Design and Performance of an Event Handling and Analysis Platform for vSGSN-MME event using the ELK stack

Sneha Adike
Vamshi Krishna, Bandari Swamy Devender

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden
This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Telecommunication Systems. The thesis is equivalent to 20 weeks of full time studies.

The Authors in this research paper grants to Blekinge Institute of technology non-exclusive right to publish the Work electronically and in a non-commercial purpose make it accessible on the Internet. The Authors warrants that the work does not contain any text, pictures, references and materials that violate the copyright laws.

Contact Information:
Author(s):
Sneha Adike
E-mail: snad17@student.bth.se

Vamshi Krishna, Bandari Swamy Devender
E-mail: vaba17@student.bth.se

External advisor:
• Josef Svenningsson X
  System Manager, Packet Core PDG, Ericsson

• Lars Nordlund
  Developer, PDG mobility MME, Ericsson

University advisor:
Prof. Kurt Tuthschku
Dept. Computer Science & Engineering
Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57
Abstract

Data Logging is the main activity to be considered in maintaining a server or database in working condition without any errors or failures. Data collection can be automatic, so, no human presence is necessary. To store the data of logs for many days and visualizing became a huge problem in recent days. Coming to node SGSN-MME, which is the main component of the GPRS network, which handles all packet switched data within the mobile operator’s network. A lot of log data is generated and stored in file systems on the redundant File Server Boards in SGSN-MME node. The evolution of the SGSN-MME is taking it from dedicated, purpose-built, hardware into virtual machines in the Cloud, where virtual file server boards fit very badly.

The purpose of this thesis is to give the better way to store the log data and add visualization using the ELK stack concept. Fetching useful information from logs is one of the most important part of this stack and is being done in Logstash using its grok filters and a set of input, filter and output plug-ins which helps to scale this functionality for taking various kinds of inputs (file, TCP, UDP, gemfire, stdin, UNIX, web sockets and even IRC and twitter and many more) , filter them using (groks, grep, date filters etc.) and finally write output to ElasticSearch. The Research Methodology involved in carrying out this thesis work is a Qualitative approach. A study is carried using the ELK concept with respect to Legacy approach in Ericsson company. A suitable approach and the better possible solution is given to the vSGSN-MME node to store the log data. Also to provide the ELK performance and uses multiple users of input providers and provides the analysis of the graphs from the results and analysis. To perform the tests accurately, readings are taken in defined failure scenarios. From the test cases, a plot is provided on the CPU load in vSGSN-MME which easily gives the suitable and best promising way.

Keywords: ELK stack, Filebeat, Logs, vSGSN-MME, X-Pack.
Acknowledgements

You have to do the research. If you don’t know about something, then you ask the right people who do.

Spike Lee

We firstly, would like to thank our parents and siblings for their continuous financial support and strong blessings to accomplish this thesis successfully. We sincerely express our gratitude towards Prof. Kurt Tutschku for all his support, suggestions, ideas and reviews. It’s an honour to work under his guidance. And we are pretty happy with his suggestions and continuous feedback throughout the thesis work.

We feel lucky to get Josef Svenningson and Lars Nordlund as our External Supervisors for our thesis from Ericsson and thank them heart-fully for their technical support, necessary guidance and feedback to every task and in documentation. The discussions during the meetings with them gave us a strong strength and pushed us towards the accomplishment of new tasks.

Finally, not the least, we thank Elastic team and all referees for their useful information and interesting insights in this research.
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANCB</td>
<td>Active NCB</td>
</tr>
<tr>
<td>AP</td>
<td>Application Processor</td>
</tr>
<tr>
<td>DP</td>
<td>Device Processor</td>
</tr>
<tr>
<td>ELK</td>
<td>Elasticsearch, Logstash, Kibana</td>
</tr>
<tr>
<td>EPG</td>
<td>Evolved Packet Gateway</td>
</tr>
<tr>
<td>ES</td>
<td>Elasticsearch</td>
</tr>
<tr>
<td>FSB</td>
<td>File Server Board</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communication</td>
</tr>
<tr>
<td>MME</td>
<td>Mobility Management Entity</td>
</tr>
<tr>
<td>NCB</td>
<td>Node Controller Board</td>
</tr>
<tr>
<td>NFS</td>
<td>Network File System</td>
</tr>
<tr>
<td>SGSN</td>
<td>Server GPRS Support Node</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>vDallas</td>
<td>Virtual Dallas</td>
</tr>
<tr>
<td>vIP</td>
<td>Virtual IP-address</td>
</tr>
</tbody>
</table>
List of Figures

1.1 GPRS network [1] .......................................................... 1
1.2 SGSN Component in Ericsson[2] ..................................... 2
1.3 ELK stack [3] ................................................................. 4
1.4 Elasticsearch Cluster [4] .................................................. 5
1.5 Basic Logstash Pipeline .................................................. 5
1.6 Kibana ........................................................................... 6

3.1 Environment setup .......................................................... 16
3.2 Expected Approach ........................................................ 18
3.3 Final Approach ............................................................... 18
3.4 Elastic Stack approach .................................................... 20
3.5 Installing Xpack ............................................................. 24
3.6 Users in Kibana Dashboard .............................................. 24
3.7 Roles in Kibana Dashboard .............................................. 25

4.1 Kibana dashboard after enabling x-pack .............................. 28
4.2 Sample figure for testing .................................................. 28

5.1 Behaviour of open source tools with one Filebeat Node ...... 30
5.2 Table for one User .......................................................... 30
5.3 Behaviour of open source tools with two Filebeat Nodes ... 31
5.4 Table for two Users ........................................................ 31
5.5 Performance of Filebeat varying number of Users ............ 32
5.6 Table for Filebeat Load ..................................................... 32
5.7 Performance of Elasticsearch varying number of Users ....... 33
5.8 Table for Elasticsearch Load ............................................. 33
5.9 Performance comparison of ELK and legacy method with 1 user node .......................................................... 34
5.10 Table for One User CPU load ......................................... 34
5.11 Performance comparison of ELK and Legacy method with 2 user nodes ......................................................... 35
5.12 Table for two Users CPU Load ....................................... 35
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Table for Installing and Configuring Xpack</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Differences between the Expected and Final Approach</td>
<td>19</td>
</tr>
<tr>
<td>5.1</td>
<td>Advantages and Disadvantages of Expected and Final Approaches</td>
<td>29</td>
</tr>
<tr>
<td>5.2</td>
<td>Advantages and Disadvantages of both modes</td>
<td>37</td>
</tr>
</tbody>
</table>
Abstract

Nomenclature

1 Introduction
  1.1 Concepts ................................ 1
    1.1.1 SGSN-MME ................................ 2
    1.1.2 Logs .................................... 2
    1.1.3 ELK stack ................................ 3
    1.1.4 Nginx ..................................... 8
    1.1.5 Beats ...................................... 8
    1.1.6 X-Pack .................................... 9
  1.2 Motivation/Problem ........................... 10
  1.3 Aim and Objectives ........................... 11
  1.4 Research Questions ........................... 11

2 Related Work

3 Methodology
  3.1 Environment setup .......................... 15
  3.2 Approach .................................... 17
    3.2.1 Expected Approach ........................ 17
    3.2.2 Final Approach ............................ 18
    3.2.3 Differences between Expected and Final Approach .... 19
  3.3 Installations .............................. 19
    3.3.1 Installing Java ............................ 19
    3.3.2 Installing ElasticSearch ................. 20
    3.3.3 Installing Kibana ........................ 20
    3.3.4 Installing Nginx .......................... 21
    3.3.5 Installing Logstash ........................ 21
    3.3.6 Installing Filebeat ....................... 21
  3.4 Configuration files .......................... 21
  3.5 SSL Certificates ............................ 23
  3.6 Security in Kibana .......................... 23
3.6.1 Limitations to security ........................................... 25

4 Validation and Testing .............................................. 27
  4.1 Validation of Security at Logstash and Filebeat ............... 27
     4.1.1 At Logstash server ..................................... 27
     4.1.2 At Filebeat server .................................... 27
  4.2 Validation of Security in Kibana ................................ 28

5 Results and Analysis .............................................. 29
  5.1 Advantages and Disadvantages of Expected and Final Approaches 29
  5.2 Behaviour of Elasticsearch, Logstash and Filebeat CPU Load varying users .................. 30
  5.3 Elasticsearch and Filebeat performance when the number of users is varied ....................... 32
  5.4 Comparison of ELK approach with Legacy method ............... 34
  5.5 Pulling Scenarios at Logstash in CloudStorage server ........... 36
     5.5.1 Advantages and Dis-advantages of Alternative method . . 37
  5.6 Confidentiality and Integrity .................................. 37
  5.7 Resource Consumption ......................................... 37
  5.8 Solutions for overcoming Failure Scenarios ...................... 39
     5.8.1 Faced Failure scenarios ................................ 39
     5.8.2 Overload Protection and Resilience Mechanism ............. 40

6 Conclusions and Future Work ...................................... 42
  6.1 Conclusion ...................................................... 42
  6.2 Research Question and Answers ................................ 42
  6.3 Future Work ..................................................... 44

References .......................................................... 45

A elasticsearch.yml ................................................. 50

B filebeat.yml ........................................................ 51

C Logstash.yml ........................................................ 53

D Kibana.yml .......................................................... 54

E Logstash config files .............................................. 55
  E.1 02-input.conf ................................................ 55
  E.2 20-filter.conf ............................................ 55
  E.3 30-output.conf ............................................ 55
Chapter 1

Introduction

In recent years, the usage of mobile devices which connect wireless and work remotely are increased rapidly. Nowadays millions of people are using wireless Internet through mobile devices and computers in their everyday life. This wireless connection is made possible by 2G, 3G and 4G technologies wireless networks.

When connecting to the Internet through 2G or 3G, all traffic will pass through a network called GPRS. The GPRS network is not only involved in Internet connections, but it is also involved in everything a mobile does except voice calls. For instance a mobile phone is browsing the web, firstly it connects to a cell tower, managed by a controller, then the signals/traffic from the mobile phone passes through the cell tower and its controller and then through the two GPRS support nodes, Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN) as shown in figure 1.1. Finally, the traffic is sent to the Internet from the GGSN node.

The logs which are received to SGSN-MME node from different users connected to this node are forwarded to the File Storage Board. FSB collects all the logs from the SGSN-MME node. The logs in FSB are in more number, so the requirement is to push the logs in FSB to different Server which can store logs and be visualized.

1.1 Concepts

This section will discuss the main components and the open source packages which are used in this research.
1.1.1 SGSN-MME

The Serving GPRS Support Node (SGSN) is a main component of the GPRS network, which handles all packet switched data within the network, e.g., the mobility management and authentication of the users. The MME or Mobility Management Entity represents the control-node for the LTE access-network or 4g. The combined SGSN-MME node[5] in wireless network handles a large number of users and has therefore stringent uptime requirements.

It supports modern mobile networks with new and enhanced functionalities for GSM, WCDMA, and LTE Accesses.

The SGSN node consists of a number of Plug-In Units (PIUs) as shown in figure1.2. Among these, Application Processors (AP’s) and Device Processors (DP’s) will function in sending and receiving the traffic via the node’s router. The Application Processors will handle the signaling, keeping track of which mobile devices are attached, where they are and which cell towers they are connected to. The Device Processors board will take care of the traffic, which is generated by the user while surfing the browser in a mobile phone. The user payload is redirected towards the GGSN and can be monitored in order to charge the user correctly, or when the phone company wants to control the Internet accessibility.

The SGSN node which in 4G migrated to a node called MME. Ericsson’s has for a long time developed an SGSN node which is widely used, and when 4G came along they put the SGSN and MME functionality into the same node, calling it SGSN-MME[5]. This node makes it possible to serve all these three generations of wireless networks through one and the same node.

1.1.2 Logs

A log file, in a computing context, is the automatically produced and time-stamped documentation of events relevant to a system. Virtually all software applications and system produce log files.

A log file is a documentation of the events that occur relevant to a system at a particular timestamp. Few examples of logs are accessed log, transaction log, and audit log. On a web server, an audit log lists all the files that users have requested from a website. From the server’s log files, an administrator can
get the information such as the number of visitors, number of times the page is requested, domain of the visitors, etc. In Microsoft Exchange, a transaction log is a record of all changes made to an Exchange database. Information to be added to a mailbox database is first written to an exchange transaction log and then the contents of the transaction log are written to the Exchange Server database. An audit log records chronological documentation of the activities such as the resources accessed, addresses of destination and source, time-stamp, etc. that could have affected a particular operation or event[6]. Every activity in the system including the restart, a shutdown is documented as a log file. Logs[7] can be categorized as follows:

- **Default logs**: These are generated by default when the execution of a project starts and ends, when an error occurs and the execution stops and when the logging settings are configured to log the execution of every activity. The events logged in this category are execution start, execution end, transaction start, transaction end, error log and debugging log.
- **User-defined logs**: They are generated according to the process designed by the user.

There are some logs which can be discarded in a few days and some which must be kept through years. To stop the logs from growing enormously we can use ‘Log Rotation’. Log Rotation provides facilities to rotate, compress, remove and alternatively mail logs thereby easing the administration of systems that generate a large number of logs[8]. The time to rotate the logs can be specified in the log-rotate configuration file according to the user’s requirement.

The logs which are generated in the SGSN-MME node from different users connected to that node are pushed and stored in FSB. The FSB will collect the logs from a different number of SGSN-MME nodes and stores them. The logs are in different format and type. They are like interface logs, fault management logs, alarm logs, etc., in the path /export/logs/.

Log data provides information for identifying and troubleshooting equipment problems[9], configuration errors, and hardware failures. Logs contain a record of all transactions. Hence logging is critical in any system.

### 1.1.3 ELK stack

ELK is the abbreviation of Elasticsearch, Logstash and Kibana open source projects as in figure1.3. This all are combined together to view parse, store, search and analyze engine. This is more applicable now in visualizing logs in real time. The logs/documents from a client are parsed to stack in sever called Elasticsearch by Logstash[10]. Then the Kibana will visualize the logs/document present inside the Elasticsearch.
Elastic Stack provides centralized logging and allows searching through all logs at a central place. The Elastic Stack is a versatile collection of open-source software tools that are implemented based on a distributed log collector approach that makes gathering insights from data easier[11]. All these concepts about open source packages are discussed clearly in the below sections.

Elasticsearch

Elasticsearch is an open-source package which is highly scalable full-text and analytics engine. It is one of the better packages to store, search, analyze the data of small and big volumes of data and in real time[12]. It has many concepts which are used to build Elasticsearch strongly. They are:

- **Clusters**: It is a collection of one or more nodes(servers) which are running the ELK software components. All these nodes will hold the entire data and provides federated indexing. The logs are stored in different nodes and search capability is across all nodes and makes simple to output the requested document. A cluster is identified by a unique name which shouldn’t be matched with other cluster names.
- **Node**: It is a single server that is with Elasticsearch package installed into it. This server is a part of a cluster with a unique identifier name inside the cluster. This node also accesses the user to store, search and analyze the documents.
- **Index**: It is a collection of documents(logs) that have somewhat similar characteristics. In a single cluster, many indexes can be declared and be defined.
- **Document**: It is a basic unit in Node which is information that can be indexed. Many documents can be stored within an index/type. Although a document physically resides in an index, a document actually must be indexed/assigned to a type inside an index.
- **Shards and Replicas**: Elasticsearch has the best facility to subdivide the index into multiple pieces into shards. When there is huge data, this facility acts more efficiently. When an index is created, simply number of shards can be defined and each shard is in itself a fully-functional and independent
"index" that can be in any node inside the Elasticsearch cluster as shown in figure 1.4. The shards will have both Primary and Replicas. The primary will give the data when it is active and if the shard fails, the replica of that shard will provide the data to the user inside the cluster.

![Elasticsearch Cluster](image)

**Figure 1.4: Elasticsearch Cluster [4]**

**Logstash**

Logstash is an open source which is used as a data collection engine [13] with real-time pipelining capabilities. It is a server-side data processing pipeline that ingests data from a multi of sources simultaneously, normalizes it, and then sends it to the “stash” called Elasticsearch.

Logstash can be configured in many ways to handle all sorts of log formats. As data travels from source to stash, this package will filter the log entries and parses each event, identifies name fields to build the structure, and transforms them into normalizing format [14].

By enabling HTTP input plug-in Logstash can receive single or multiline events over HTTP(s) [15]. The events from Applications can send them by HTTP post request with a body to the endpoint started by this input. Then Logstash will convert the incoming event and parses them in a normalized way with all the other events. There are also other usages with this plug-in like receiving web-hook requests to integrate with other services and applications. This plug-in supports standard HTTP basic authentication headers to identify the requester [16]. A set of User name and Pass-
word is attached while sending data to Logstash. Setting up an SSL is another option to send data securely over HTTP’s.

Kibana

Kibana is an open source platform that offers visualization feature[17]. Here, it is associated to work with Elasticsearch. With the help of Kibana, the user can view, search, analyze and interact with the data in Elasticsearch. It is easy to deal with huge data with the help of Kibana because Kibana offers features of visualization like graphs, tables, charts, etc. using which we can view and understand data in an abstract form.

Kibana consists of a side navigation bar with various sections like Discover, Visualize, Dashboard, Timelion, Management, Dev Tools[18]. All these tools are explained in below section. Primarily, we need to specify our index type in Time-line section and create an index pattern. In our case, the index pattern we need to create is Filebeat.

- **Discover**: Then if we check the Discover application we can view the histograms of the data in Elasticsearch and the data itself in the data table below the histograms. We also have a time picker where we can select the range of time in which we want to view the data and also select refresh time for Kibana[19]. There is a search bar for any queries when needed. Any individual data inside the Elasticsearch can be visualized from this dashboard
Chapter 1. Introduction

- **Visualize**: Visualize application enables the visualization of data entering into the stash. It has many options to visualize the data like Line, Area and bar chats[20].

  Time Series Visual Builder is a time series data visualizer with an emphasis on allowing you to use the full power of Elasticsearch aggregation framework. Time Series Visual Builder allows you to combine an infinite number of aggregations and pipeline aggregations to display complex data in a meaningful way[21].

- **Dashboard**: It displays the collection of searches and visualizations of every individual data. A dashboard can be saved with the options arrange, resize and edit. The saved dashboard can be shared[22]. If the visualization option is enabled in Kibana then the raw data under the visualization will be displayed using Expand/Collapse button.

- **Timelion**: Timelion application is a time series data visualizer. It enables the all independent data sources to combine totally in a single visualization[23]. It’s used to retrieve the number of data sources entered into the Elasticsearch at a specific time defined in a timelion section above the dashboard. This section can solve many complex questions like Number of data sources from different geographical locations, the particular location of majority data sources among all data, etc.

- **Dev Tools**: This page contains development tools which are used to interact with the data in the Kibana. This application has Console plugin which provides a User Interface with the Rest API of Elasticsearch. It has two main areas, Editor pane and response pane[24]. In Editor pane, requests are done to Elasticsearch like search, edit, read, write, delete, etc. and in response pane, will display the output/responses to the request made in Editor pane.

- **Management**: This application is used to perform the runtime configurations of Kibana, including both the initial setup and ongoing configurations of index patterns. It also performs the advanced settings that tweak the Kibana behavior itself and the saved objects like searches, visualizations, and dashboards[25]. The saved searches, visualizations and dashboards can be viewed, edited, deleted, exported and imported from the management settings.

- **Monitoring**: This application is enabled with the X-pack plugin (clearly discussed in sec 1.1.5). This application enables a user to monitor the Elastic Stack from Kibana like viewing health and performance data for Elasticsearch, Logstash, and Kibana in real time, as well as analyze past performance[26].

In the Visualize section, we can visualize the data in the form of pie charts, tables, graphs, etc. The dashboard displays the collection of all visualizations and searches. The Management section allows us to perform a runtime configuration
of Kibana. This section is modified when there are any plugins attached to Kibana like X-Pack (It is explained in section 1.1.5). When there are plugins Management section provides additional features. Dev Tools is a console where we can issue any queries.

### 1.1.4 Nginx

Nginx is an Open-source free web server. It is also used for high-performance HTTP serving\(^{[27]}\), reverse proxy, media streaming, TLS/SSL with SNI and more. It started out as a web server designed for maximum performance and stability. In addition to its HTTP server capabilities, NGINX can also function as a proxy server for email (IMAP, POP3, and SMTP) and a reverse proxy and load balancer for HTTP, TCP, and UDP servers.

Nginx is built to offer low memory usage and high concurrency. Nginx uses an asynchronous event-driven approach to create a web request in a single thread rather than creating the new process. Nginx will have one master process which will control multiple work processes and maintains them. The worker process will do the actual processes. As Nginx is asynchronous, each request can be executed by the worker concurrently without blocking other requests.

As this research is going to use only one server for installing Elasticsearch and Kibana, we have to use and set up a reverse proxy to allow external access to Kibana. So, we are preferred to use Nginx as it is lightweight and efficient load balancer compared to Apache web server\(^{[28]}\). After the Nginx configuration the HTTP traffic into this server is directed to the Kibana.

### 1.1.5 Beats

Beats are light-weight, single-purpose data shippers that ship data from various machines. These beats are mostly used along with Logstash and Elasticsearch\(^{[29]}\). The data from beats are shipped to Logstash or Elasticsearch. Beats serve a great purpose to gather data and are usually installed on servers. They centralize data in Elasticsearch and can also ship data to Logstash for indexing and parsing. Beats include modules that simplify collecting, parsing and visualizing common log formats.

There are many types of beats\(^{[30]}\) like Metricbeat, Packetbeat, Winlogbeat, Auditbeat, Filebeat, Heartbeat, etc. Coming to some beats description Metricbeat is a lightweight shipper for metrics such as CPU load, memory, etc. Packetbeat is a lightweight shipper for Network Data that ship information about transactions between servers. Winlogbeat is light-weight shipper that ships windows event logs. Auditbeat collects audit data and ships these events to the Elastic Stack for further analysis. Filebeat is Light-weight shipper for logs. This research is going to use Filebeat among all beats because of its main purpose and as it is better among all beats for shipping logs.
Chapter 1. Introduction

Filebeat

Filebeat is a log data shipper for local files. It monitors the locations specified by the user and forwards them either to Elasticsearch or Logstash or Kafka or Redis for indexing/storing depending on configuration. It reads and forwards logs line by line when it is interrupted it remembers the exact position of where it was interrupted and starts sending the logs again from that particular position. Filebeat uses back-pressure-sensitive protocol when sending logs to Logstash or Elasticsearch. It means that whenever Logstash is busy with the log files in it, Logstash sends a signal to Filebeat to slow down its process and Filebeat slows down the shipping of logs until Logstash resolves its issue. Once Logstash is free to receive new log files Filebeat continues its shipping process with the previous speed. Filebeat has no run-time dependencies.

1.1.6 X-Pack

X-Pack is a Plug-in extension to Elastic Stack that provides various features like securing the data, monitoring the data, detecting unusual changes in the data, reporting the data, alerting, graphs, etc. into one easy-to-install package. Prior to Elasticsearch-5.0.0, we had to install Shield, Watcher, Marvel to get all the features that are now available all together in X-Pack. Tools like SearchGuard, Elastalert, Kibi, Skedler, etc. can be used as alternatives to X-pack. But these act as an alternative to X-Pack only in some particular features. For example, Search Guard, Elastalert are alternatives to X-Pack only in terms of the Security feature. Skedlar offers only monitoring feature and Kibi offers reporting and graphical features. Thus, we can say that X-Pack is superior to other tools because it is everything including support bundled into one single package. The table 1.1 gives the command to the Elasticsearch.yml and Logstash.yml files to enable or disable the respective feature.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine Learning</td>
<td>xpack.m1.enabled</td>
<td>Set to false to disable X-Pack machine learning features</td>
</tr>
<tr>
<td>Monitoring</td>
<td>xpack.monitoring.enabled</td>
<td>Set to false to disable Kibana’s monitoring features</td>
</tr>
<tr>
<td>Security</td>
<td>xpack.security.enabled</td>
<td>Set to false to disable X-Pack security features</td>
</tr>
<tr>
<td>Graph</td>
<td>xpack.graph.enabled</td>
<td>Set to disable X-Pack graph features</td>
</tr>
</tbody>
</table>

Table 1.1: Table for Installing and Configuring Xpack
• **X-Pack Security**: Formerly this feature was offered by Shield. It provides the facility where only authenticated users can access the documents or fields of the documents that they have permission to access[33]. The users of Elastic Stack can rest knowing that the data is safe and secure. X-Pack security supports configuring SSL/TLS for securing the communication channels too, from and within the node.

• **X-Pack Alerting**: Formerly this feature was offered by Watcher. When any unwanted change occurs like unexpected increase of CPU load, the increase of response time, fall in the indexing rate of Elasticsearch, etc. this feature alerts us before anyone knows about this changes[34].

• **X-Pack Monitoring**: Formerly this feature was offered by Marvel. This enables the user to monitor the Elastic Stack from Kibana like viewing health and performance data for Elasticsearch, Logstash, and Kibana in real time, as well as analyze past performance[35].

• **X-Pack Reporting**: It helps to generate reports on Kibana dashboards, visualizations; schedule the reports and also emailing them to customers[36]. Reporting operates by creating and updating documents in Elasticsearch in response to user actions in Kibana[37].

• **X-Pack Graph**: It helps us to analyze the relationships among the data of the Elastic Stack.

• **X-Pack Machine Learning**: It helps in modeling the behavior of Elastic Stack, identifying issues faster, reduce false positives[38].

While X-Pack by default provides all these features users have the ability to enable or disable its features as per their requirement. In our Thesis we only use X-Pack to provide security for logs in Elasticsearch, so we can enable only that feature of X-Pack. X-Pack benefits include making the creating and emailing of automated reports. Xpack does not encrypt the actual log instead it benefits in Authentication, Authorization, Encryption with SSL/TLS encryption, Layered security, Auto Logging, and Compliance[39].

### 1.2 Motivation/Problem

In present-day storing data in Cloud is significant. Everyone is using many products and the data produced from them is of in petabytes and zettabytes. So then there will be a huge amount of log generation in the system. Finding the errors and problems using log data is a very risky job to do in the present day scenario. And also focusing on each log is a terrible task and waste of energy. So, to overcome this Elasticsearch is a better and faster solution in indexing compared to Solr in the most better search engine open sources[40].

So, this is one scenario which motivated me to use Elastic search in this research. The problem in the SGSN-MME is that the logs generated are stored
in virtual file server board which fit very badly with a huge amount of logs in
a short period of time. So to overcome this problem we used Elasticsearch with
SGSN-MME logs.

1.3 Aim and Objectives

The aims and objectives of this thesis are:

- **Aims:**
  - Determining the efficient way to store logs by considering speed, disk
    usage, time parameters.
  - Decide whether the pulling mode at Filebeat is better option.
  - To overcome failure scenarios when the elastic search is down or too
    slow.
  - To restrict the access of logs only to the users associated to work with
    those logs

- **Objectives:**
  - Designing the set up for ELK server and Filebeat client.
  - Establishing the tunnel between CloudStorage (ELK server) and Node
    control board (Filebeat client) inside the Design host (explained clearly
    in section 3.1).
  - Installing Elasticsearch, Logstash, Kibana, Nginx and Filebeat.
  - Configuring Elasticsearch, Logstash, Kibana, Nginx and Filebeat.
  - Sending logs to Logstash through Filebeat.
  - Storing logs in Elasticsearch.
  - Accessing logs through Kibana.
  - Providing external access to Kibana through Nginx.
  - Adding Security to the logs coming from FSB using X-pack plug-in.
  - Creating users and roles for the logs inside the Kibana.

1.4 Research Questions

This research focuses on the following research questions:

- How efficient is the ELK concept for appropriately handling of log in the
  SGSN-MME such that CPU load and resource consumption’s are mini-
  mized, and reliability increased?
- How many events can be handled by the ELK platform considering different
  modes? How does the Elasticsearch platform scale its performance when
  the number of user/connections handled at the SGSN is increased?
Chapter 1. Introduction

- Would there be an advantage if an alternative data transfer mode to the Filebeat is applied?
- What kind of overload protection and resilience mechanisms are appropriate for the Elasticsearch mechanism?
- How the Confidentiality and Integrity of data and system are handled for logs in Elasticsearch?
Chapter 2

Related Work

Research is to see what everybody else has seen, and to think what nobody else has thought.

Albert Szent-Gyorgyi

Multi-level Log management has been investigated in [41] and the conclusion was cloud computing services which are associated with many users, logs of huge size are generated by various functionalities like transactions between systems, user information update, huge data processing, etc. Therefore, it is very difficult to analyze services using logs when there is an emergency. To make analyzing log better, a method was proposed that divides log priority according to the security level. The auditing priority of logs is decided by the level of users. The logs generated by the user who has the highest level are audited with top priority. And logs generated by users with low priority are audited at last. This method can efficiently cope with potential attacks from dangerous users. From this paper, the observation towards this research is adding security to the CloudStorage virtual machine from other attacks towards the log messages. This issue is considered in this research and solved using the X-pack package which enables security to the logs inside the Elasticsearch.

System logs like the Windows Event log or the Linux system logs are investigated from [42], which discuss the important resource for computer system management. To rank these system log messages by their estimated value to users, and generating a log view that displays the most important messages, a method was invented. The ranking process is done by sampling a population of computer systems to score messages. For efficient scoring unsupervised clustering is used to identify system sets that behave similarly. The difference in the ranking of messages by frequency is measured by a construction scheme which shows that it leads to better clustering results. The distribution of messages in a given system is estimated using the resulting clusters, and log messages are scored using this estimation. From this paper, the observation towards this research is considering the prioritizing the incoming logs from the FSB. So, as the research is going on, adding a field "tags" solved the issue faced in this related paper by
Chapter 2. Related Work

creating a separate index with the there respective IP address.

How reliable and tamper-resistant centralized logging can be achieved in Ericsson’s SGSN-MME product is studied from the research [43]. This research is discussed on proving security to the log entries is the main concept and the theme of this project. For the reliability and secure transmission of log entries syslog-ng and r-syslog entries are used in the SGSN-MME node. It carried a seven test with failure scenarios and there overcome for the reliability. The test results showed the possibility to zero loss transmission for the most test cases in both Syslog-ng and rsyslog. For this achievement, it used application based transport protocol and reliable disk buffer. For tamper protection, it used a cryptographic hardware device for encryption key storage.

As there is slow internal routing and problems faced at the SGSN node, this report[44] is focused to optimize the above problem. The Author, Svensson focused on SGSN node internal routing protocols to improve the routing process in terms of speed and packet loss, memory space and internal network traffic. The packet loss and backplane traffic part of the solution is implemented by the source code and tested in several conditions. The suggested solution is implemented in the programming language C. All implementation is on top of the original source code. The test results from the proposed solution show a much smaller time window during which packets sent will be lost. This resulted in fewer packets lost.

The importance of shard selection during the query from kibana into the Elasticsearch is investigated from the research [45]. This research gave a solution using "SAFE" plug-in to Elasticsearch for the shard selection during the query. The results indicate the three algorithms of SAFE performed when documents are distributed to shards depending on a linguistic topic they belong to. SAFE seems unusable and doesn’t show the result when Elasticsearch is allocated documents randomly to the shards. This paper gives the idea of how Elasticsearch is allocating the documents and sharing the indexes using Lucene.
3.1 Environment setup

Firstly, an environment called design host which is a Dell Server is set up in the cloud as shown in figure 3.1, where we can install the virtual machines, boards, into it. This design host is having enough RAM and CPU to run a few small setups of the vSGSN-MME plus the vDallas traffic generator. The design host is installed with two GPB’s, two NCB’s and one FSB which is a part of the vSGSN-MME. A vDallas(virtual Dallas) is also installed into this design host which generates the traffic(logs inside the vSGSN-MME). We also installed vSGSN-MME node into this environment. This setup is created twice with two users(User1, User2). NCB’s are symlinked to FSB. And now finally, we installed a Ubuntu-16.04 Virtual Machine of RAM 8GB and named it as Cloud-Storage to run the ELK packages and run this experiment.

The arrows in the figure 3.1 describes the ssh login from one node to other. When we enter the design host, firstly we can log in into the Cloudstorage virtual machine or FSB(that is shown with arrows). The line below all nodes which is started at FSB is a backplane where all the machines and boards are connected from FSB towards the other nodes. The points B, C, D, and E are the points where FSB(A) tries to connect them through ssh.
Chapter 3. Methodology

Below are the points which clearly describe the importance of installing machines and boards in the environment setup and their usage in this setup.

- The GPB’s are the machines which collect the logs coming from different servers through the vSGSN-MME node. This GPB will process the logs to NCB through the Application Processor (AP) inside the GPB.
- The NCB’s are the boards which collect the logs from two configured GPB’s and parse into the FSB. There are two NCB’s to handle redundancy. The passive will take over in case the active fails for some reason.
- vDallas is installed to generate the logs or to increase the number of logs. The frequency of logs can be changed for the log generation inside the vDallas.
- The purpose of the FSB is to serve a file system to all other boards in the SGSN-MME, which stores all the logs coming from all boards inside the vSGSN-MME node to a directory /export/Core/sysadm/tmp/logs. This logs are divided into different sub-folders according to their field and stored in the path logs/*/tmp/* (* indicates field name). These logs are accessed only with user rights.
- Cloud-storage is a Ubuntu-16.04 LTS virtual machine which is installed to access logs from FSB. This Virtual Machine is used to test the ELK stack approach and do the remaining stuff in this thesis. To this virtual machine, two users are created to work differently in the environment. The packages which are required for the ELK stack approach are installed in the Cloud-Storage.

Before installing Cloud-Storage, all the boards are bridged under one link with a common interface. When the new machines(Cloud-Storage) are installed into this design host, they are attached to this bridged interface to connect internally. As this boards are virtual will not have Interface with the Internet. Only
Chapter 3. Methodology

the Cloud-Storage have Internet access with a proxy server internally from the Ericsson. SSH-key is the option which is used to log into every board. In this research, the installation of the package inside the boards is done by installing the package in Cloud-Storage and then secured copy to that respective board using scp command. This research used only the Debian packages because of it more flexible and can extract easily compared to other packages.

3.2 Approach

This section will describe the Expected approach and Final approach which are considered in research.

3.2.1 Expected Approach

This approach 3.2 is what we decided to work on. Where we installed the Filebeat package in ANCB node and ELK packages in Cloud-Storage nodes. To ship logs between the ANCB to Cloud-Storage through FSB, an SSH tunnel is created twice between ANCB-FSB and FSB-Cloud-Storage. The Filebeat Debian package is installed in Cloud-Storage server and secured copied to FSB and then to ANCB. The tunnel between this nodes is done using port 5044.

But, the problem issued here is, the logs which approached ANCB from GPB’s cannot be shifted to Cloud-Storage through FSB because of network configurations and reverse keying into ANCB to FSB. As this environment is virtual software, and there are no extra bridges, it is failed to parse the logs to Cloud-Storage. The solution with SSH is chosen for simplicity, compared to opening up a path in the rather complicated internal networking of the SGSN-MME. The Advantages and Dis-advantages of this approached are clearly discussed in the section 5.1.
3.2.2 Final Approach

Figure 3.3 is the approach, where we worked on and succeeded in parsing logs. In this approach, the Filebeat package is installed in FSB rather than in ANCB.
This package is installed in Cloud-Storage and secure copied to FSB. The ELK-stack packages are installed in Cloud-Storage machine as we done in Expected approach.

After installing the packages, configuration changes are done which are discussed in the section 3.4 and created SSL certificates for strong authentication which also discussed in the section 3.5. A tunnel is created between the Cloud-Storage and FSB. At FSB and Cloud-Storage servers port 5044 is used and configured in Logstash at Cloud-Storage and in Filebeat in FSB. This ports play a vital role in configuration files and parsing logs.

In Cloud-Storage, the logs from FSB are parsed to Logstash and stored in ElasticSearch which is parsed using port 9200 internally. Then the Kibana is viewed using port 5601 with the help of reverse proxy server Nginx.

### 3.2.3 Differences between Expected and Final Approach

This subsection will give the failure and success-ed scenarios faced in both approaches and it also provides the differences between them.

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Expected Approach</th>
<th>Final approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Failed scenario</td>
<td>Success-ed scenario</td>
</tr>
<tr>
<td>2</td>
<td>2 SSH Tunnels are required.</td>
<td>Only 1 SSH tunnel is required.</td>
</tr>
<tr>
<td>3</td>
<td>Longer path to parse logs.</td>
<td>Shorter path of parsing logs.</td>
</tr>
<tr>
<td>4</td>
<td>Takes more time to parse the logs.</td>
<td>Takes less time to parse logs.</td>
</tr>
<tr>
<td>5</td>
<td>2 boards(ANCB and FSB) are used.</td>
<td>Only 1(FSB) board is used.</td>
</tr>
<tr>
<td>6</td>
<td>Keying is not possible from ANCB to FSB.</td>
<td>Keying is possible from FSB to Cloud Storage machine.</td>
</tr>
<tr>
<td>7</td>
<td>Less throughput</td>
<td>More throughput</td>
</tr>
</tbody>
</table>

Table 3.1: Differences between the Expected and Final Approach

### 3.3 Installations

The Figure 3.4 shows the packages which are used in this research. Logstash, ElasticSearch, Kibana and Nginx packages are installed in the Cloud-Storage[47]. The filebeat package is installed in FSB.

### 3.3.1 Installing Java

Before installing these packages Java is to be installed into the Cloud-Storage. All these packages are dependent on Java repositories. So, Firstly, we installed Java
Chapter 3. Methodology

repository from webupd8team and later installed Oracle Java8 with OpenJDK environment.

3.3.2 Installing ElasticSearch

Elasticsearch is installed with a package manager by adding Elastic’s package source list. So, we imported the ElasticSearch public GPG key into apt. Then ElasticSearch Source list is added to the Cloud-Storage. Finally, from second party package, ElasticSearch package is installed. After Installing ElasticSearch, Configuration changes are done to get access to the local server. Then The ElasticSearch is started and added to initialization to startup on system boot up.

3.3.3 Installing Kibana

Kibana is installed with a package manager by adding Elastic’s package source list. Next, Kibana Source list is added to the Cloud-Storage. Then Finally, from the second party package, Kibana Package is Installed. The Kibana configuration file in the path /etc/kibana/kibana.conf is changed so that it listens on localhost. Similar to ElasticSearch, Kibana is also added to initialization to startup on system boot up.
3.3.4 Installing Nginx

As Kibana is configured to listen on localhost, a reverse proxy is to be added to allow external access to it. For this purpose, we installed Nginx in Cloud-Storage. After installing, Nginx is also initialized to startup on system boot up. Now the Nginx configuration is changed to direct the Cloud-Storage server’s HTTP traffic to the Kibana application which is listening on localhost:5601. Now we used OpenSSL file to create an admin user to access the Kibana web interface. Nginx will use the password from the file we generated in OpenSSL for basic authentication.

3.3.5 Installing Logstash

As the public key is installed already for Elasticsearch and same for Logstash, there is no need to install the key from the same repository. Then, the Logstash is added to the source list and installed from Secondary party.

3.3.6 Installing Filebeat

Firstly, the public key is installed into the FSB which is same as ElasticSearch key. Then the Filebeat is added to the source list. And now the Filebeat is installed. As the FSB doesn’t have network access, all these packages, keys are installed in Cloud-Storage and secure copied to FSB. Here, to install Filebeat, third party, a Debian package is used. As it is easy to handle and locate the package directory.

As Filebeat is used to ship logs to the ElasticSearch, the Filebeat index template is loaded into the ElasticSearch to analyze incoming Filebeat fields in an intelligent way.

3.4 Configuration files

It is very important to change the configuration files to get link in between different packages.

- **ElasticSearch configuration**: As to restrict outside access to ElasticSearch instance (port 9200), like reading the data, shutting down the ElasticSearch through HTTP API, the network.host: is given with localhost. It looks like this:
  
  network.host: "localhost"

- **Kibana Configuration**: The line with server.host is given the value as localhost, to deny them access to other servers. This is fine because we will use a Nginx reverse proxy to allow external access if needed. It looks like this:
  
  server.host: "localhost"
Chapter 3. Methodology

- **Nginx Configuration:** Now the Nginx configuration is changed to direct the Cloud-Storage server’s HTTP traffic to the Kibana application which is listening on localhost:5601. Now we used OpenSSL file to create an admin user to access the Kibana web interface. Nginx will use the password from the file we generated in OpenSSL for basic authentication. The server name is also added to this file. Proxies are also added into this configuration file for access of Kibana to outsiders/external devices or machines.

- **Logstash Configuration:** This is the path where we create a pipeline to access the logs from Filebeat and send to Elasticsearch. It has three files inside this directory. They are (02-beats-input.conf, 10-filter.conf, 30-elasticsearch-output.conf).
  - **02-beats-input.conf:** This is the Input Configuration file for Logstash package. The input configuration file is specified with beats input that will listen on TCP port 5044 and uses SSL certificate and private key.
  - **10-filter.conf:** This file is used to create the filtering function in Logstash. "grok" is used to parse incoming logs to make it structured and query-able. And also exclude_lines is used inside this file to remove the "beats-codec-plains" command into each log.
  - **30-elasticsearch-output.conf:** This is the Output Configuration file for Logstash package. This output configuration file is configured Logstash to store the beats data in ElasticSearch which is running at localhost:9200, in an index named with Filebeat.

  Note: To add filters for other applications that use Filebeat input, are to name the files so they sort between the input and the output configuration (i.e. between 02- and 30-).

- **Filebeat Configuration:** There are many changes done in inside the configuration file for Filebeat. Firstly, a prospector is written to create a harvester for every log.
  - The type of the prospector is written as a log, to access all types of logs from the ANCB. It is like:

```plaintext
-type:log
```
  - The paths section is filled with the directory of logs storage coming from ANCB. This directory contains many folders with different file names. To access all these files, the regression expression is used. It looks likes:

```plaintext
-/export/logs/*/tmp/*
```
  - some extra commands are also added. Exclude command is added to remove the duplicate files in the respective folder. Tags command is added, because to add the IP address of the FSB to every log, so that,
Chapter 3. Methodology

the differentiate of logs coming from different server becomes easy.

```
exclude_lines: ["*.index"]
tag: ["XXX.XXX.XX.XXX"]
```

- Output.Logstash: The bridged network IP address is added to the hosts value. The SSL certificate, key path are added in this section.

```
hosts:["XXX.XXX.XX.XXX"]
ssl.certificate_authorities: [path where the certificate is present]
ssl.key: [path where the key is present]
```

3.5 SSL Certificates

As this ELK-stack approach is shipping logs from one server to other, there is a need of creating an SSL certificate and key pair for strong and secure sharing. The certificate will be shared between the two servers and verifies each other server to send and receive the logs.

A directory in Cloud-Storage server is created to add the SSL certificate and Keys. As this server doesn’t have DNS setup, there is no option of creating a record and sharing certificates. So, here, in the OpenSSL.conf file the field under the [v3_ca] section subjectAltName (SAN) field of the SSL certificate is added with the server private address.

Now the SSL certificate and private key are created in an appropriate directory which we were created previously using RSA:2048 algorithm. Then the certificate created in Cloud-Storage server is secure copied to FSB and stored in a specific directory.

In Logstash configuration file, under the Certificates section the Certificate and private key path are added, so that, there will be a correct and exact sharing of logs between them.

3.6 Security in Kibana

X-pack package is downloaded as a zip file and attached to Elasticsearch-plugin, Logstash-plugin, and Kibana-plugin and make necessary changes in their configuration files like enabling X-Pack security and providing the login credentials for Elasticsearch and Kibana as shown in figure3.5. After the installation is complete, if we open Kibana it asks to log in and we should log in using the default usernames “Kibana” or “elastic” and password “XXXXX”.
Now that X-Pack is installed, we can start with securing the log files. The approach to provide security is to create different users and assign them different roles so that only the user who has permission to access any particular log file can view it and make changes if he wishes to. The figure 3.6 shows the default users that have been defined on the Elasticsearch node except the first which is created for the testing purpose. The user can make only the changes that he has “privilege” too. That is, we can give the user the privileges like “read”, “write”, “delete”, etc in the user tab [48]. If the user is given only “read” privilege then he cannot “write” something to the log file which he has access to.

The roles tab in figure 3.7 shows the default roles created in the Elasticsearch node. Roles tab customizes exactly which actions a user with the role can do, both on node level and an index level. A user can have multiple roles and at least a role. In Kibana the “Dev Tools” is a console where we can give the queries to create users, assign roles and give privileges to the users. This can also be done directly in “Management” section of Kibana instead of writing queries in Dev Tools console.

We can provide two kinds of security to logs. They are Document-Level Security and Field-Level Security. This both security levels are provided on a per-index basis. Document-Level Security is providing access to particular documents to the user.
Field-Level Security is providing the user to access a particular log file and restricting his access only to some particular fields like “time”, “date”, etc. in the log file. In this thesis, document level security is used for the logs inside the Elasticsearch because of the securing only to users.

3.6.1 Limitations to security

Plug-in

Elasticsearch allows all plug-ins in an easier way to attach and make the extra feature availability without any cost. When it comes to the security level, this high extensibility level comes to cost. But, minimum security is available free of cost, which can give basic level like document level, field level, SSL/TLS certificates in transmission, etc.

Behavior change in indexes with wild-card

With X-pack enabled to cluster with all wild cards and also the _all wild-card to the indices where the current user is having privileges for, not the set of all indices on the node. During the creation and retrieving aliases by providing this wild-card expressions for that alias names, if there are no existing authorized aliases that match the given wild-card expression will return the value as IndexNotFoundException.

With multi Document API’s

The Multi term vectors API and the term Multi get will return the value IndexNotFoundException when the user tries to access the non-existing indexes which he/she is not authorized for. By doing this action the user will leak the information of the fact that the index doesn’t exist, while he is not authorized to know anything about other extra indices.

Filtered Index Access

If the aliases are provided with filters, then this is the unsecured way to restrict the access to the document level. X-Pack can avoid this limitation using the document level security to the indices.
Field and Document Level Security limitations

When Field and Document level Security is enabled into a User’s role, for an index, following are the circumstances observed:

- Write operations cannot be performed in Kibana by the user:
  - The update API function is not supported in the Dev Tool of Kibana.
  - Bulk Update requests are not supported.

- The request cache is disabled for search requests.

When only the Document level Security is enabled into a User’s role, for an index, following are the circumstances observed:

- If the API’s aren’t Document based then it is not valid For Document level security. For example, field stats API is not valid for document level security.

- Document level security doesn’t affect global index statistics that relevancy scoring uses. So this means that scores are computed without taking the role query into account. Note that documents not matching with the role query are never returned.

- The has_child and has_parent queries aren’t supported as a query in the role definition. The has_child and has_parent queries can be used in the search API with document level security enabled.

- Any query that makes remote calls to fetch data to query by isn’t supported. The following queries aren’t supported:
  - term lookup
  - geo_shape query with indexed shapes
  - percolate query

- If suggesters are specified and document level security is enabled then the specified suggesters are ignored.

- A search request cannot be profiled if document level security is enabled.
Chapter 4
Validation and Testing

4.1 Validation of Security at Logstash and Filebeat

The validation will play a major role before the experiment is started. It initially gives the troubleshooting problems before the problems were faced in parsing logs. By validating, it is confirmed that security is done at both ends and the parsing the logs are done through the secured pipeline with SSL/TLS certificate.

4.1.1 At Logstash server

Before running the Filebeat, Logstash server, CloudStorage server should validate the certificate. To validate the Logstash server curl command is used even though the protocol used to communicate with Logstash is not based on HTTP. The command is

```
curl -v --cacert logstash-forwarder.crt https://localhost:5044
```
If the test is successful, an empty response error is produced.

4.1.2 At Filebeat server

Now, validation is done at Filebeat server. To validate the server, firstly stop the Filebeat server and then test the setup by running Filebeat in the foreground. The command is

```
filebeat -c filebeat.yml -e -v
```
If the test is successful, then you will get successful message.
Chapter 4. Validation and Testing

4.2 Validation of Security in Kibana

After installing and configuring X-Pack in Kibana, Elasticsearch and Logstash, all the servers should be stopped and started then. After a restart of all servers, we can observe a dialogue box asking for the user credentials for Kibana server. After logging into the Kibana by the respective user assigned with respective roles will observe the only logs to which the user is enrolled.

For example, if we look into the figure 4.2 of a user which is assigned only logs with the tag of its IP address, will display the only logs with that tag. This observation will depend on assigning roles to that assigned user. The above bar lines indicate the events per second entering into the Elasticsearch from the Filebeat. The below texts are the logs having fields and message. The refresh time can be set according to the requirement. There will be entry of new logs when there is a refresh function is done in page. We can confirm the parsing of logs from filebeat to Elasticsearch by observing this dashboard.

Figure 4.1: Kibana dashboard after enabling x-pack

Figure 4.2: Sample figure for testing
### Chapter 5

**Results and Analysis**

#### 5.1 Advantages and Disadvantages of Expected and Final Approaches

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Expected Approach</th>
<th>Final approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Advantages</strong></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Logs are stored in only 1 machine</td>
<td>No extra bridge is required to parse the logs, if there is huge traffic at Elasticsearch</td>
</tr>
<tr>
<td></td>
<td><strong>Disadvantages</strong></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Troubleshooting takes more time if there is an event missing at Elasticsearch.</td>
<td>Logs are stored in both machines</td>
</tr>
<tr>
<td>2</td>
<td>Extra reverse bridge is required (which is not possible)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>If the FSB is shutdown or rebooted then shipping of logs become slow and the load after the start or restart at FSB will be huge and it takes more time to parse them into the Elasticsearch.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Filebeat responses late when there is acknowledgement from the Elasticsearch as to pass extra machine compared to success-ed scenario.</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: Advantages and Disadvantages of Expected and Final Approaches
5.2 Behaviour of Elasticsearch, Logstash and Filebeat CPU Load varying users

Figure 5.1: Behaviour of open source tools with one Filebeat Node

<table>
<thead>
<tr>
<th>Size</th>
<th>Elastic</th>
<th>Logstash</th>
<th>Filebeat</th>
</tr>
</thead>
<tbody>
<tr>
<td>253</td>
<td>13.00</td>
<td>7.30</td>
<td>0.50</td>
</tr>
<tr>
<td>506</td>
<td>3.80</td>
<td>6.80</td>
<td>0.30</td>
</tr>
<tr>
<td>3k</td>
<td>24.60</td>
<td>7.30</td>
<td>0.50</td>
</tr>
<tr>
<td>25k</td>
<td>11.88</td>
<td>6.30</td>
<td>1.00</td>
</tr>
<tr>
<td>248K</td>
<td>14.45</td>
<td>6.29</td>
<td>3.30</td>
</tr>
<tr>
<td>2.5M</td>
<td>55.66</td>
<td>20.29</td>
<td>5.08</td>
</tr>
<tr>
<td>25M</td>
<td>46.97</td>
<td>27.56</td>
<td>7.64</td>
</tr>
<tr>
<td>242M</td>
<td>84.08</td>
<td>25.07</td>
<td>6.10</td>
</tr>
<tr>
<td>2.4G</td>
<td>68.34</td>
<td>29.44</td>
<td>20.18</td>
</tr>
</tbody>
</table>

Figure 5.2: Table for one User

This section provides the behavior of each open source tools with varying users. This scale gives the performance results of each tool at the different size of logs. This Figure 5.1 describes the load of each tool when the logs are parsed from one User i.e. One Filebeat. As the size of logs is increasing the load on each tool is also increasing. Out of this three open source tools, Elasticsearch is using more
CPU load than other open source tools. When the size of logs are moved from kilobytes to Megabytes the CPU load on Elasticsearch is increased rapidly to 50% of the total load. The confidence interval of every open source tool is given with there maximum and minimum fluctuation levels. The below figure 5.2 shows the intersection points of open source tools CPU load at respective size of logs.

![Figure 5.3: Behaviour of open source tools with two Filebeat Nodes](image)

If we look into this Figure 5.3 the CPU load of Elasticsearch is more compared to the other both tools same as in with one User. The load on Elasticsearch crossed 50% of total only after crossing the 50Mega bytes of logs size. The Error bars will define the maximum and minimum fluctuations of the open source tools. The below figure 5.4 will give the values at respective log size.

![Figure 5.4: Table for two Users](image)

<table>
<thead>
<tr>
<th>Behaviour with Two Users</th>
<th>Elasticsearch</th>
<th>Logstash</th>
<th>Filebeat</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5M</td>
<td>44.05</td>
<td>30.1</td>
<td>6.3</td>
</tr>
<tr>
<td>4.9M</td>
<td>43.6</td>
<td>25.35</td>
<td>7.97</td>
</tr>
<tr>
<td>25M</td>
<td>42.44</td>
<td>31.2</td>
<td>7.6</td>
</tr>
<tr>
<td>49M</td>
<td>46.49</td>
<td>26.3</td>
<td>8.5</td>
</tr>
<tr>
<td>242M</td>
<td>56.53</td>
<td>26</td>
<td>8.18</td>
</tr>
</tbody>
</table>
Thus from both the figures, we can conclude that the Elasticsearch is consuming more CPU load than other tools. It also says that the Elasticsearch load is increasing rapidly when log size is entering into the Megabytes. This point can be noted and can be decreased by making multiple nodes in the same cluster/data center. This also gives the advantage to this experiment is that for Pulling mode, Filebeat is carrying a very low load to parse the logs into the Logstash and which seems good note in using this method.

5.3 Elasticsearch and Filebeat performance when the number of users is varied

![Filebeat load varying users](image)

**Figure 5.5: Performance of Filebeat varying number of Users**

<table>
<thead>
<tr>
<th>Size (bytes)</th>
<th>1User</th>
<th>2Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5M</td>
<td>5.08</td>
<td>6.3</td>
</tr>
<tr>
<td>4.9M</td>
<td>6.1</td>
<td>7.97</td>
</tr>
<tr>
<td>25M</td>
<td>7.64</td>
<td>7.6</td>
</tr>
<tr>
<td>49M</td>
<td>7.82</td>
<td>8.5</td>
</tr>
<tr>
<td>242M</td>
<td>7.91</td>
<td>8.18</td>
</tr>
</tbody>
</table>

**Figure 5.6: Table for Filebeat Load**
Figure 5.7: Performance of Elasticsearch varying number of Users

![Elasticsearch Load Varying Users](image)

Figure 5.8: Table for Elasticsearch Load

<table>
<thead>
<tr>
<th>Elasticsearch</th>
<th>1User</th>
<th>2Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5M</td>
<td>24.6</td>
<td>44.06</td>
</tr>
<tr>
<td>4.9M</td>
<td>29.31</td>
<td>43.6</td>
</tr>
<tr>
<td>25M</td>
<td>34.14</td>
<td>42.44</td>
</tr>
<tr>
<td>49M</td>
<td>38.34</td>
<td>46.49</td>
</tr>
<tr>
<td>242M</td>
<td>46.97</td>
<td>56.53</td>
</tr>
</tbody>
</table>

Figure 5.5 and Figure 5.7 shows the CPU load performance when the number of users are varying. As the filebeat load is less than 10% even with more load, it says that this open source tool is negligible and can be ignored compared with Elasticsearch load. In figure 5.7 the load of Elasticsearch is below 50% with one user and crossed 50% in 2 users after reaching the 50 Megabytes of logs size. This indicates that as the logs size is increased to Gigabytes the load may reach to 80% value and will increase the burden at Server node. So, to decrease this load at server side another node with Elasticsearch is installed in the same cluster/design host. The figures 5.8 and 5.6 will give the tabular section of CPU load at different log size with respect to their performance graphs.
5.4 Comparison of ELK approach with Legacy method

![Graph showing CPU load comparison of ELK and Legacy method](image)

Figure 5.9: Performance comparison of ELK and legacy method with 1 user node

<table>
<thead>
<tr>
<th>Size (bytes)</th>
<th>ELK</th>
<th>Legacy</th>
</tr>
</thead>
<tbody>
<tr>
<td>253</td>
<td>10.3</td>
<td>0.00</td>
</tr>
<tr>
<td>506</td>
<td>10.6</td>
<td>0.00</td>
</tr>
<tr>
<td>3k</td>
<td>31.9</td>
<td>0.00</td>
</tr>
<tr>
<td>25k</td>
<td>18.18</td>
<td>0.06</td>
</tr>
<tr>
<td>248K</td>
<td>20.74</td>
<td>25.00</td>
</tr>
<tr>
<td>2.5M</td>
<td>70.00</td>
<td>75.95</td>
</tr>
<tr>
<td>25M</td>
<td>74.40</td>
<td>84.63</td>
</tr>
<tr>
<td>242M</td>
<td>82.90</td>
<td>88.15</td>
</tr>
<tr>
<td>2.4G</td>
<td>80.20</td>
<td>97.78</td>
</tr>
</tbody>
</table>

Figure 5.10: Table for One User CPU load

This figure 5.9 compares the CPU load of averaged ELK open source tools with the legacy approach. Legacy approach graph is described by a summation of log generation and log shipment inside the SGSN node. Thus this graph is started from zero and stayed till 25K Bytes of log size because of very small size. Going to the figure this clearly indicates the better performance of ELK approach to the Legacy method. As we mentioned in the section 5.2 that the CPU load is increased to 50% after reaching the Megabytes of logs size. Even after the rapid
increase in this graph, it stays lower than the Legacy approach. At 2.4G bytes the ELK graph is decreased to 80.2 from 82.9 that is because of slow packet input from the filebeat, which is slow logging scenario and it can be neglected. Thus this situation defines that the graph lies at around 80-85% of load in Giga bytes.

![CPU load 2 Users](image)

**Figure 5.11**: Performance comparison of ELK and Legacy method with 2 user nodes

<table>
<thead>
<tr>
<th>Total Comparison of CPU 2 User</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>2.5M</td>
</tr>
<tr>
<td>4.9M</td>
</tr>
<tr>
<td>25M</td>
</tr>
<tr>
<td>49M</td>
</tr>
<tr>
<td>242M</td>
</tr>
</tbody>
</table>

**Figure 5.12**: Table for two Users CPU Load

This figure 5.11 also compares the CPU load of averaged ELK open source tools with the legacy approach when there is 2 user nodes inside the design host. This graph is taken from the 2.5M bytes of log size and then it is compared with both approaches. This curves also clearly says that ELK approach is staying low compared to the legacy method at all different size of logs. At a maximum of 242M bytes, both curves are approaching to each other and having a intersect
and indicating the high CPU load in ELK approach which leads to install another Elasticsearch node inside the cluster/design host.

5.5 Pulling Scenarios at Logstash in CloudStorage server

There are other alternative open source tools than Filebeat to parse the logs from Client to Server. But, this method also uses the pushing mode inside this environment. The only method to make Pulling mode possible is to write a script which can access all the logs from the SGSN node and other script in Server which pulls the log from the client and finally parse into the Elasticsearch.

Coming to the other alternative method (Pushing mode), We can use Rsyslog. Rsyslog is based tool of syslog which collect important information from the kernel and many of the programs that run to keep UNIX-like servers running. This experiment is used more storage than the ELK stack approach with Filebeat. Rsyslog will only collect the syslog files and will parse them to the server. Whereas in our experiment, there are many other logs with security accessed and in different formats and sizes. To run this experiment [49], Rsyslog is set up at client side named rsyslog-client and another rsyslog is set up in server side named rsyslog-server. The rsyslog server will collect all the logs from the client side and then parse to server side and then rsyslog-server will parse logs to Logstash and then finally to Elasticsearch. Thus how this experiment is done. This experiment needs additional package than our original method and also additional parsing which takes more time for logs reaching Elasticsearch.
5.5.1 Advantages and Dis-advantages of Alternative method

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Pushing mode</th>
<th>Pulling Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Logs are parsed only once.</td>
<td>Logs have to be parsed twice</td>
</tr>
<tr>
<td>2</td>
<td>Tags can be attached to the Filebeat.</td>
<td>No tags can be attached to Rsyslog.</td>
</tr>
<tr>
<td>3</td>
<td>more reliable.</td>
<td>Less reliable compared to Pushing mode.</td>
</tr>
<tr>
<td>4</td>
<td>Filebeat can slow the parsing of logs.</td>
<td>Rsyslog can’t slow down.</td>
</tr>
<tr>
<td>5</td>
<td>Consumes less storage for open source tools</td>
<td>Consumes more storage than pushing mode</td>
</tr>
<tr>
<td>6</td>
<td>Less time required to transfer the logs</td>
<td>more time required to transfer</td>
</tr>
</tbody>
</table>

Table 5.2: Advantages and Disadvantages of both modes

5.6 Confidentiality and Integrity

Security feature is added to this experiment using an additional package X-Pack as mentioned in section 3.6.

Log files which are shipped with the help of Filebeat from FSB to Logstash and stored in Elasticsearch are now secured by attaching the X-Pack plug-in to the plug-ins of Elasticsearch, Logstash and Kibana. We can say they are secured because their authentication is now restricted only to some specific users. This is done by creating users and assigning them roles and privileges.

When x-pack is attached, the Kibana is only accessed with user credentials as shown in figure 4.1. After logging into the web server with them, we can create the roles and users assigning them.

5.7 Resource Consumption

Disk Usage is the main element where every individual focuses to maintain there server, program, project. As all the product runs using the disk storage, allocating the memory before the overload will make the owner of the product be successful. So, here, in this context, the amount of storing the entered logs will be considered and there performance is shown in the scenario.

The logs entering to Elasticsearch are stored in the node in Json format. These logs are indexed and stored in a path which will be defined in the configuration.
file after the installation. Each incoming log is indexed and allowed for searching. This indexing is done in Lucene format inside the Elasticsearch and stored in respective shards and replicas. It is said that the data volume in Elasticsearch will experience the expansion in size during indexing which is true. It doesn’t have to be true, depending on the types of queries you expect to run and how you configure your indexing accordingly. The disk usage of logs is dependent on how you decide the searching, indexing, enabling and dis-enabling the required fields based on the requirements[50].

The parameters like analyzed and not-to-be-analyzed in the configuration of certain text fields will have a significant role in disk utilization. In this research, the fields like time-stamp, offset, beatname, beat version, prospector_type, source, _id, _index, host, tags are enabled. By adding this many parameter to Elasticsearch, the disk utilization is more increased compared to the original log. The disk utilization is increased in the ratio of approximately 1.3:1 to that of the original log.

**Parameters effecting the Disk Utilization**

The parameters that play a major role in disk usage will be discussed in this subsection. they are _all, doc values, replication.

- **doc values**: This field will decrease the heap memory size of the Elasticsearch which is very appreciable and useful. Enabling doc values field will results in additional on disk data structures to be created at index time which results in larger index files.

- **_all**: This is a catch-all field, which concatenates the all the text fields of a log into one long string using space as a delimiter and will additionally increase the storage of logs. This field helps to search a total log if one of the text fields is known from the log. This field will be analyzed and indexed inside Elasticsearch but is not duplicated again. As this field uses extra CPU load and extra disk space it is disabled by default.

- **Replication**: Elasticsearch ensures the resiliency through the use of replication. Already by default, there will be a replica shard for every shard, which will store all index values present in shards. If the replica field is enabled it will give more resilience to the Elasticsearch but will make the disk storage twice the original size. So, for the less disk Usage, dis-enabling the replication field is recommended.

The more detailed view of disk Usage on structured and unstructured data is available from [50]. Where this site shows the data and table of string fields with enabling and disabling the above-mentioned parameters.

The analysis from this citation is, the disk storage depends and changes according to various parameters. Out of all the parameters, _all is the parameter...
which increases the disk storage in higher rate comparing to other parameters. There is a significant reduction in storage footprint for the _all parameter, which is an easy win if the users are familiar with the fields they want to search against. Even if we can’t assume users know what fields to search, it can customize your search application to take what the user perceives as a non-fielded search and construct a multi-field search query behind the scenes.

From the above citation analysis, we can conclude that highly structured data allows for better compression compared to semi-structured data. When there are fewer deployments of logs/data, this structured or semi-structured data will not make a huge difference. If there is a huge deployment, it will certainly be worth having some intentionally in configuring the mapping.

5.8 Solutions for overcoming Failure Scenarios

5.8.1 Faced Failure scenarios

Elasticsearch is too slow

When there is a decrease in the rate of log entry into the Elasticsearch from Logstash it specifies that the Elasticsearch is slow and accessing fewer logs. This happens because the heap size of Elasticsearch is set to low and it should be changed according to the node RAM of the node. This heap size has Xms and Xmx, minimum heap size and Maximum heap size accordingly [51]. The Xmx is set not more than 50% of physical RAM of the node, to ensure that there is enough physical RAM left for kernel file system caches. It is also possible to set the heap size via an environment variable. This can be done by commenting out the Xms and Xmx settings in the JVM.options inside the node.

The other way to make the Elasticsearch run normal way is increasing the number of shards and replicas inside the Elasticsearch node. These settings can be changed from the developer tool in Kibana by updating the indices [52].

Kibana can’t connect to Elasticsearch

This failure is faced when the Elasticsearch server is down or failure in configuration files. To overcome this failure, the configuration files of Elasticsearch and Kibana is crosschecked and then restarting the Elasticsearch and Kibana will resolve this issue. Port number and host IP address to be crosschecked inside the configuration files.

No access to Kibana

This is faced when there is no firewall setup. To solve this issue this researched used Nginx and making the correct configuration by adding the port number and
host IP address in respective areas[53]. To check the Nginx error logs we can move into the /var/log/nginx/error.log path and can check the error.

No shipment of logs to Elasticsearch from Filebeat

This is a common failure which is faced by many developers from the Google domain. This issue is faced because of many failure configurations or missing dependent packages or wrong installations. Some of the important points to be noted while editing the configuration file of Filebeat are

- Log file paths
- Indentation
- IP address of Server
- SSL Certificates

Filebeat is working slow

This situation is faced due to the opened file handlers of log files which are configured. Filebeat keeps the file handlers open in case it reaches the end of a file so that it can read new log lines in near real time.

- If filebeat is harvesting a large number of files, the number of open files can become an issue.
- In the most environment, the number of files that are actively updated is low. The close_inactive configuration option should be set accordingly to close files that are no longer files.
- There are additional configuration options that to close the file handlers, but all of them should be used carefully because they can have side effects. Options are:-
  - close_removed
  - close_renamed
  - close_eof
  - close_timeout
  - harvester_limit

5.8.2 Overload Protection and Resilience Mechanism

Thus from above subsection 5.8.1 the Overload protection and Resilience mechanism is increasing the heap size or increasing the number of shards inside the Elasticsearch node. This will decrease this issue to a level of the extent and for further problems adding another node inside the same cluster with installing Elasticsearch inside it can shut down the above issue. Also, at Filebeat side, closing file handlers can remove 90% slow mechanism of parsing logs to the Logstash.
Adding other extra nodes inside the same cluster makes the work simple and reliable because of sharing of workload by many Elasticsearch than one Elasticsearch. This scenario is followed in many cloud computing techniques. To implement this scenario in the SGSN-MME node is adding another virtual machine to the design host with Elasticsearch installed inside it.
6.1 Conclusion

The logs generated in the vSGSN-MME node from vdallas are stored in FSB. As mentioned in chapter 3.1 the storing of Logs in Elasticsearch from the vSGSN-MME node is successful and it is achieved with the help of open source tools (Logstash, Kibana, and Filebeat). This research showed the way of storing the logs in the Elasticsearch in an efficient way than the legacy method and adding security feature for the logs to make them more secure by restricting the access of the logs to assigned users only. These shipped logs are viewed from Kibana user interface.

6.2 Research Question and Answers

RQ 1: How efficient is the ELK concept for appropriately handling of log in the SGSN-MME such that CPU load and resource consumption’s are minimized, and reliability increased?

A: The graphs in the result section 5.9, 5.11 gives the compared graph between the ELK stack approach and legacy approach. These curve graphs indicate the reliability, CPU load, and resource consumption values which is better in the ELK stack approach compared to the legacy method. From the tables 5.10 and 5.12 will confirm that, as number of logs get increased the CPU load at Legacy approach is getting decreased. Thus, we can clarify that ELK approach is more efficient than Legacy method.
Chapter 6. Conclusions and Future Work

RQ 2 How many events can be handled by the ELK platform considering different modes? How does the Elasticsearch platform scale its performance when the number of user/connections handled at the SGSN is increased?

A: The Elasticsearch can handle huge number of events depending on the ROM and RAM of the nodes deployed with the Elasticsearch and running explained in 5.7. As no other modes are possible in this research environment, only the pushing mode is considered and its performances are given in section 5.3 figure 5.7. The Elasticsearch is using less CPU load compared to legacy Method in both the scenarios of having single and multiple users. When the number of users is increased, the Elasticsearch performances are measured and a line graph is drawn using its values in subsection 5.3.

RQ 3 Would there be an advantage if an alternative data transfer mode to the Filebeat is applied?

A: The better and suitable solution to this research and experiment is Pushing mode. The advantages of pushing mode is explained clearly in table 5.2. This table 5.2 clearly gives the difference of studied experiment and Pulling mode which is done literature survey and explained clearly in section 5.5.

RQ 4 What kind of overload protection and resilience mechanisms are appropriate for the Elasticsearch mechanism?

A: Increasing the heap size or increasing the number of shards inside the Elasticsearch node will be a solution for Overload Protection and resilience mechanism for the Elasticsearch in SGSN-MME node. Adding other extra nodes inside the same cluster makes the work simple and reliable because of sharing of logs by many Elasticsearch than one Elasticsearch.

RQ 5 How the Confidentiality and Integrity of data and system are handled for logs in Elasticsearch?

A: Confidentiality and Integrity of logs inside the Elasticsearch are provided using X-Pack package attached Elasticsearch inside the node. By attaching this package no unassigned user can read, write or delete the log with a specific tag. This research also concluded the difference between with other security packages with X-Pack.
6.3 Future Work

As the present day is focusing on micro service to deploy all there services in Kubernetes cluster and decrease the load at each nodes and increase the efficiency with more throughput. So, considering the present day challenge and future goals, the below section is defined as a future work.

In future, if the SGSN-MME is containerized, then the virtual machines focused in this research will be available in the container to which the installing of ELK stack will be the best solution and giving the best suitable and efficient method to use ELK stack is probable. In this research, the open source tools services can be deployed in different and multiple containers in a Kubernetes cluster and can be visualized with more events including Cluster logs.
References


References


Appendix A

elasticsearch.yml

1
2
# Elasticsearch Configuration
3
4 network.host: localhost
5
6 http.port: 9200
Appendix B

---

**filebeat.yml**

```yaml
# = = = = = = = = = = = = = = = = = = = = = = = = = = = Filebeat prospectors
#============================= Filebeat prospectors

filebeat.prospectors:

- type: log
  enabled: true
  paths:
    - /export/logs/*/tmp/*
  exclude_lines: ["^*.index"]
  tags: ["XXX.XXX.XXX.XXX"]
  document_type: log

#============================= Filebeat modules
#============================= Filebeat modules

filebeat.config.modules:
  path: ${path.config}/modules.d/*.yml
  reload.enabled: false

#-------------------------- Elasticsearch template setting
#-------------------------- Elasticsearch template setting

setup.template.settings:
  index.number_of_shards: 3

#-------------------------- Elasticsearch output
#-------------------------- Elasticsearch output

#output.elasticsearch:

#-------------------------- Logstash output
#-------------------------- Logstash output

output.logstash:
```

---

hosts: ["XXX.XXX.XXX:5044"]

# List of root certificates for HTTPS server verifications
ssl.certificateAuthorities: ["/etc/pki/tls/certs/logstash-forwarder.crt"]

# Certificate for SSL client authentication
ssl.certificate: "/etc/pki/tls/certs/logstash-forwarder.crt"

# Client Certificate Key
ssl.key: "/etc/pki/tls/private/logstash-forwarder.key"
Appendix C

Logstash.yml

# ----------------- Other Settings -----------------

xpack.monitoring.enabled: "true"
xpack.monitoring.elasticsearch.url: "http://localhost:9200"
xpack.monitoring.elasticsearch.username: "elastic"
xpack.monitoring.elasticsearch.password: "changeme"
Appendix D

Kibana.yml

```yaml
1 elasticsearch.username: "elastic"
2 elasticsearch.password: "changeme"
```
Appendix E

Logstash config files

E.1 02-input.conf

```ruby
input {
  beats {
    port => 5044
    ssl => true
    ssl_certificate => "/etc/pki/tls/certs/logstash-forwarder.crt"
    ssl_key => "/etc/pki/tls/private/logstash-forwarder.key"
  }
}
```

E.2 20-filter.conf

```ruby
filter {
  if "beats_input_codec_plain_applied" in [tags] {
    mutate {
      remove_tag => ["beats_input_codec_plain_applied"]
    }
  }
}
```

E.3 30-output.conf

```ruby
output {
  elasticsearch {
    hosts => ["localhost:9200"]
    user => "elastic"
    password => "changeme"
    sniffing => true
    manage_template => false
  }
}
```
Appendix E. Logstash config files

```ruby
index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"
document_type => "%{[@metadata][type]}"
```