perform operations on elastic search Curl commands are used [1].

curl

Curl is a command line tool used to transfer data to and from the server. It supports many protocols like HTTP, HTTPS, FTP, FTPS, SFTP, IMAP, SMTP.

List of request methods used by curl are:

- GET method
- POST method
- DELETE method
- PUT method

GET method

Used to fetch the objects from the specified API or URL. To fetch the information about the indexes from elastic search following command is used.

- curl -XGET 127.0.0.1:9200/test/_search?pretty

In the above command test is the index which is created on elastic search. It retrieves all the indexes with name test in the json format.

POST method

To post information on the server post method is used.

- curl -XPOST "HTTP://LOCALHOST:9200/ test /message" -d'
  {
  "text": "This is test index"
  }'

DELETE method

Delete method deletes the objects from the specified URL in the command.

- curl -XDELETE localhost:9200/test

The above command is used to delete the index with name test on the elastic search

PUT method

PUT method will update the all objects in the elastic search.

PUT /test/_mapping

{

“properties”: {

“server”:

{

“type”: “RC123 “

}

}

Logstash

2 Introduction

Logstash is a lightweight server-side application used to collect logs. Logstash can transfer data from more than one source to multiple destinations at once. Logstash can parse, transform, and filters data before it transforms into the destination. The logic for parsing and filtering are edited in the conf file. Users can write the conf files by analyzing the data generated at the source and filter accordingly. Unstructured data can be converted to structured data with the help of plugins in the conf file. The purpose of logstash in ELK is log aggregation and filtering logs. If multiple applications produce logs, a centralized log management application is required for better processing of multiple logs. Logstash filter plugin can exclude the unwanted logs [2].

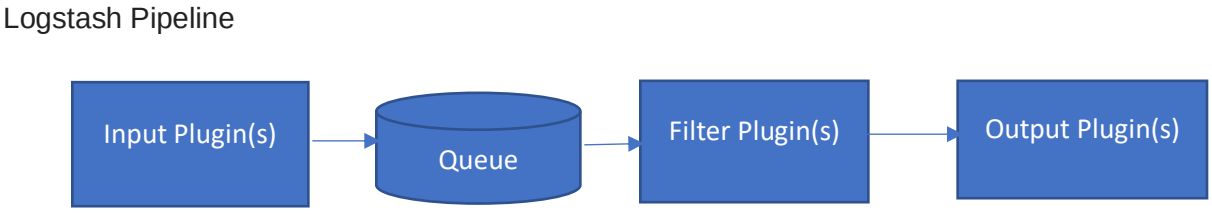


Figure 1. Logstash Pipeline

2.1 Input Plugin

By using the plugin present in the input, Logstash can collect data from the source. Input plugin helps the Logstash to extract logs from various sources. Input plugin can be anything like beats, file, TCP, UDP, and many more. Using these plugins Logstash can read events on the source. This document focuses on two plugins they are UDP and File Plugin.

UDP: User Datagram Protocol is an IP protocol, used for connectionless communication protocol [2]. UDP is an unreliable packet delivery system. When logs are generated on the server, one can ship the logs to Logstash using the UDP plugin in the input session of the conf file. UDP uses the port number that should be mentioned in the input of the UDP plugin to establish a connection. Once the connection is established, it can read all the log events on the server.

Example of UDP plugin in Input session

```
input {
 udp {
 port => 6666
 codec => json
 }
}
```

File: This plugin ships the logs from the file to logstash. All the new events or logs that are appended to the end of the file can be captured by using the tail -0F. The file can be read from the beginning of the file and this plugin captures an event in a single line of text. If multiple events are logged on to the file ‘multiline codec’ is used inside the file plugin. Multiline codec identifies the new logs and processes the logs within the loop.

Example of File Plugin

```
input {
  file {
    path => "/var/log/file.log"
    type => JSON
    start_position => "beginning"
  }
}
```

2.2 Filter Plugin

This filter session is in between the input and output sessions of the conf file. In this session one can apply all the available filter plugins and filter the logs as per the requirements and then transfer them to the output plugin. To display the logs information more understandable manner in the elastic search, filter plugins are used as they parse the logs and transform the logs into a structure format. Some of the filter plugins are GROK, JSON, DATE, MUTATE, and many others. Using those filter plugins unstructured data can be transformed to structure data. This report will discuss more on JSON and mutate filter plugins.

Mutate: Mutate filter plugins allows general mutations on the log fields like rename, replace, add, lowercase, uppercase remove or modify the fields present in the logs. This plugin helps to enrich the log data. Using mutate filter, merging and joining of fields is possible.

Example of mutate filter plugin

```
filter
{
if 'unicast' in [message]
```

```
{
mutate{
add_tag => ["Unicast"]
convert =>{"port" =>"integer"
}
rename =>{"Inunicast => "packet type"
}
}
}
}
```

In the above mutate filter plugin, add tag will add a tag to message. Data type of a field can be set using convert. Rename is used to rename any field in the log message. Like wise many functions can be applied inside mutate filter plugin.

JSON: To process the JSON logs and get structure data, which can be easily read and analyzed from those logs JSON filter plugin is used. This filter plugin can parse and extract the JSON format. While parsing JSON data if the plugin fails to parse the data, the event will be untouched and displays with the message json_parse_failure.

Example of json filter plugin

```
filter {
    json {
      source => "message"
        target => "log"
    }
}
```

2.3 Output plugin

Once the logs are shipped and parsed in the input and filter plugin, using output plugin all the parsed logs are shipped to the destination. Logstash can send event data to multiple destinations at once like elastic search, CloudWatch, graphite, influxdb and many others. In this report all the logs are shipped to the elastic search. If one wants to build any visualizations with the log data on the Kibana then the log data should be transformed to elastic search. Kibana can use data present in the elastic search. Using stdout all the logs messages can be displayed on the console screen.

Example of elastic search output plugin and stdout

```
output {
        stdout { codec => json }
    elasticsearch {
      index => "%{test}-%{+YYYY.MM.dd}"
    }
  }
```

In the above example, the parsed logs are displayed on the console screen. Debugging of logs is possible by displaying logs on the console screen. Also, logs are shipped to the elastic search based on the mentioned index. In the above example test is the index to which data is sent. So using output plugin data can be sent to multiple destination at once.

Finally, all the three plugins input, filter and output are mentioned in the config file of logstash. Once the ELK services are up and running, need to run config file at the location /usr/bin/logstash. This is the default folder exists when Logstash is installed. All the conf files are saved at this location.

Kibana

3 Introduction

Kibana is a data visualization tool. It can access the data present in the Elastic Search. Huge volumes of logs data can be analysed in Kibana. Kibana supports various visualizations like line, graph, heatmap, bar chart and many other. On the server it runs on the port number 5601.Kibana can be accessed by using the URL **http://localhost:5601**. To access data on Kibana index must be created.

3.1 Creating Index

- Before creating the indexes, user must run the conf file in Logstash.
- To create index in Kibana, Go to Management tab.
- Click on Index patters under Kibana.

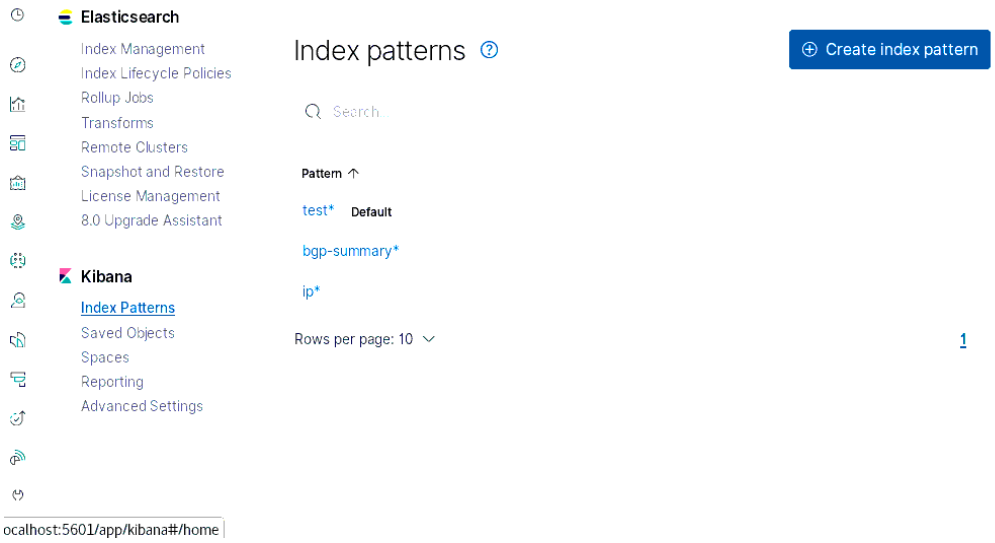


Figure 2. Index Patterns tab on Kibana

- The existing index information in the Logstash configuration file is displayed under patterns tab.
- User can select the index and click on next, finally user needs to select the time filter field and click on create index.
- The index is successfully created.

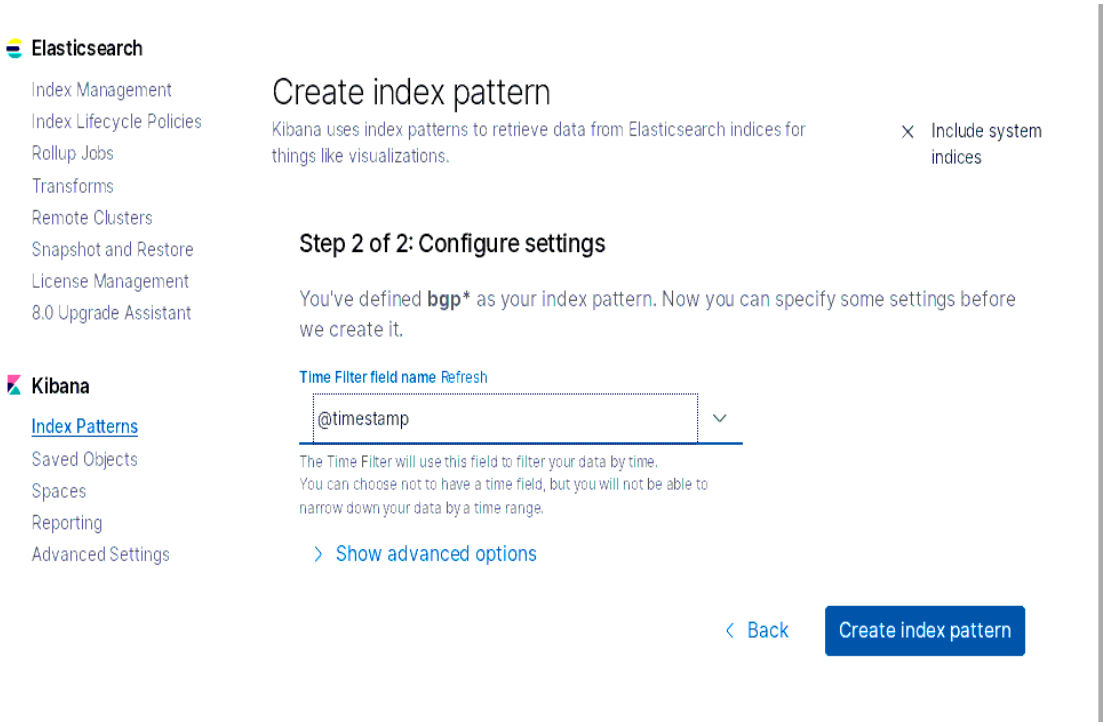


Figure 3. Select the time filter from the drop-down menu

- To access the created index, go to the discover tab and select the created index.
- The data sent to that index is displayed as shown in below imag

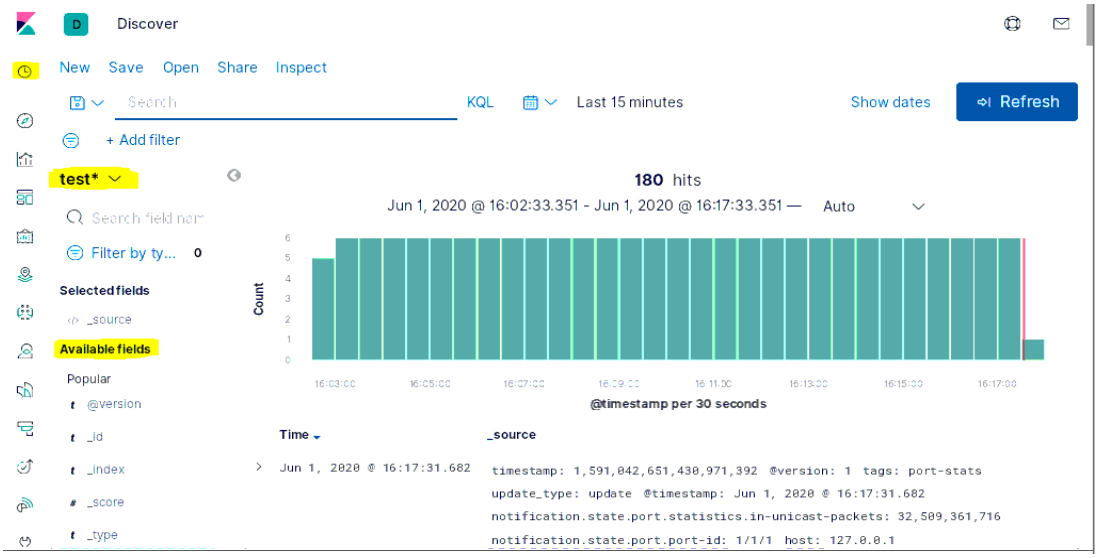


Figure 4. Selecting the Index on the discover tab of Kibana

3.2 Aggregations in Kibana

Kibana visualizations supports two types of aggregations they are

- Metric aggregations
- Bucket aggregations

Metric aggregation calculates the values for each of the bucket aggregation. Most of the metric aggregations are performed on the numeric value. Metric aggregation can be single values aggregations or multiple value aggregations. In single value aggregation the output will be single value for example count, max, average etc. In multiple value aggregation the resultant output will have multiple values example

Bucket aggregations are responsible for grouping documents together. Bucket aggregations can hold metric aggregations. There are several bucket aggregations they are histogram, date histogram, range, ipv4 range and many other. In the histogram, data can be grouped together based on the selected time interval. Likewise, in date histogram, based on date the data is grouped.

4 Installation of ELK on CentOS

CentOS is an Operating system of Linux distribution. Most of the organizations use this server for production servers. The installation is very simple with Yum.

Yum is an open source command line package management tool for RPM(RedHat Package Manager). The prerequisite for ELK installation is java.

To check java version open command line on centos

- Java -version

Import ELK repository GPG key.

- sudo rpm –import https://artifacts.elastic.co/GPG-KEY-elastisearch

Add ELK repository.

```
cat > /etc/yum.repos.d/elasticsearch.repo << EOF
> [elasticsearch-7.x]
> name=Elasticsearch repository for 7.x packages
> baseurl=https://artifacts.elastic.co/packages/7.x/yum
> gpgcheck=1
> gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
> enabled=1
> autorefresh=1
> type=rpm-md
> EOF
```

Install **Elastic search** using yum command.

- Sudo yum install -y elasticsearch

Enable and start Elastic search using below command

- systemctl daemon-reload
- systemctl enable elasticsearch.service
- systemctl start elasticsearch.service

Elastic search is now up and running at port tcp 9200.

Install **Logstash** using yum command.

- sudo yum -y install logstash

Enable and start Logstash search using below command

- systemctl daemon-reload
- systemctl enable logstash.service
- systemctl start logstash.service

Install **Kibana** using yum command.

- sudo yum -y install kibana

Enable and start Logstash search using below command

- systemctl daemon-reload
- systemctl enable kibana.service
- systemctl start kibana.service

5 Shipping the logs to ELK stack

In the above sessions we have learned about the functionality of three services Elastic search, Logstash, Kibana and installation of ELK stack. This session will provide more information about shipping the logs to ELK stack with sample example.

In Networking, the client server communication is established using gRPC protocol. The telemetry data is send to a UDP port using gRPC protocol. gNMI is a gRPC Network Management Interface used in enabling the telemetry data. It provides three types of services like ‘GET’, ‘SET’, ‘SUBSCRIBE’. SUBSCRIBE in turn provides three modes of data transmission they are ‘STREAM’, ‘ONCE’, ‘POLL’. In this example we use ‘SUBSCRIBE’ service in ‘STREAM’ mode to enable the logs. gNMI shell needs to be installed on the same machine where ELK stack is installed. By using below commands, we can send the Port stats to UDP port.

(grpc-shell) > gnmi_subscribe --name Test1 forward_stream --ip 127.0.0.1 --port 6666

- To subscribe the stream, use below command.

  (grpc-shell) > gnmi_subscribe --name Test1 --mode STREAM subscribe _state_port[port-id=1/1/1]_statistics_in-unicast-packets

- To execute the stream, use below command

  (grpc-shell) > gnmi_subscribe --name Test1 execute

5.1 Configuring and Executing the Logstash conf file

After enabling the streams, the next step is to configure the Logstash configuration file. Logstash configuration file can be created and saved at the location **/etc/logstash** [3]. The configuration files are saved with .conf extension.

Sample Logstash conf file

```
input{
            udp {
                    id =>"test"
                    host => "127.0.0.1"
                    port =>6666

                }
          }
filter{
                json {
                source => "message"

                }
          if "in-unicast-packets" in [message] {
            mutate { add_tag => "port-stats"
          remove_field => ["message"]    }
           }
           }
output {
                stdout{
                codec =>rubydebug
                }
```

```
        if "port-stats" in [tags]
        {     elasticsearch{
              hosts =>"127.0.0.1:9200"
              index =>"port-stats-%{+YYYY-MM-dd}"
              }
        }
}
```

Before executing the configuration file, the file must be tested to check if the created configuration file is free from errors. To test and execute the configuration file user must be on the home directory of Logstash **cd /usr/share/logstash**.

- Below command is used to test the configuration file.

  bin/logstash --path.settings /etc/logstash/ -f/etc/logstash/Name of configuration file with extension --config.test_and_exit.

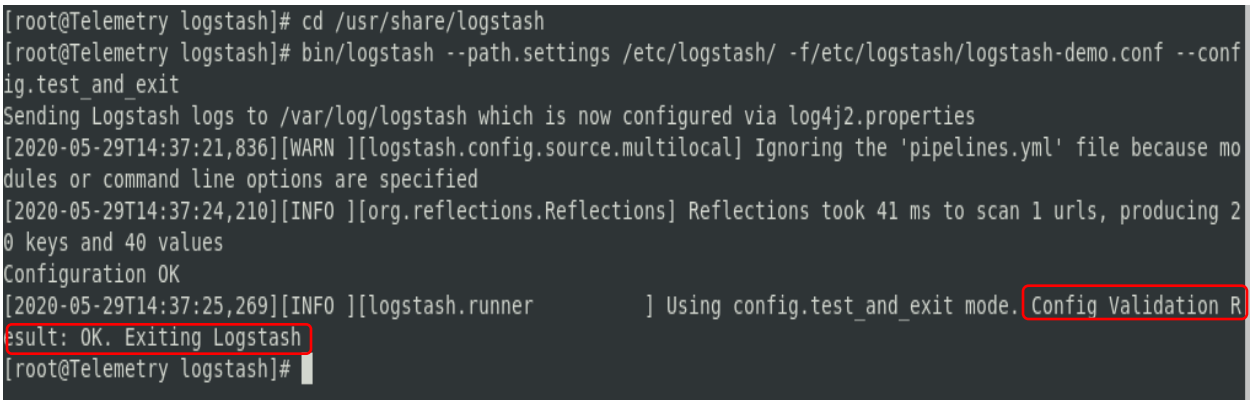


Figure 4. The Output after testing the configuration file

If the configuration file is free from errors, then 'Configuration Validation Result Ok' message is displayed as shown in the above image. If there are any errors error message along with the line number is displayed.

- Following command is used to execute the Logstash conf file.

bin/logstash --path.settings /etc/logstash/ -f/etc/logstash/Name of configuration file with extension --config.reload.automatic

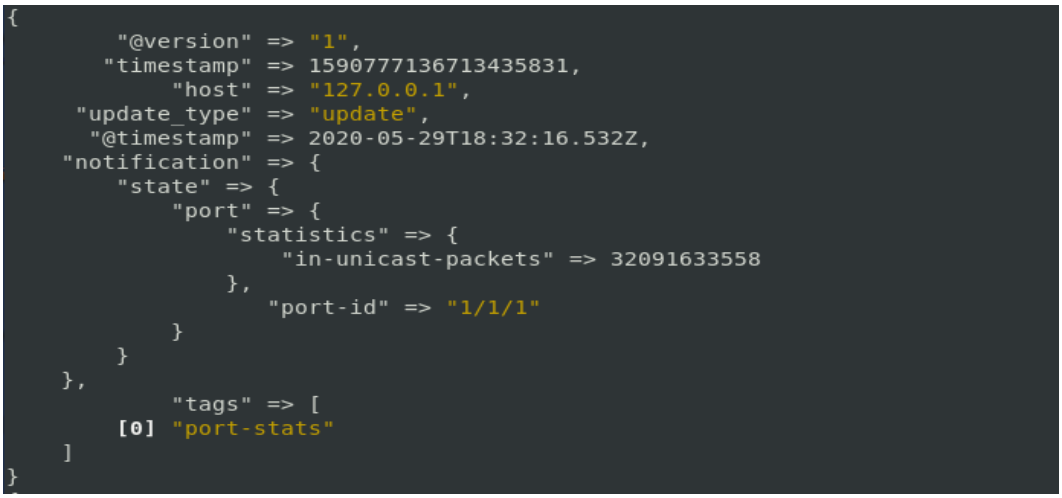


Figure 5. The output returned on the Logstash console screen

As logstash conf file hold stdout plugin, the output is displayed on the console screen before sending the logs to Elasticsearch.

5.2 Logs on Elastic search

Logs are shipped to Elastic Search based on the index mentioned in the Logstash configuration file. As per our configuration file the index mentioned is “port-stats”. The Elastic search should have an index with “port-stats”, if not create an index with the same index name mentioned in the configuration file. The index creation has been discussed in the section 3.1. If the index mentioned in the Logstash configuration file matches with the index created on the Elastic search, then the logs are shipped to that index. Likewise, Logstash can ship the data to multiple sources and multiple indexes on Elastic search. Elastic search does not have user interface. It uses the Kibana interface.

In general, the logs refresh every 30seconds on Elastic search. User can also change the refresh time by selecting the list from the drop-down menu available on the top right corner

of the discover tab. If the user selects the ‘Auto’ option from the drop-down menu, the logs will automatically refresh when ever the Logstash sends new logs to elastic search. The number of logs displayed on the discover tab can be controlled by selecting the show dates option available on the top right corner. Here user can select the range of logs by providing the date time range. On the left-hand side of the discover tab there are two sections one is available fields, selected fields. In the available fields section, the list of fields or columns available in the logs data is displayed here. If can select the required fields for his analysis purpose and the selected fields are shown in the “selected fields” section. Simultaneously, the data of selected fields are displayed in the table format at the center of the discover page.

5.3 Building visualizations on Kibana

The logs generated are timely in the Elastic Search, Time-series visualizations can be built on the Kibana. The complex data in a meaningful way can be displayed by combining multiple aggregations and metrics. Apart from the timeseries visualizations several interactive dashboards can be created. As per our logs that we shipped to Elastic search based on the data lets build few visualizations on the Kibana.

5.3.1 Creating Visualizations

Below are the steps to create visualization on the Kibana.

- Click on the visualizations tab present in the discover page. This will land on the homepage of visualizations.
- Now click on the ‘create visualization’ button present on the top left of the home page.
- A dialogue box with set of available visualizations are displayed.
- Select the visualization that we want to create as shown in the below figure.

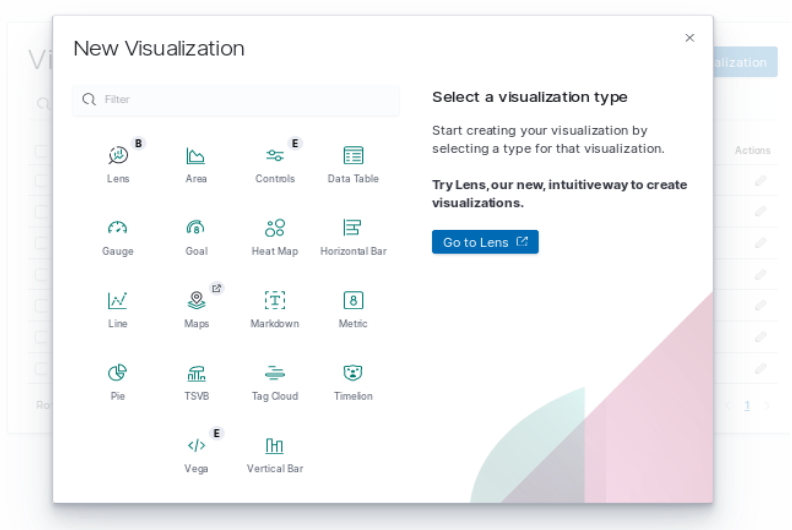


Figure 6. List of available visualization types

5.3.2 Line Graphs

- To create line graphs, click on the Line chart button from the list of available visualizations.
- Provide the name of the visualization.
- Select the index from which you want to analyse the data as index holds the data. Here we are selecting the ‘Port-stats’ index as we have our log data in that index.
- Now a page is launched where the visualization is displayed at the center and the metrics on the left side.
- As discussed in the section 3.2 there are two levels of aggregation for each visualization like Metrics and buckets.
- Click on the metric tab, now select the y axis and x axis that we want in the line chart.
- In buckets we can aggregate the data based on the distinct values available in the field.
- Click on the buckets, select the column. Select the split condition.

- Once the visualization is done click on the save button present on the top of the page.

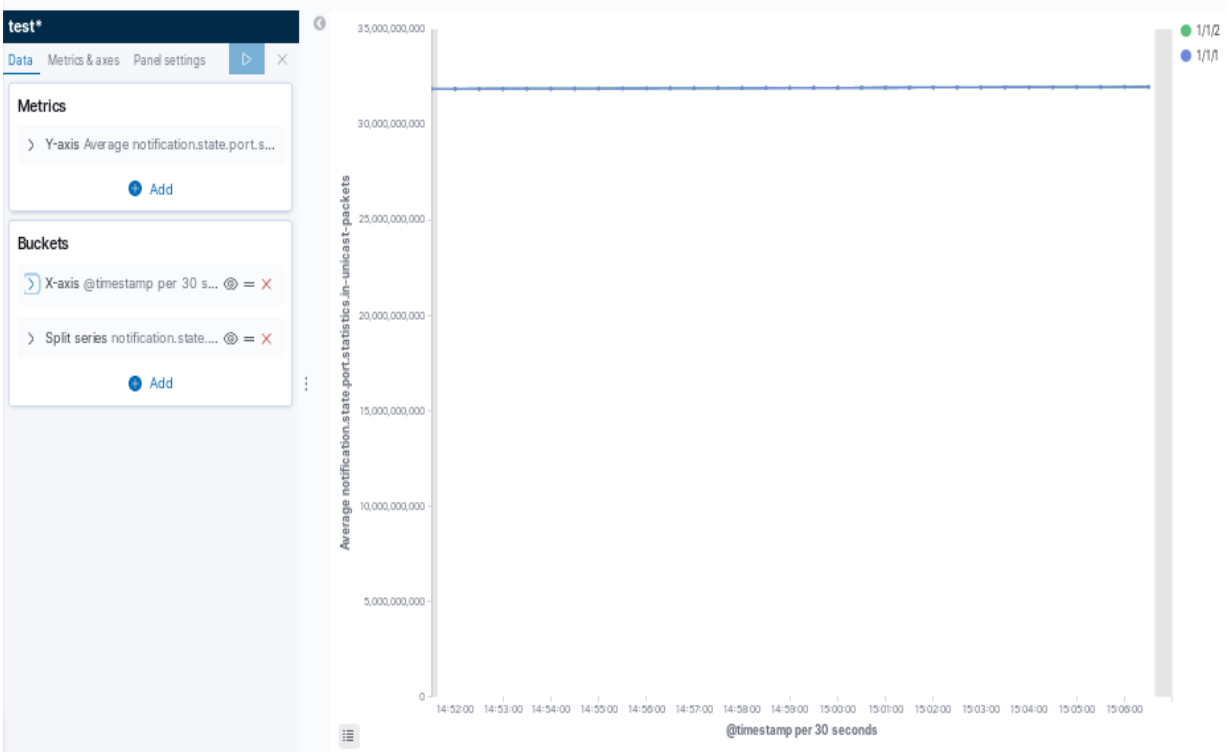


Figure 7. Representing the Bi-directional traffic of two ports on Line chart

In the above chart, we are displaying the average in-unicast-packets with respect to time and the time refreshes for every 30sec. so for every 30 sec there will be a change in the graph. The port stats 1/1/1 and port stats 1/1/2 are displayed. The reason behind displaying just one line instead two lines are the two lines data are overlapping that is they are bidirectional, and they hold same data. If we hover the mouse around the line, we can notice the difference.

5.3.3 Pie chart

For creating the pie chart.

- Click on the pie chart option present from the list of available options.
- Provide the name of the chart.
- Now select the Bucket aggregations, In this pie chart we want to show two distinct protocols namely BGP and local.
- Select the field based on which we want to split the series.
- When a mouse is hovered over a particular pie, it displays the information related to that pie.
- Save the visualization once done.

- This chart shows the relation between the ipv4 prefix and protocols. It represents the relation in the form of a pie and their individual percentage can be seen by hovering over the chart or by hovering over the prefix-IPs on the right hand of the chart.

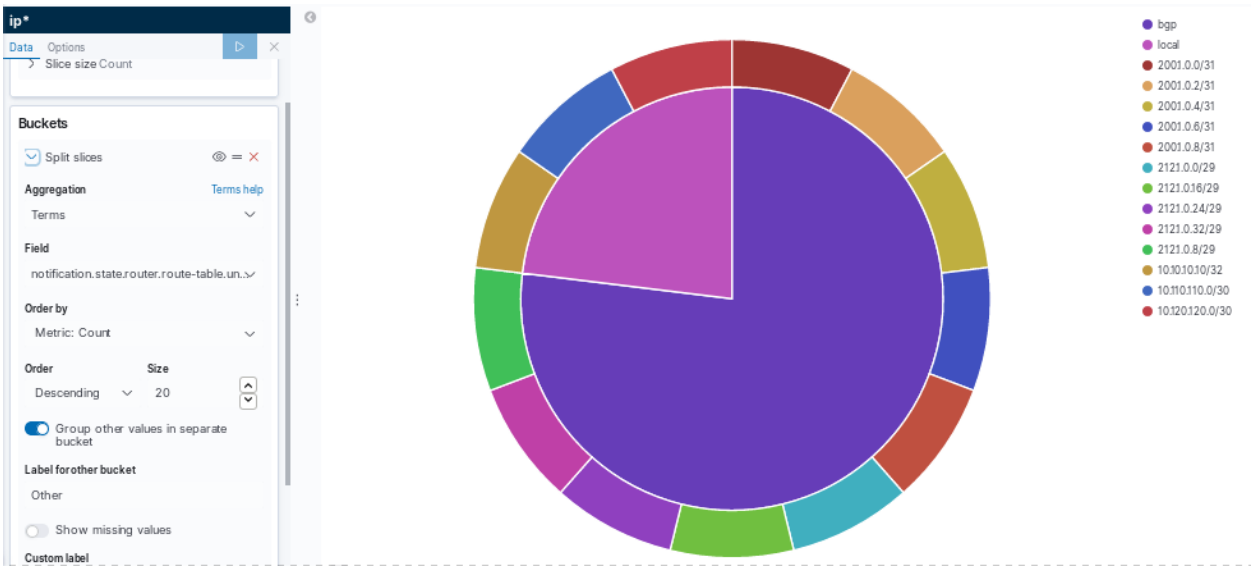


Figure 8. Protocol-ipv4-prefix Pie chart

6 Kibana Dashboards

- Once all the visualizations are created, we can add all the created visualizations to the Kibana dashboards. Below are the steps to create dashboards.
- Click on the dashboard panel present on the left side.
- This will launch the dashboard page. Click on the create dashboard.

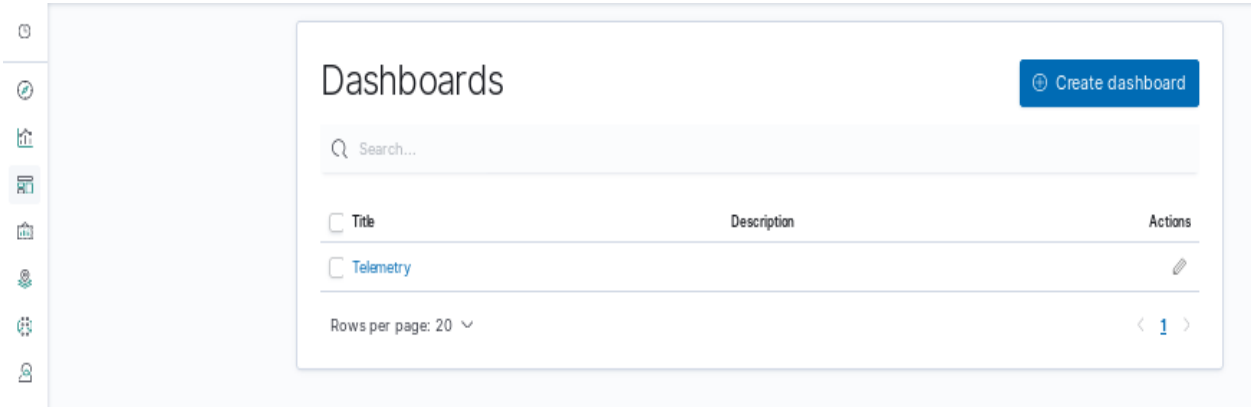


Figure 9. Dashboards creation page on Kibana

- Provide a name for the dashboard.
- To add visualization on to the dashboard, click on the add option on the top left of the screen and select the visualization that we want to add.
- To delete a selected visualization from the Dashboard, click on the Edit option, and then click on the gear button displayed on top right of that graph. Select ‘Delete from Dashboard’ option from the list.

- To save the dashboard click on the save button on the top of the page.

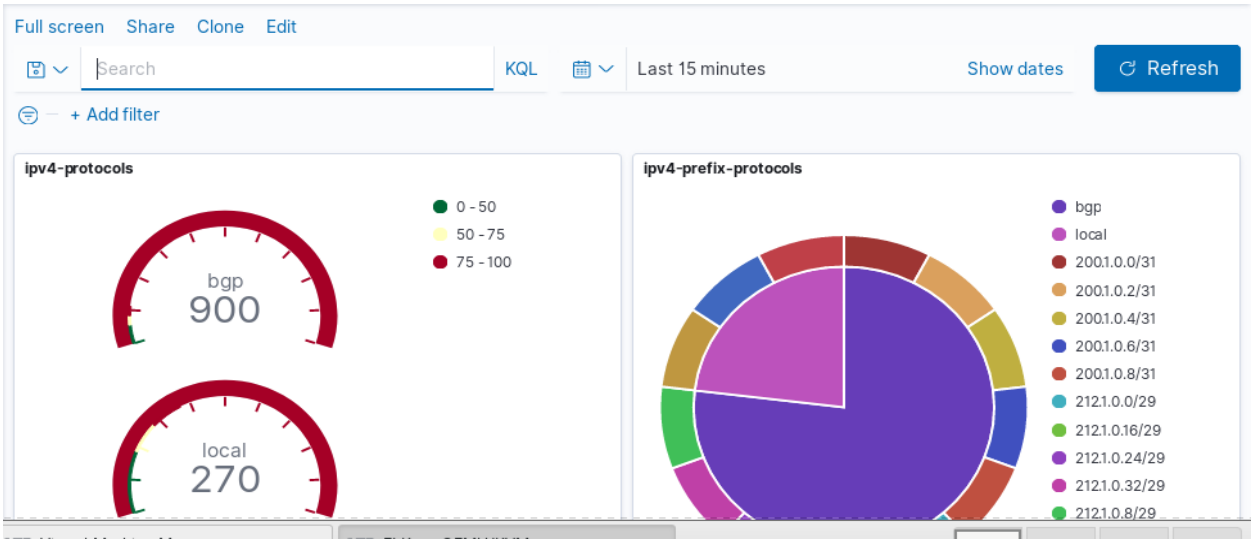


Figure 10. Final Dashboard page on Kibana

6.1 Deletion of saved objects in Kibana

Everything that we save on the Kibana is saved as objects [4]. The delete is bit different from the regular user interfaces that we use. For deleting any object in the Kibana user must go to the saved objects.

- To access the save objects, Management > Kibana > Saved Objects
- In the saved objects list of all the saved visualizations, index, tables are present here.
- The user can select the object to be deleted and click on the delete option.

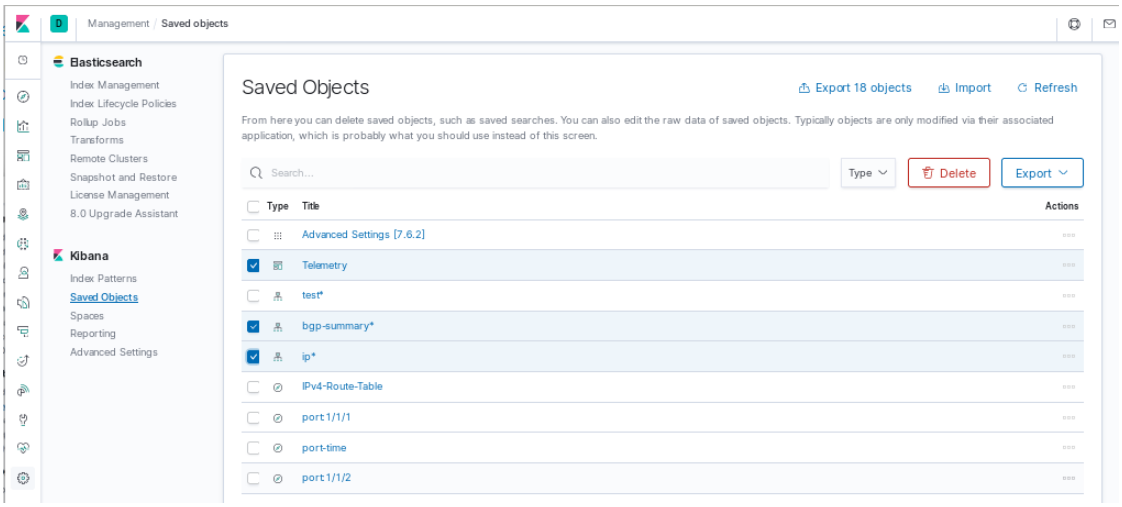


Figure 11. Object deletion on Kibana

7 Conclusion

ELK stack is the three open source tools. Elastic Search is an analytics search engine. Logstash can preprocess the data from source to multiple destinations. While Kibana is a visualization tools where interactive visualizations can be built. All the three tools together called as ELK stack. This stack is a powerful log analysis tool. Logs from the servers can be collected and stored in a centralized storing system. So that one can search all the logs at one place. By analysing the logs network engineer can be aware of server failures, traffic on the server, IP route table information and several other information can be analysed from the collected server logs. Apart from log analysis, some other applications are application monitoring, data integration tasks and data transformation task can be performed. As discussed in the report the Logstash can collect the logs or any other data from the specified source. In this project our source telemetry data, so Logstash collects the telemetry logs and ships to the Elastic search. Elastic search stores all the logs at the centralized location based on the index. Kibana uses the data present in the elastic search to build some analytics and visualization. As Kibana build analytics based on the real time data several network failures can be accessed and predicted.

References

[1] Elastic. 2020. *What Is Elasticsearch?* | *Elastic*. [online] Available at: https://www.elastic.co/what-is/elasticsearch.

[2] Elastic.co. 2020. *How Logstash Works* | *Logstash Reference [7.8]* | *Elastic*. [online] Available at: <https://www.elastic.co/guide/en/logstash/current/pipeline.html>.

[3] Elastic.co. 2020. *Logstash Configuration Examples* | *Logstash Reference [7.8]* | *Elastic*. [online] Available at: <https://www.elastic.co/guide/en/logstash/current/config-examples.html>

[4] Logz.io. 2020. *Creating The Perfect Kibana Dashboard* | *Logz.Io*. [online] Available at: <https://logz.io/blog/perfect-kibana-dashboard/>