B.E. Project Report on
SERVER LOG ANALYSIS - USING ELK STACK

Submitted in partial fulfillment of the requirements
of the degree of

BACHELOR OF ENGINEERING
IN
INFORMATION TECHNOLOGY

BY

Group No: 38

Roll No.	**Name of student**
126	Sushant Sawant
127	Pranav Tella
128	Anuj Vyas

Under the guidance of:
Prof. Sanober Sultana Shaikh
(Assistant Professor, Department of Information Technology, TSEC)

Information Technology Department
Thadomal Shahani Engineering College
University of Mumbai

2020-2021

CERTIFICATE

This is to certify that the project entitled **"Server Log Analysis - Using ELK Stack"** is a bonafide work of

Roll No.	**Name of student**
126	Sushant Sawant
127	Pranav Tella
128	Anuj Vyas

Submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"BACHELOR OF ENGINEERING"** in **"INFORMATION TECHNOLOGY"**.

Ms. Sanober Sultana Shaikh	Dr. Maduri Rao	Dr. G.T. Thampi
Project Guide	Head of Department	Principal

Project Report Approval for B.E.

Project report entitled ***Server Log Analysis - Using ELK Stack*** by

Roll No.	Name of the student
126	Sushant Sawant
127	Pranav Tella
128	Anuj Vyas

is approved for the degree of ***"BACHELOR OF ENGINEERING" in "INFORMATION TECHNOLOGY"***.

Examiners

1._____

2._____

Date:

Place: Mumbai

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1) _____
Sushant Sawant - 126

2) _____
Pranav Tella - 127

3) _____
Anuj Vyas - 128

Date:

Acknowledgment

The work presented in this report is purely constructed as a part of the course in Bachelor of Engineering (Information Technology), Mumbai University. We express immense pleasure in presenting this report on "**Server Log Analysis - Using ELK Stack**". We take the opportunity to thank all those who have contributed to conceive this project report in the most presentable manner and with attention to complete detail.

We wish to express sincere thanks and heartfelt gratitude to respected mentor and guide **Prof. Sanober Sultana Shaikh,** Assistant Professor of Department of Information Technology Engineering, for her in-depth and enlightening support with all the constructive criticism, engrossment in the project, implantation detail, timely analysis, and helping us and suggesting new ideas during hard times. Her effort, support, advice, and mentorship shall always remain priceless and memorable.

Lastly, a word of thanks is also due to our parents and friends who have supported us throughout and helped us to develop this project, without the help from those this would not have been what it stands today.

Sushant Sawant - 126

Pranav Tella - 127

Anuj Vyas - 128

Abstract

In this project, a new type of system is described which can be used for analyzing the server logs. We will be designing the GUI on which we can see the dashboard and all the detailed information about the server. Also, this dashboard can help the admins to detect the different kinds of attacks on the server and to take the appropriate action.

The proposed system provides the GUI for visualizing the server logs and to take the important mission-critical decisions by analyzing the server logs. We can visualize the server logs using pie-charts, bar-graphs, histograms,heat-map, etc. A user just needs to open the dashboard to see the data. If the user wants then the user can also see the time series data which is coming from the elastic search database.

Table of Contents

List of Figures

Chapter 1
Introduction

1.1 Introduction

In the modern world, most of our day to day work, activities, and procedures are automatized. The use of an organization's network or use of the internet has become an inevitable resource of work. On the other hand, technology developments have aided threat actors to use newer, stealthier ways to invade networks and to gain persistence in the network. They use obfuscation techniques to defame or damage or to obtain ransom benefits. The general threats include computer virus, security software, Trojan horse, Adware and Spyware, Computer Worm, Denial of Service (DoS), Distributed Denial of Service (DDoS), Phishing, Rootkit, SQL Injection, and Man in the Middle Attacks (MITM). In the current statistical surface, every system is infected by one or other types of Malware.

Malware is a generic term that covers all kinds of threats. Malicious software or in other terms Malware in the current digital world are modernized. Contrary to traditional malware, modern malware is highly stealthy focusing primarily on unknown vulnerabilities of the network. These modern malware attacks randomly or on specific targets. Modern malware is well defined with clear goals. Due to these characteristics, the network defenses of today like Intrusion prevention system (IPS), antivirus software, firewalls, cloud led alternatives and virtual private networks are no longer sufficient to prevent modern malware entry and persistence in the network.

Firewalls can inspect and monitor ports but cannot inspect communications that are happening through the ports. IDS/IPS prevents attacks with the help of signatures of known threats. Though literature states that IDS/IPS prevents unknown threats, in reality, it is not the case. Unknown threats can be addressed only with a rich understanding of vulnerability. Hence, known and unknown vulnerabilities play a vital role in the plane of threats. This necessitates anti-malware to be smart, intelligent enough to critically analyze not only the network traffic but also the processes that spawn them. The signature or list-based security alternatives alone may not be effective.[1]

Also, in the current scenario of threats, threat actors use newer, stealthier to invade networks and to gain persistence in the network. A plausible and easy way for threat actors is to repudiate as legitimate software like PowerShell and Windows Management Instrumentation. By using these tools, the modern malware deceit antivirus software and include themselves in the whitelist of a network. These kinds of threats are stateless and file-less. Once such threats are file-less malware variants of modern malware.

1.2 Aim

In real life there are lots of network operation and security management issues, we have lots of users which include students, faculty, and staff. Also, we have a lot of systems from which we are generating data like routers, firewalls, and servers. That’s why we have a lot of log files like Netflow, Syslog, and access log. service logs and audit logs. We are developing a GUI on which we can view and analyze this large amount of data very easily and conveniently. To visualize the data dashboards are created on which the entire summary will be shown in the form of the bar

graphs, pie-charts, and other pictorial techniques. Also, the system admin can possibly identify the possible attacks on the server.

1.3 Objective

- Logs & events analysis for network managements
- Logs & events collection from multiple sources
- Accept and parse different log formats
- Handling large amount, and various formats of data
- The scalable architecture of the graphical user interface(GUI)

1.4 Scope

Elasticsearch is the main data store, analytics, and search engine. Logstash handles log integration, processing, parsing, and enrichment. Kibana is a visualization layer sitting on top of Elasticsearch. It allows us to perform search queries and aggregations and create visualizations using a simple web user interface(UI).

The ELK stack is a very flexible platform and it has been used for multiple use-cases across different industries. In the Information Security domain, it is usually compared with the Splunk platform.

Some of our use-cases of the ELK stack include:

- Dashboarding
- Threat Hunting
- Log Management
- Security Monitoring
- Forensic Log Investigations

The common theme in all the above use-cases is that the ELK platform provides you with capabilities to analyze the logs in a significantly better manner.
Further for the people who don't understand the visualization for them, we can also develop the chatbot so that everyone can get to know the detailed information of the server.
In our system, there are several features we want to reach:
1. Detect DDOS attack: As DDOS attack is one of the most popular attacks and will cause catastrophic consequences to the system, we need to pay much of our attention to this part.
2. Detect botnet IP: Find out whether there is an IP address belongs to the botnet trying to visiting our server and alert for it.
3. IP DNS Blocklist: Check the IP address of the sending mail server against a public list of mail servers known to send spam.
4. Detect abnormal behavior: Try to find out unusually dangerous actions, for example, is there anyone in my organization visiting a known malware URL/domain.
In order to describe system functionality, we would make some definitions and assumptions. In the current stage, these assumptions would simplify the system situation. And we would try to reach the assumptions in the next stage.
1. All normal traffic and intrusion can be recorded by the system. And from the records, we can find out the IP address, time, events and all the other necessary information.

2. The IP addresses or the IP addresses controlled by the attacker are in the dataset we are going to use.
3. DNS blocklist and the malware domain list contains all the upcoming attacks.

1.5 Term to Explain

1. DDOS attack: Distributed Denial of Service (DDoS) attacks refer to the use of client/server technology to combine multiple computers as an attack platform to launch DDoS attacks on one or more targets, thereby multiplying denial of service attacks by power. There are many ways to attack DDoS. The most basic DoS attack is to use a reasonable service request to occupy too many service resources, so that legitimate users can not get the response of the service. A single DoS attack generally adopts a one-to-one approach. When the target CPU speed is low, the memory is small, or the network bandwidth is small, and the performance is not high, the effect is obvious.
2. Malware Domains: The Malware Domains page lists domains that are known to generate spam, host botnets, create DDoS attacks, and generally contain malware.
3. Botnet: A botnet is a network that can be used to control a bot program virus by using one or more means of communication to form a one-to many control network between the controller and the infected host. The attacker spreads bots through various channels to infect a large number of hosts on the Internet, and the infected host will receive an attacker's instructions through a control channel to form a botnet. Launching DDos attacks using Botnet is one of the most important threats at present. Attackers can send instructions to all bots they control, allowing them to simultaneously access specific network targets at a specific time, thus achieving the goal of DDos. Because Botnet can form a huge scale, and it can be better synchronized by using DDos attack, it can make DDos more harmful and prevent more difficulty when issuing control commands.
4. DNS blocklist: IP DNS Blocklist checks the IP address of the sending mail server against a public list of mail servers known to send spam. The DNS blocklist is actually a list of IP addresses that can be queried. The DNS query method is used to find out whether an A record of an IP address exists to determine whether it is included in the DNS blocklist. The IP address in the list indicates that the spam has been posted externally. Therefore, it is used by the spam filter to filter spam sent by the list like a virus definition file.

Chapter 2
Review of Literature

2.1 Literature Overview

The definition of logs are time-series based machine data, including IT system information (servers, network devices, operating systems, application software), and various sensor information for the Internet of Things. The log reflects user behavior and in fact data. The log processing system has undergone a long period of development and has three main phases:

1. Log processing v1.0: The log is not processed centrally; the log is only traced after the event, and the log is not detected after the hacker is invaded; using the database to store the log is not suitable for complex event processing.
2. Log processing v2.0: Using Hadoop platform to realize log offline batch processing, the disadvantage is poor real-time performance; using Storm stream processing framework, Spark memory computing framework to process logs, but Hadoop/Storm/Spark are programming frameworks, not a real-time system
3. Log processing v3.0: Use log real-time search engine to analyze logs. The first feature is fast. The log is delayed from the generation to the search analysis. The second is large, and the amount of TB logs is processed every day. The third is flexible. Search for any logs. The solutions represented are Splunk, ELK, SILK.

2.1.1 Domain Explanation

Once an application is deployed the maintenance and monitoring is the last part of the software development life cycle (SDLC). The monitoring of the application can be achieved by processing the data stored in the log files and parsing and visualizing the log files. We generate a lot of data. For example

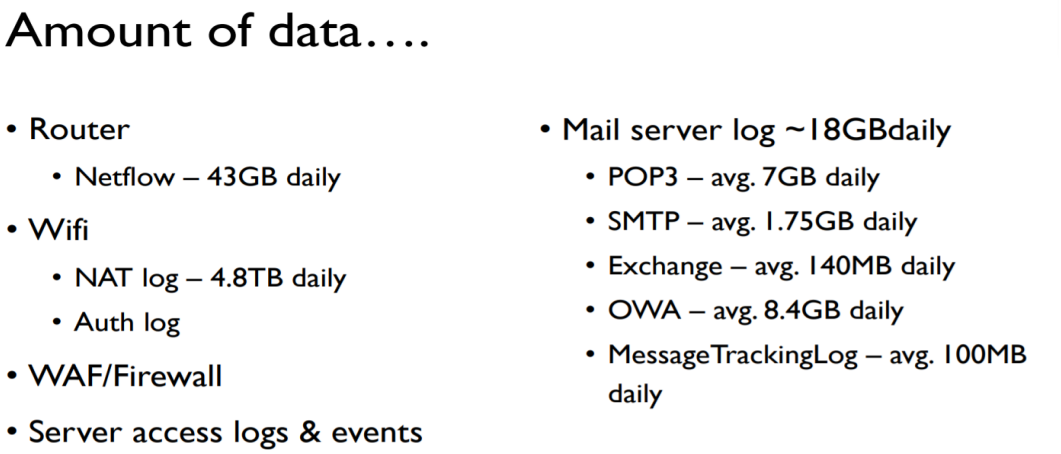

Figure 2.1.1.1 - Analysis of data generated in one day

What is ELK?

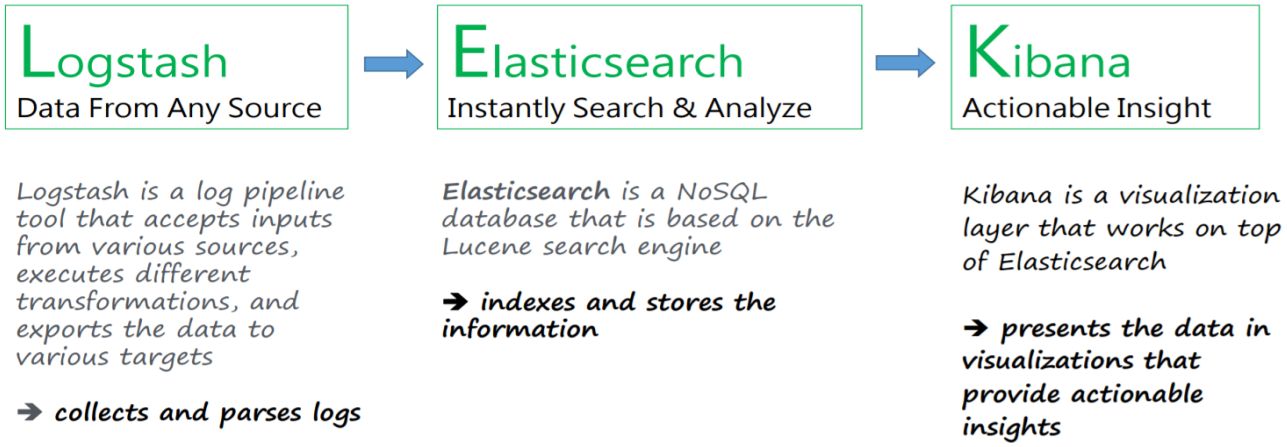

Figure 2.1.1.2 - The diagram explaining what is ELK

Why ELK?

With ELK stack we can easily visualize this large amount of data as follows:

- Rapid on-premise (or cloud) installation and easy to deploy
- Scalability and resilience: clusters, nodes, and shards
  Elasticsearch is built to be always available and to scale with your needs. It does this by being distributed by nature. No need to overhaul your application.
- Easy and various APIs to use
- Ease of writing queries, a lot easier than writing a MapReduce job
- Availability of libraries for most programming/scripting languages
- Elastic offers a host of language clients for Elasticsearch, including Ruby, Python, PHP, Perl,.NET, Java, and Javascript, and more
- Tools availability
- It's free (open source), and it's quick

Overall structure of ELK stack

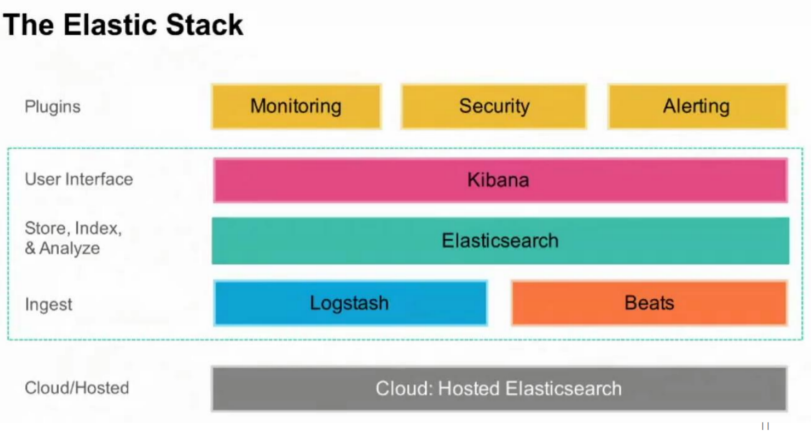

Figure 2.1.1.3 - Structure and modules of ELK stack

Modules of ELK stack

Open Source:

- ElasticSearch
- Logstash
- Kibana
- Beats
    - Data shippers (collect, parse & ship)

Extension plugins:

- Alerting (Watcher)
  Proactively monitoring and alerting based on elasticsearch queries or conditions
- Security (Shield)
  Protect and provide security to elastic stack
- Monitoring (Marvel)
  Monitor and diagnose health and performance of elastics cluster
- Graph
  Discover and explore the relationships live in data by adding relevance to your exploration

2.1.2 GUI

Kibana

No centralized logging solution is complete without an analysis and visualization tool. Without being able to efficiently query and monitor data, there is little use to only aggregating and storing it. Kibana plays that role in the ELK Stack — a powerful analysis and visualization layer on top of Elasticsearch and Logstash.

What is Kibana?

Completely open-source, Kibana is a browser-based user interface that can be used to search, analyze, and visualize the data stored in Elasticsearch indices (Kibana cannot be used in conjunction with other databases). Kibana is especially renowned and popular due to its rich graphical and visualization capabilities that allow users to explore large volumes of data.

Kibana can be installed on Linux, Windows, and Mac using .zip or tar.gz, repositories, or on Docker. Kibana runs on node.js, and the installation packages come built-in with the required binaries.

Kibana visualizations

Kibana is renowned for its visualization capabilities. Using a wide variety of different charts and graphs, you can slice and dice your data any way you want. You can create your own custom visualizations with the help of vega and vega-lite.

- Vega — A declarative format to create visualizations using JSON. Generate interactive displays using D3(Data-Driven Documents).
- Vega-Lite — An easier format to use than Vega that enables more rapid data analysis. Compiles into Vega.[2]

You will find that you can do almost whatever you want with your data.

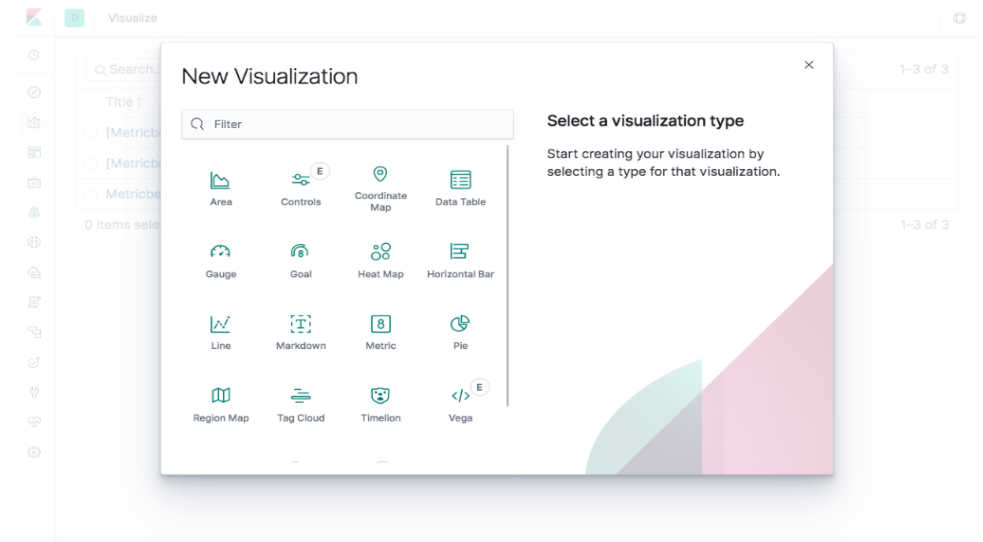

Figure 2.1.2.1 - Different type of visualization available in kibana

Creating visualizations, however, are now always straightforward and can take time. The key to making this process painless is knowing your data. The more you are acquainted with the different nooks and crannies in your data, the easier it is.

Kibana visualizations are built on top of Elasticsearch queries. Using Elasticsearch aggregations(e.g. sum, average, min, max, etc.), you can perform various processing actions to make your visualizations depict trends in the data.

Kibana dashboards

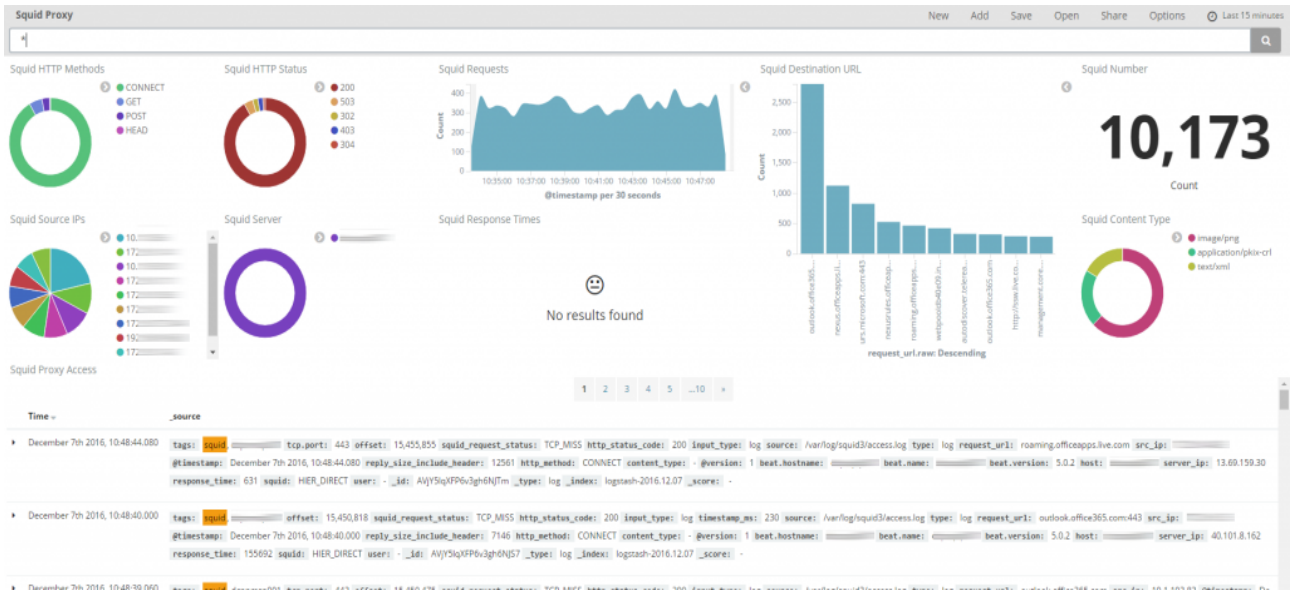

Figure 2.1.2.2 - Sample kibana dashboard

Dashboards give you the ability to monitor a system or environment from a high vantage point for easier event correlation and trend analysis. Dashboards are highly dynamic — they can be edited, shared, played around with, opened in different display modes, and more.

2.1.3 Hardware Requirements

Hardware

To deploy Elasticsearch to production, there are a few recommendations that you should consider. Nothing is a hard-and-fast rule; Elasticsearch is used for a wide range of tasks and on a bewildering array of machines. But these recommendations provide good starting points based on our experience with production clusters.[3]

Memory

A machine with 64GB of RAM is the ideal sweet spot, but 32GB and 16GB machines are also common. Less than 8GB tends to be counterproductive (you end up needing many, many small machines), and greater than 64GB has problems that we will discuss in Heap: Sizing and Swapping

CPUs

Most Elasticsearch deployments tend to be rather light on CPU requirements. As such, the exact processor setup matters less than the other resources. You should choose a modern processor with multiple cores. Common clusters utilize two- to eight-core machines.

If you need to choose between faster CPUs or more cores, choose more cores. The extra concurrency that multiple cores offer will far outweigh a slightly faster clock speed.

Disks

Disks are important for all clusters, and doubly so for indexing-heavy clusters (such as those that ingest log data). Disks are the slowest subsystem in a server, which means that write-heavy clusters can easily saturate their disks, which in turn become the bottleneck of the cluster.

If you can afford solid-state drives(SSDs), they are by far superior to any spinning media. SSD-backed nodes see boosts in both query and indexing performance. If you can afford it, SSDs are the way to go.

Network

A fast and reliable network is obviously important to performance in a distributed system. Low latency helps ensure that nodes can communicate easily, while high bandwidth helps shard movement and recovery. Modern data-center networking (1 GbE, 10 GbE) is sufficient for the vast majority of clusters.

Avoid clusters that span multiple data centers, even if the data centers are co-located in close proximity. Definitely avoid clusters that span large geographic distances.

2.1.4 Software Requirements

ElasticSearch

Java (JVM) Version

Elasticsearch is built using Java and includes a bundled version of OpenJDK from the JDK maintainers (GPLv2+CE) within each distribution. The bundled JVM is the recommended JVM and is located within the JDK directory of the Elasticsearch home directory.

Kibana

Packages of Kibana are provided for and tested against Linux, Darwin, and Windows. Since Kibana runs on Node.js, we include the necessary Node.js binaries for these platforms. Running Kibana against a separately maintained version of Node.js is not supported.

Logstash

Logstash requires Java 8 or Java 11. Use the official Oracle distribution or an open-source distribution such as OpenJDK.

2.2 Existing System

How do traditional system managers treat logs?

- **Cronjobs executed periodically**

  Cron is one of the most useful utilities that you can find in any Unix-like operating system. It is used to schedule commands at a specific time. These scheduled commands or tasks are known as "Cron Jobs". Cron is generally used for running scheduled backups, monitoring disk space, deleting files (for example log files) periodically which are no longer required, running system maintenance tasks and a lot more

  ○ Compute stats and send out report/alert

  ○ Detect possible abnormal behavior and react accordingly
- **Unix commands**

  -- grep + awk + sed + sort + Uniq + Perl + shell script ….
- **Searching with grep**

  One of the simplest ways to analyze logs is by performing plain text searches using grep. grep is a command-line tool that can search for matching text in a file, or in output from other commands. It's included by default in most Linux distributions and is also available for Windows and Mac.
- **Filtering and Parsing With awk**

  Filtering allows you to search on a specific field value instead of doing a full-text search. This makes your log analysis more accurate because it will ignore undesired matches from other parts of the log message. In order to search on a field value, you need to parse your logs first, or at least have a way of searching based on the event structure. To do this, we can use awk.

  Awk is a powerful command-line tool that provides a complete scripting language, so you can filter and parse out fields more effectively.
- **Set up one or more log servers for receiving logs from servers/routers/appliances.**
- **Plain text reports or stats trends webpage.**
- **Splunk**

  Splunk is a centralized logs analysis tool for machine-generated data, unstructured/structured, and complex multi-line data which provides the following features such as Easy Search/Navigate, Real-Time Visibility, Historical Analytics, Reports, Alerts, Dashboards, and Visualization.

2.2.1 Sample file logs

By implementing the existing system we get to know how grep and awk commands are being used for server log analysis to extract the particular important fields from the server log files.

Screenshot of the sample log file is shown below

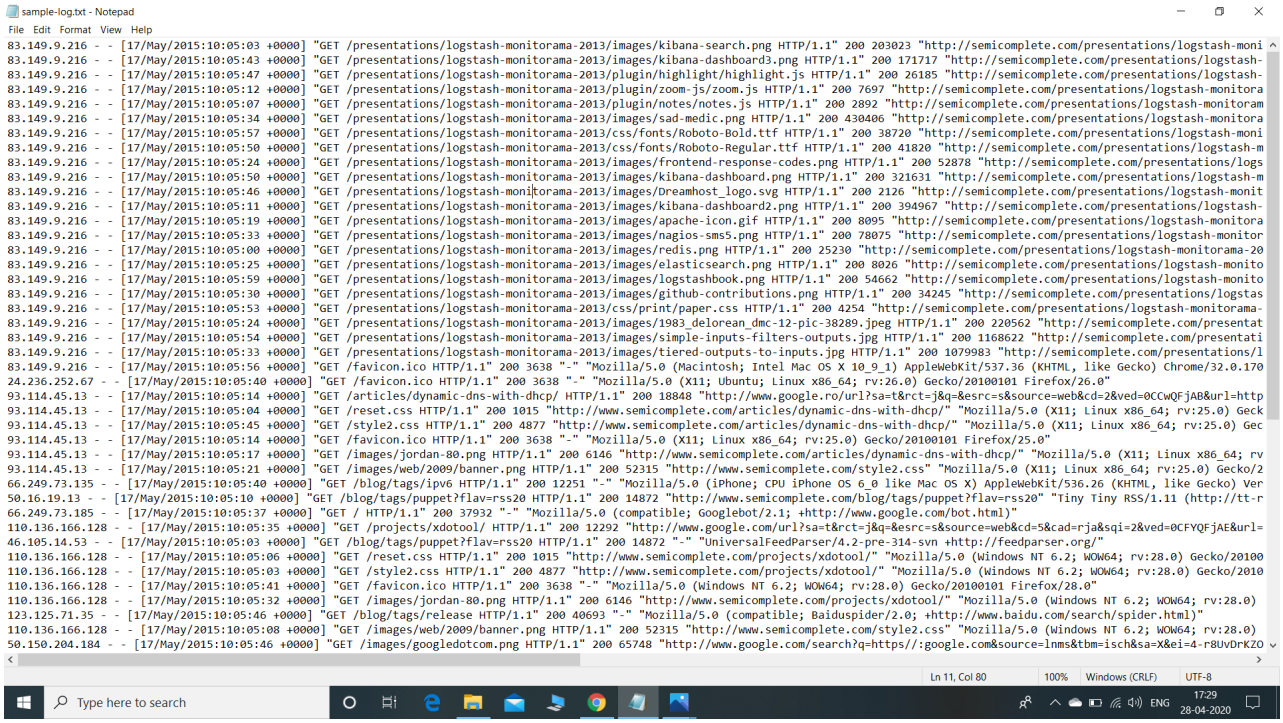

Figure 2.2.1.1 - Sample log file

Where the first field represents an IP address, the second field represents a timestamp, and so on.

2.2.2 GREP commands

1. To count the number of occurrences of the particular term:
   The command will count the number of occurrences of word GET in the log file. Here is the request server receives according to the user. We can also search for the other fields like a particular URL or many times if a particular URL is being hit by the client.

Figure 2.2.2.1 - Command and output of GREP

2.2.3 AWK commands

1. To view the which from which browser the URL request is being sent
   - Using awk command we can filter the requests coming from the Mozilla browser or any of the browsers and also we can view those log records.
   - Command and the corresponding output is as shown below figure.

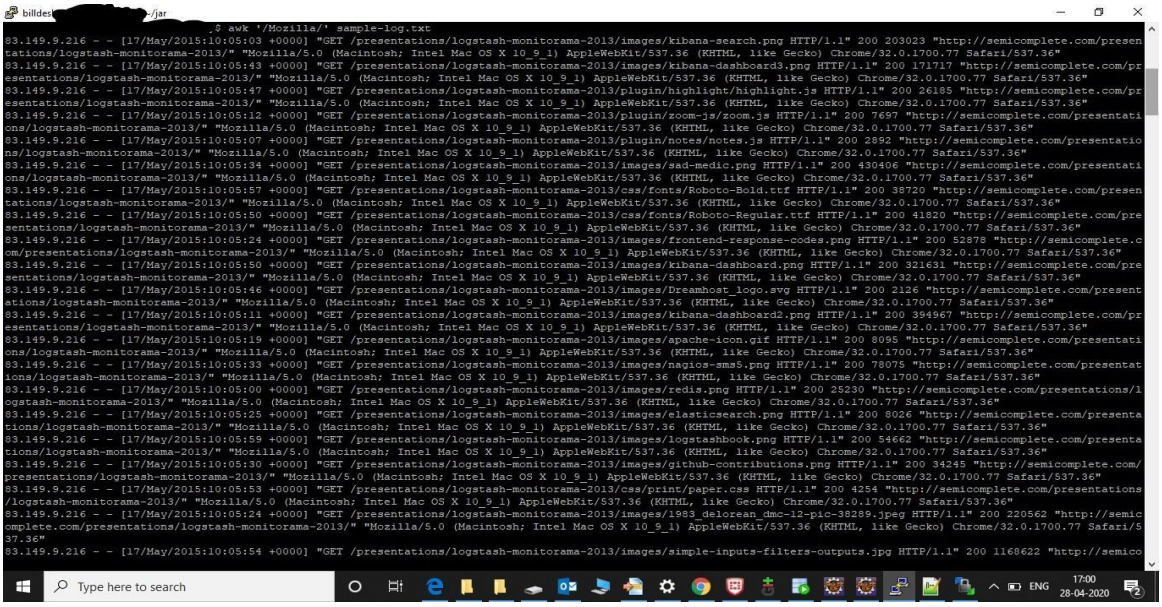

Figure 2.2.3.1 - Command and output of awk (searching for a particular term)

2. To view the status code given by the server
    - Whenever a server receives any requests it processes the request and generates the status code accordingly. Here status code 200 represents the received requests are successfully processed and response is given back to the user. Command and the corresponding output is as shown below fig

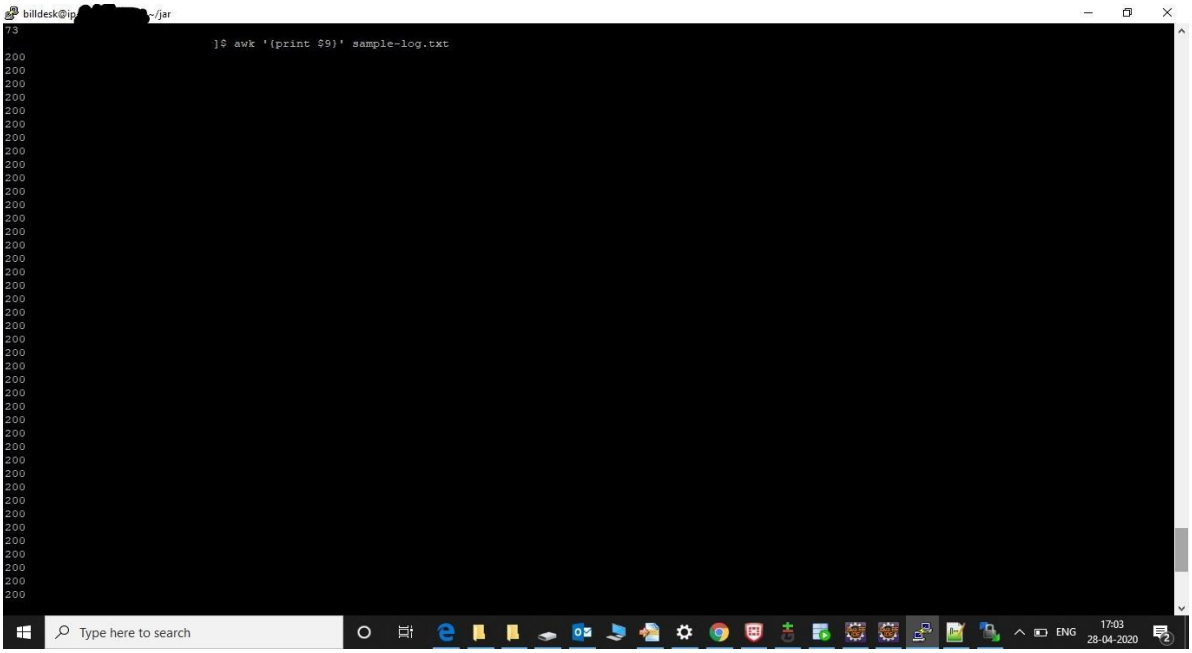

Figure 2.2.3.2 - Command and output of AWK (to find out status code of requests)

3. To see the IP address and URL of the incoming requests
    - We can extract the IP address and URL from the log files to monitor the requests coming from the clients and if anything goes wrong we can run the above command and see the IP address and URL.
    - Command and the corresponding output is as shown below fig

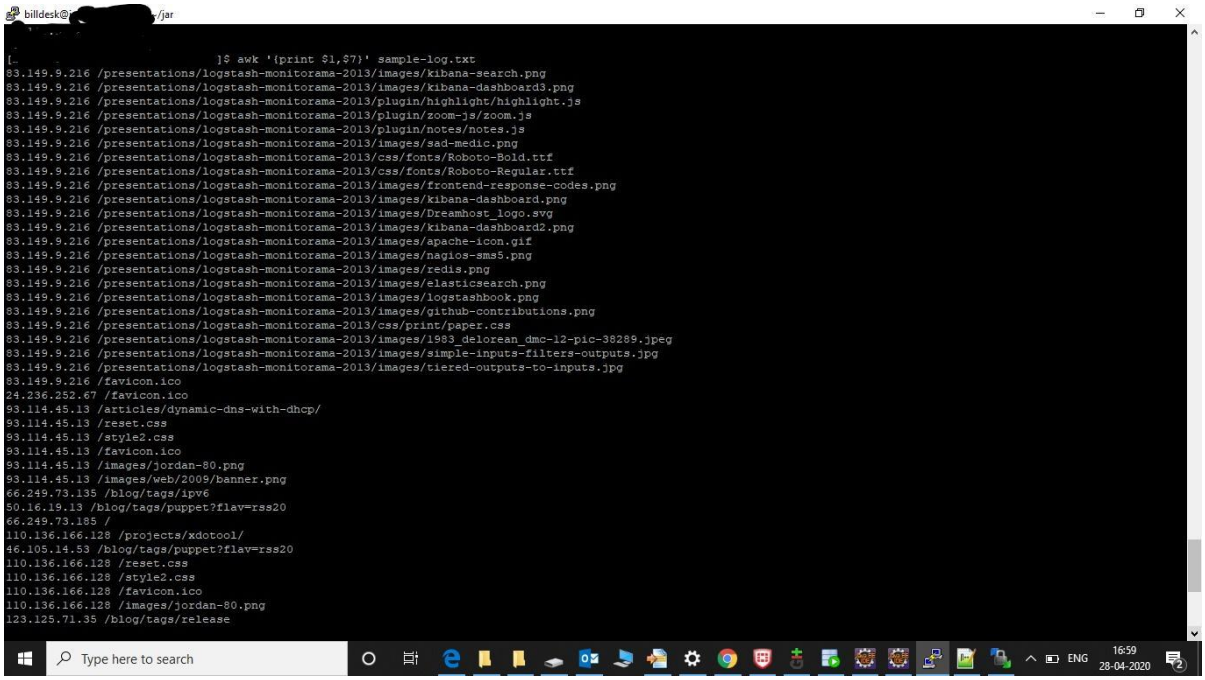

Figure 2.2.3.3 - Command and output of AWK (IP address and URL)

2.3 Current Problems and Traditional Methods

All information system platforms generate a large number of logs every day, usually based on streaming data, including user access records, database operation records, etc. When the amount of data reaches a certain order of magnitude, the traditional single-node system cannot complete the retrieval and analysis tasks. They must be processed using a distributed logging system. In general, these systems need to have the following characteristics:

1. Build a bridge between application systems and analysis systems
2. Support quasi-real-time online analysis systems and off-line analysis systems
3. Have high scalability and reliability, that is, when the number of data increases, the horizontal performance can be extended by adding nodes; when one or some nodes fail, only the performance of the system is affected, and the system functionality is not affected.

In a large-scale cluster system, logs are distributed and stored on different devices. The traditional method uses filtering tools such as cat, tail, sed, awk, grep to filter and output the analysis method because the efficiency is low and no longer applicable and also increase workload.

The most urgent task is to use a centralized log management platform to collect and collect logs from all servers for monitoring and analysis. The excellent system operation and maintenance platform can not only realize the centralized management of the components of the data platform, facilitate the daily monitoring of the system operation and maintenance personnel, improve the operation and maintenance efficiency, but also feedback the system operation status to the system developers.

Chapter 3
Design and Implementation

3.1 Design Consideration

- **Enterprise Search :**
  The system can be used to search for everything from anywhere. The system has unified the content of the server across the different platforms into a highly personalized and relevant search experience.
- **Observability :**
  The Elastic stack brings real-time metrics, logs, and APM traces into a single easily consulted view.
- **Scalability :**
  The system is highly scalable and can be scaled at any time according to the demand of the Company/Enterprise.
- **Security :**
  On top of the platform's internal security (index encryption, field-level security on documents) the SIEM app collects security information across the enterprise and provides richly detailed dashboards that allow close scrutiny of security operations.

3.2 Design Details

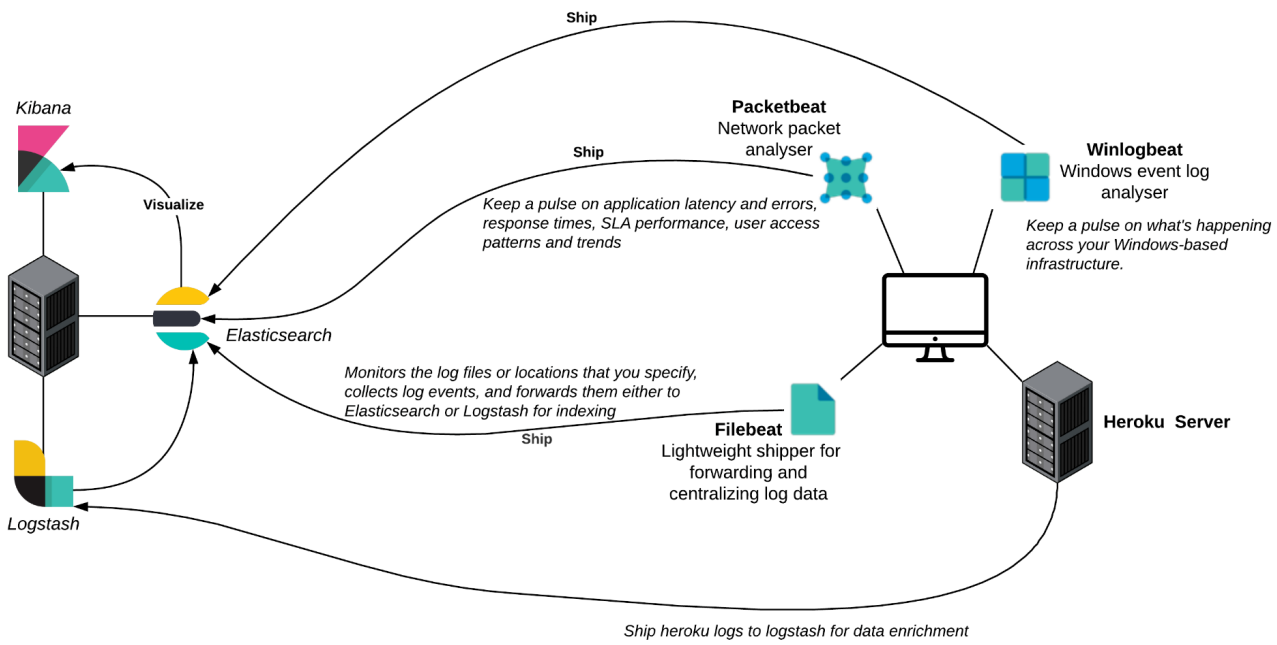

Figure 3.2.1 - Design details

The design of the Server Log Analysis project proceeds as depicted in the above flow diagram. The traditional way to do log management and analytics is to log in to systems (most likely the Linux systems) and use command line tools such as cat, tail, sed, awk, grep, etc. to filter and

output the data for analytics. This method could only process a small amount of data, and more often, sometimes developers who need to look at log files are not allowed to have certain server privileges. Now Elk Stack provides developers an efficient way to manipulate log data. Our design is trying to make good use of this amazing framework to get the information we need in a handy way. Since in this project our goal is to conduct security and log analytics on web applications, we design the whole system and select the ELK stack components based on our requirement and use cases. Our solution makes use of several most popular components of ELK stack, which have complete functionality that covers a ton of log and security analytics use cases.

Generally speaking, Logstash is mainly used to collect the logs sent by each Shipper, and then do some filtering and parsing functionality, and then send the processed data to Elasticsearch to store. Elasticsearch servers as a database, it is not similar to any relational database or non-relational database, it has its own way to store data, indexing data for further query and for the most important aggregation and analytics on data. Kibana is used as the user interface of Elasticsearch and the visualization tool for our input data. It provides a configurable user-friendly interface for visualization data that is stored and indexed in Elasticsearch. The shippers, in this case, are also Elk Stack components. Elastic provides a lot of different kinds of shippers called Beats for different use cases, for example, Filebeat for log files, Packetbeat for network data, Winlogbeat for Windows Event Logs.

Based on this design, we use Heroku server as a host to deploy our web application server. We can access the logs of our node.js server through Heroku dashboard and we can directly download the logs to our system and then as soon as logs files get downloaded logstash will ingest that data to elasticsearch for processing and visualisation.

3.3 Implementation

3.3.1 Log Analytics on Open-source Server Log files

Since the goals of our project mainly focus on utilizing the ELK Stack framework for log management and security analytics, the best way to get started is making good use of the public open-source log files. Typically, the web server applications generate different types of logs, including access logs, error logs, agent logs, and etc. Currently, the popular web servers or HTTP servers in the industry such as Tomcat, Nginx, and Apache are all automatically generating these multiple types of logs for the application deployed above it.

An HTTP server or web server is essentially a Layer 5 application of OSI network layers. It usually runs on the host server, binding server's IP address and listening on a TCP port to receive and process HTTP requests, so the clients such as Chrome, IE, Firefox can obtain web pages (HTML format), documents (PDF format), audio (MP4 format), video (MOV format) and many other things on the server through HTTP protocol. Typically, an access log message of an HTTP server can contain information as follows.

1. A time-frame indicating response time
2. An IP address indicating the client's IP
3. A HTTP Status Code indicating the status of the requesting resource
4. An URL of requesting resource
5. An user agent string that can be used as a identifier such as a computer's hostname or browser's version.

The open source Apache HTTP server we used are from a web blog. A typical message looks like as shown in below figure.

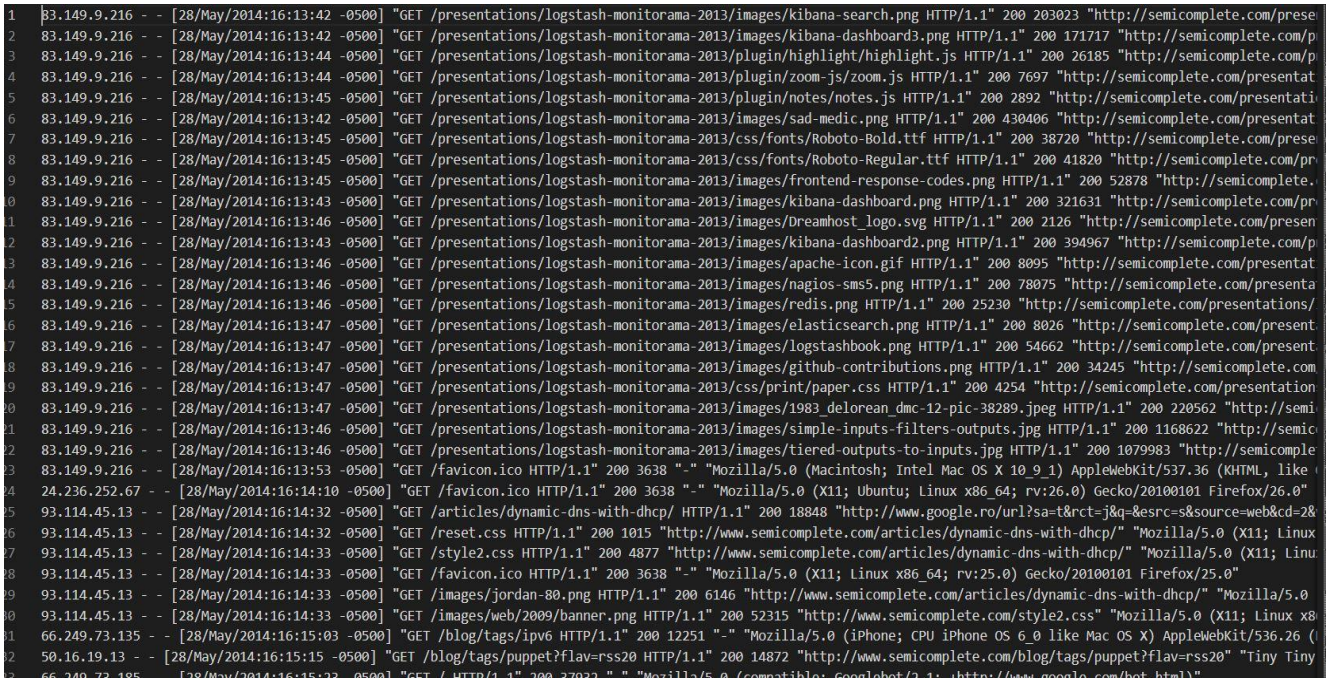

Figure 3.3.1.1 - Sample Apache Log file

3.3.1.1 ELK Stack Component Configuration

We use Filebeat as a log files collector and a light-weight shipper. Therefore, we configure Filebeat to read the log files from a certain location. The configuration details are specified in the filebeat.yml file.

The three main functions in the Logstash instance are event input, event data filtering, and event output. These three functions of Logstash are performed based on configuration information, which is stored in an easy-to-understand .conf file. There are different configuration sections in the .conf file that correspond to the three different types of plug-in, including inputs, filters, and outputs. Each Logstash instance is customized based on its requirements in the overall architecture.

There are multiple plug-ins we have used to configure Logstash, which are shown as the following Table.

Plug-in	Description
beats	receive events from the Elastic Beats framework
grok	parse unstructured log to be structured and queryable
mutate	perform general mutations on fields
date	parse dates from fields
geoip	add information about the geographical location of IPs
useragent	add information about user agent like operating system
stdout	a simple output which prints to the STDOUT
elasticsearch	output log data into Elasticsearch

Figure 3.3.1.1.1 - Logstash configuration

3.3.2 Security Analytics on Web Server Log

3.3.2.1 Heroku server web application log

In order to simulate the real-life production environment and to explore how ELK Stack can be utilized on the real-life production environment, we have built a real web application server and deployed it on Heroku server to let it run all the time to collect server log data. There data are streamed to a Elasticsearch endpoint to be indexed and eventually being visualized and analyzed. We then built a light-weight node.js web application based on a node.js project and to receive static requests. After we had setup the node.js application and successfully deployed it on heroku server we could access it by using the domain(https://sush-weather-app.herokuapp.com/) to send the https requests the client would receive response as shown in following figure,

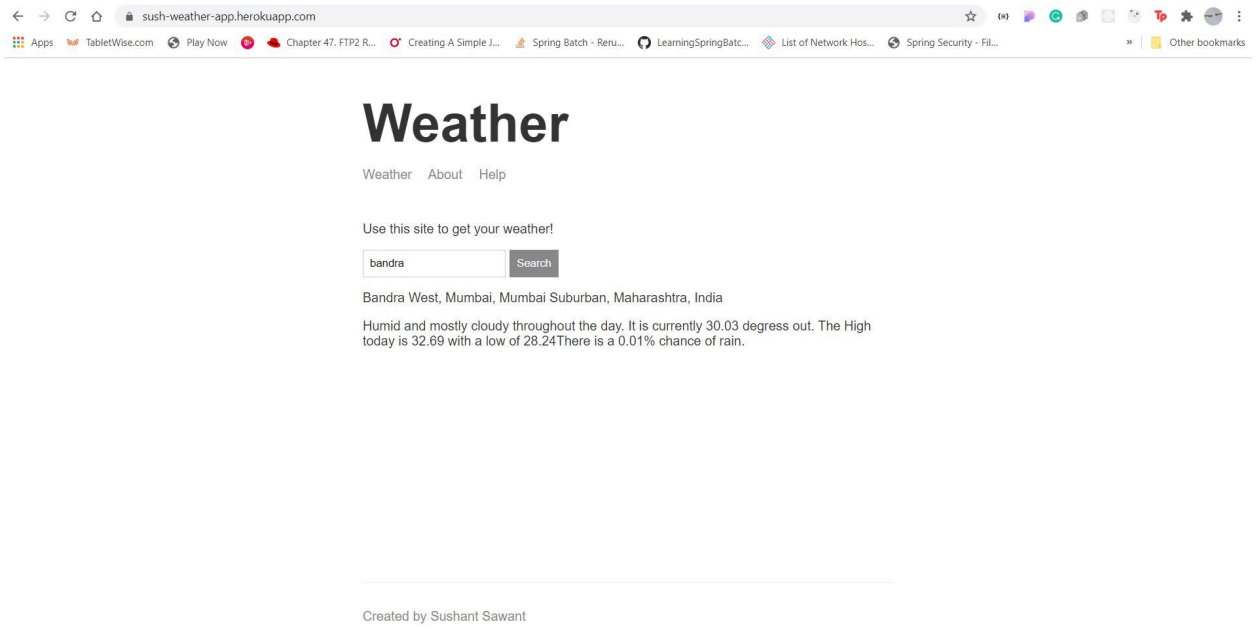

Figure 3.3.2.1.1 - Node.js Website

After we set up our web application for a while, we are able to gather some log files for further analytics, including the server access log , and finally they are streamed to the Logstash endpoint and Elasticseach endpoint. A typical access log message are shown as follows,

2021-05-10T13:48:38.472347+00:00 heroku[router]: at=info method=GET path="/about" host=sush-weather-app.herokuapp.com request_id=a8f4b821-a323-4bce-bc15-4ef1200b47ac fwd="27.4.4.181" dyno=web.1 connect=1ms service=9ms status=200 bytes=841 protocol=https

3.3.2.2 Logstash Data Enrichment for Security Use Cases

For security use cases, since the log data files we used are mostly web application server access logs, even though these data are simple, they still can provide a ton of security-related insights of the web applications and servers. Based on the features provided by Logstash data enrichment, our solution utilized some third parties data feeds, such as Malware Domain List and Blueliv threat data feeds.

By utilizing these free threat data feeds combined with ELK stack analytics functionality, we are able to gather security related insights from our web application server access logs.

3.3.2.3 Dataset Description

In our capstone project, in order to detect the DDOS attacks and other intrusions, we mainly use the following datasets to conduct our experiments:

1. Blueliv Data Set

This is a dataset created and maintained by Blueliv, that allows accessing Blueliv's Cyber-Threat Intelligence feeds. All the information in the dataset is the real data collected from enterprises. The crime servers data set contains the information of the malicious servers previously tried to launch an attack to other servers, which means it is controlled by the attackers or it was successfully attacked by the malicious servers and then became another source of the attack. The Bot IPs dataset contains the information of the botnet. Combining the Bot IPs dataset with the Crime Server Set, we can have a more complicated and complete data set to implement our experiment.[4]

2. Malware Domain Lists

This data set is widely used, we can use this data set to analyze the user's abnormal behavior. More specifically, we can try to find out is there anyone in our organization try to visit a known malware domain recorded in the dataset. And if we can find out the abnormal actions, we can know the visit to the malware domain is by mistake or is on purpose, and give out alerts and countermeasures if it is already controlled by the attacker.

3. Spamhaus Spam Database

Spamhaus is an international non-profit organization whose main task is to track Internet spam gangs, real-time blacklisting technology. Spamhaus has released a large number of spam organization databases, including SBL, XBL, and PBL. [5]

3.3.3 Malicious and Botnet IPs Visiting

Malware Domain List is a non-commercial community project, providing frequently updated threat data in a CSV format. It can be used to analyze malicious websites, domains, and URLs.

The first threat data feed we are using is the Malware Domain List, which provides threat intelligence data in a CSV format.Basically, it is a CSV file that includes thousands of line of threat messages, each message including a malicious URL or malicious domain name, a corresponding malicious IP address, a category of malicious type, a region code, and some other related information.In this project, we mostly focus on the malicious IP addresses in this list, anytime a malicious IP hitting or visiting our server, we are able to tag this request as a malicious visiting and alert or record it in our Elasticsearch endpoint, visualizing it in Kibana so that corresponding responsive actions could be taken.

Figure 3.3.3.1 - Malicious.yaml File

In order to make use of the Malware Domain List as a filter in the Logstash, we configure the .conf file of Logstash to use the CSV file as a filter. The CSV file is converted into a YAML file, in order to be best suitable for Logstash configuration, the YAML file includes the same information as the CSV file, most importantly the information about malicious IPs and domains.
It basically converts each CSV column into a key-value pair of a dictionary, and then outputs the dictionary in a YAML format. Also, it is worthy to mention that we use the translate plugin of Logstash to read the information from the YAML file, and if there is a coming request sent by a malicious IP or Botnet IP, we would tag this request as malicious and stored it in Elasticsearch index, which could be visualized in Kibana to notify any user or administrator in front of Kibana endpoint.The YAML file of Malware Domain List, in this case, could be seen as a dictionary file, every request sent by clients would be checked based on this dictionary file.
The data pattern indicates the information created and parsed by Logstash. In each message, there are fields indicating the client and the request information. We focus on security-related information, as our expectation by using threat data feeds, every time a malicious IP or a botnet IP visits our server it would be added a tag as a malicious URL, as shown in following Figure,

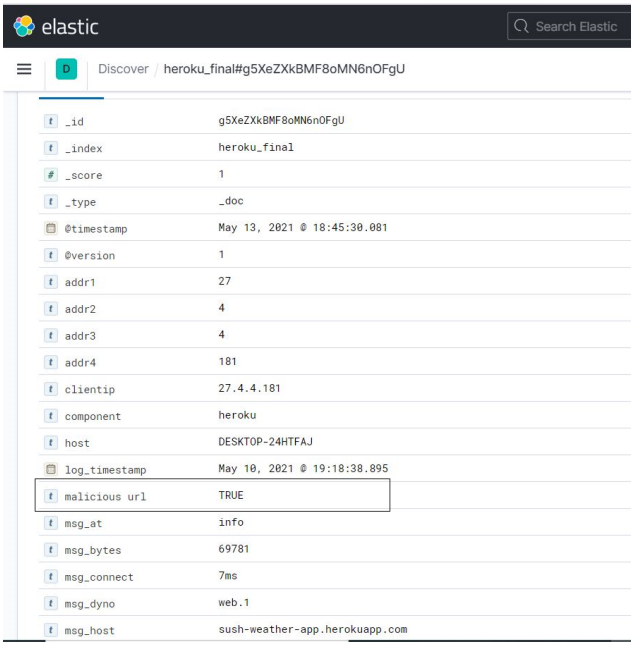

Figure 3.3.3.2 - Malicious URL field in Data

This field could be used in visualizing other charts or graphs. On the other hand, the normal requests, or the requests made by clients that did not exist on the malicious and botnet IPs data feed, would not have such a malicious URL field.

3.3.4 Crimeserver IPs Visiting

To achieve the goal that we will be able to detect any crimeserver hitting our web application server, we utilize another threat data feed: Blueliv threat data feeds. Blueliv is a cyber threat intelligence provider, providing services from Botnet information, crimeservers information, to malwares and attack patterns information.

It also provides a Logstash plugin for Logstash users. Based on its official documentation, we are able to use the plugin and configure it for the altering purpose of our web application server.The Logstash plugin provides services and its description are as shown in following Table,

Services	Type
Crimeservers	Partly Free
Botnet IPs	paid
Malwares	paid
Hacktivism	paid

Figure 3.3.4.1 - Plugin Services

For our project we have used crime servers which was free to use and provides following services,[4]

1. The free feed only reports crimeservers from open source sites.
2. Crime servers: Malware distribution domains, C and Cs, phishing, exploit kits and backdoors, ID, type, country, domain, geolocation, ASN ID, status.

After we configure the Logstash plugin accordingly, we get information from Logstash std output saying the threat data is read successfully, as shown in the following Figure,

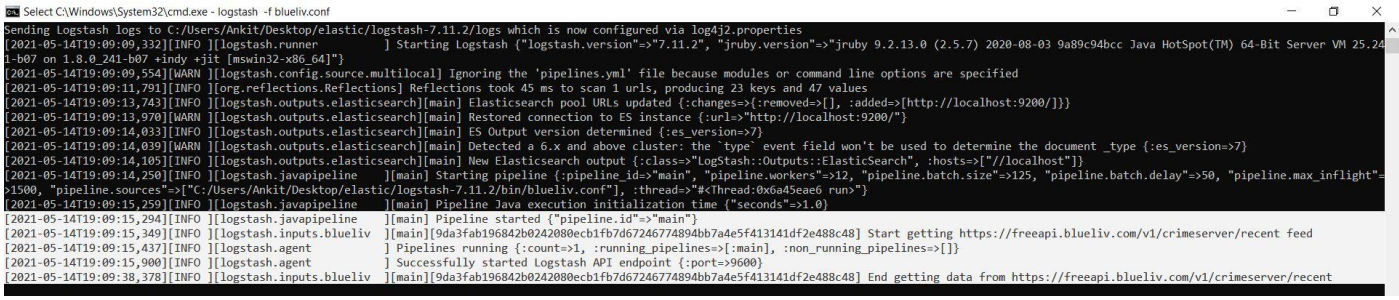

Figure 3.3.4.2 - Reading crimeserver threat data from blueliv

3.3.5 DNS Blocklist IPs Visiting and DDOS Attack

IP DNS Blocklist checks the IP address of the sending mail server against a public list of mail servers known to send spam. The DNS blocklist is actually a list of IP addresses that can be queried. The DNS query method is used to find out whether an A record of an IP address exists to determine whether it is included in the DNS blocklist.

Specifically, DNS Blacklist is a spam blocker that allows web administrators to block messages from specific systems that have a history of sending spams. As the name implies, these lists are Internet-based domain name systems that convert digital IP addresses (such as 66.171.248.182) into domain names like example.net, making the list easier to read, use, and search. If the maintainer of the DNS blacklist has received any type of spam from a particular domain in the past, the server will be blacklisted and all messages sent from it will be flagged or rejected. Therefore, DNSBL is a list of a large number of IP addresses that are considered to belong to a mail server that sends or forwards a large amount of spam. [5]

In our project, we also make good use of this list, which provides us great insights about whether our web application server is continuously visited by any spam server.

We utilize the data feed from zen.spamhaus.org, which is a database of IP addresses from which does not recommend the acceptance of electronic mail. We configure the Logstash to use the DNS filter to read data from zen.spamhaus.org, and if there is a match found, then we add an extra field to our indexed data indicating it is a spam address request. As shown in following Figure

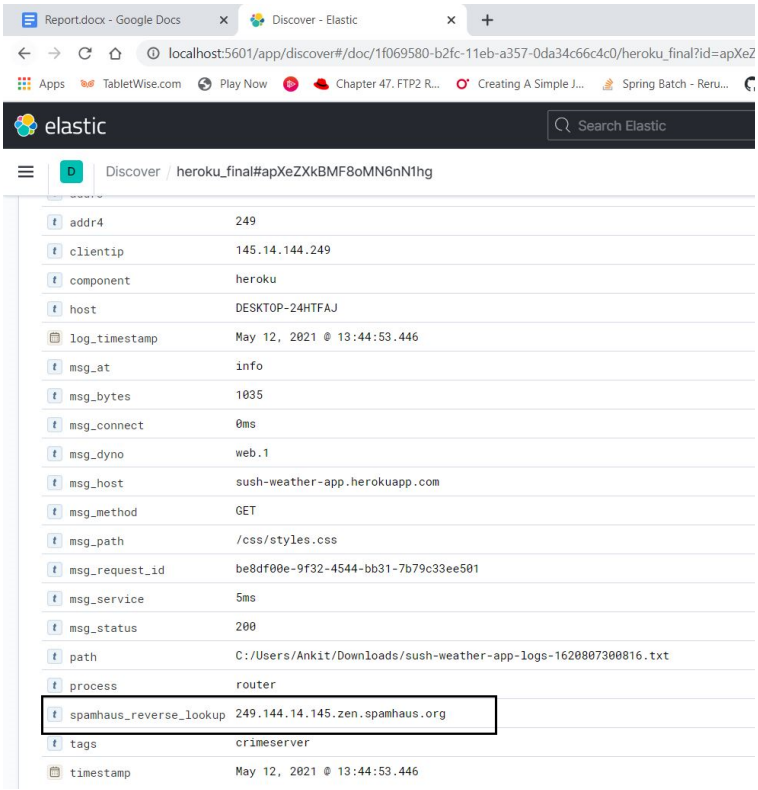

Figure 3.3.5.1 - Spam checking field

For the configuration of the Logstash DNS filter, the details are shown as following Table,

DNS Filter Option	Value
resolve	spamhaus_reverse_lookup
nameserver	10.0.1.1
add_tag	dns_successful_lookup
action	replace

Figure 3.3.5.2 - DNS plugin

Spamhaus defines a specific way of checking, which is the reversed IP checking. Therefore, we have to convert the IP address in each message to the reversed format, and then send it to zen.spamhaus.org database to check to see whether it is spam or not. If the answer is yes, the zen.spamhaus.org would return a special return address, which is 127.0.0.2, to indicate there is a matching happen. And then, we add a field to our data to be indexed. After we finish indexing data into Elasticsearch, we can set up some limits and alerts in Kibana to indicate some abnormal situations happening, such as servers under the DDOS attack. Kibana provides a lot of methods and options for users to create a threshold alert. We are able to utilize these functions to detect a potential DDOS attack happening, and the information would be sent to the email address we provided in Kibana.

3.3.6 Network and Windows Log Analysis

What is Beats?

The Elastic Stack expands the capabilities of Elasticsearch by adding extremely useful tooling to work alongside Elasticsearch. One of the most useful of these tools is the Beats ecosystem. Beats are essentially lightweight, purpose-built agents that acquire data and then feed it to Elasticsearch.

The magic of Beats is the libbeat framework that makes it easy to create customized beats for any type of data you'd like to send to Elasticsearch. Due to that flexibility, the number of Beats available and the capabilities of Beats overall are rapidly expanding.

Beats can send data directly to Elasticsearch or via Logstash, where you can further process and enhance the data, before visualizing it in Kibana.[6]

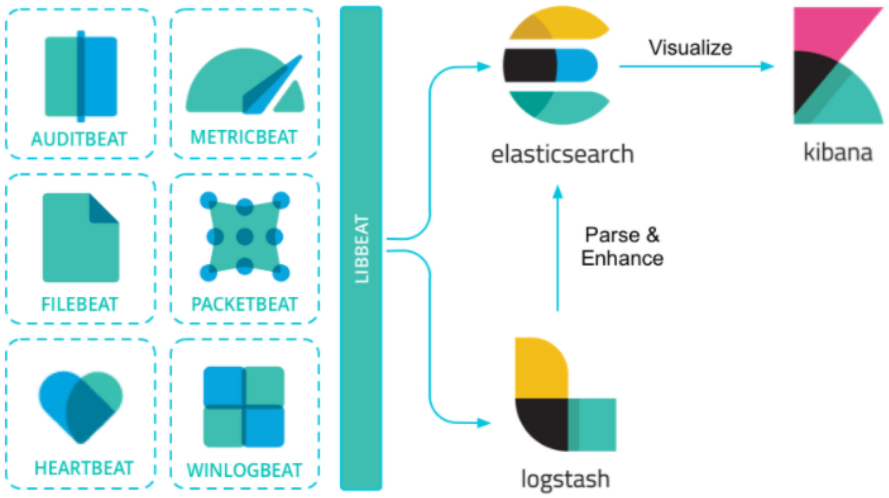

Figure 3.3.6.1 - Overview of Beats

While each beat has its own distinct use, they all solve the common problem of gathering data at its source and making it easy and efficient to ship that data to Elasticsearch.

Examples of Beats

There are currently six official Beats from Elastic:[6]

- **Filebeat**
  Filebeat is designed to read files from your system. It is particularly useful for system and application log files, but can be used for any text files that you would like to index to Elasticsearch in some way.
- **Metricbeat**
  As the name implies, Metricbeat is used to collect metrics from servers and systems. It is a lightweight platform dedicated to sending system and service statistics.
- **Packetbeat**
  Packetbeat, a lightweight network packet analyzer, monitors network protocols to enable users to keep tabs on network latency, errors, response times, SLA performance, user access patterns and more.

- **Heartbeat**
  Heartbeat is a lightweight shipper for uptime monitoring. It monitors services basically by pinging them and then ships data to Elasticsearch for analysis and visualization. Heartbeat can ping using ICMP, TCP and HTTP.
- **Winlogbeat**
  Winlogbeat is a tool specifically designed for providing live streams of Windows event logs. It can read events from any Windows event log channel, monitoring log-ons, log-on failures, USB storage device usage and the installation of new software programs.
- **Auditbeat**
  Auditbeat performs a similar function on Linux platforms, monitoring user and process activity across your fleet. Auditd event data is analyzed and sent, in real time, to Elasticsearch for monitoring the security of your environment.

Which beats to consider?

Packetbeat Overview

- Packetbeat is a real-time network packet analyzer that you can use with Elasticsearch to provide an application monitoring and performance analytics system. Packetbeat completes the Beats platform by providing visibility between the servers of your network.
- Packetbeat works by capturing the network traffic between your application servers, decoding the application layer protocols (HTTP, MySQL, Redis, and so on), correlating the requests with the responses, and recording the interesting fields for each transaction.

Packetbeat sniffs the traffic between your servers, parses the application-level protocols on the fly, and correlates the messages into transactions. Currently, Packetbeat supports the following protocols:

- ICMP (v4 and v6)
- DHCP (v4)
- DNS
- HTTP
- AMQP 0.9.1
- Cassandra
- Mysql
- PostgreSQL
- Redis
- Thrift-RPC
- MongoDB
- Memcache
- NFS
- TLS
- SIP/SDP (beta)

Winlogbeat Overview

There's a lot to learn from your Windows event logs. Interested in security events like logon successes (4624) and failures (4625)? How about when a storage device is attached (4663) or a

new service is installed (4798)? Winlogbeat can be configured to read from any event log channel, giving you access to the Windows data you need most.

Winlogbeat can capture event data from any event logs running on your system. For example, you can capture events such as:

- Application events
- Hardware events
- Security events
- System events

Processor and GEO IP Processor

Pipeline

A pipeline is a definition of a series of processors that are to be executed in the same order as they are declared. A pipeline consists of two main fields: a description and a list of processor.[7]

Processor

Each processor defines its own configuration parameters, but all processors have the ability to declare tag, on_failure and if fields. These fields are optional.A tag is simply a string identifier of the specific instantiation of a certain processor in a pipeline. The tag field does not affect the processor's behavior, but is very useful for bookkeeping and tracing errors to specific processors.

GEO IP processor

The geoip processor adds information about the geographical location of IP addresses, based on data from the Maxmind databases. This processor adds this information by default under the geoip field. The geoip processor can resolve both IPv4 and IPv6 addresses.

```
PUT _ingest/pipeline/geoip-info
{
  "description": "Add geoip info",
  "processors": [
    {
      "geoip": {
        "field": "client.ip",
        "target_field": "client.geo",
        "ignore_missing": true
      }
    },
    {
      "geoip": {
        "field": "source.ip",
        "target_field": "source.geo",
        "ignore_missing": true
      }
    },
    {
      "geoip": {
        "field": "destination.ip",
        "target_field": "destination.geo",
        "ignore_missing": true
      }
    },
    {
      "geoip": {
        "field": "server.ip",
        "target_field": "server.geo",
        "ignore_missing": true
      }
    }
  ]
}
```

Figure 3.3.6.2 - GEO IP processor

Packetbeat config:

Configure Packetbeat

To configure Packetbeat, edit the configuration file. The default configuration file is called packetbeat.yml. The location of the file varies by platform.[8]

Configure traffic capturing options and network flows

You can configure Packetbeat to collect and report statistics on network flows. A *flow* is a group of packets sent over the same time period that share common properties, such as the same source and destination address and protocol. You can use this feature to analyze network traffic over specific protocols on your network.

```
# ============================== Network device ================================

# Select the network interface to sniff the data. On Linux, you can use the
# "any" keyword to sniff on all connected interfaces.
packetbeat.interfaces.device: 0


# ================================== Flows =====================================

# Set `enabled: false` or comment out all options to disable flows reporting.
packetbeat.flows:
  # Set network flow timeout. Flow is killed if no packet is received before being
  # timed out.
  timeout: 30s

  # Configure reporting period. If set to -1, only killed flows will be reported
  period: 10s
```

Figure 3.3.6.3 - Network flow configuration (Packetbeat)

Configure which transaction protocols to monitor:

Capture DNS traffic

The DNS section of the packetbeat.yml config file specifies configuration options for the DNS protocol. The DNS protocol supports processing DNS messages on TCP and UDP. Here is a sample configuration section for DNS.

```
packetbeat.protocols:
- type: dns
  ports: [53]
  include_authorities: true
  include_additionals: true
```

Figure 3.3.6.4 - DNS traffic configuration (Packetbeat)

Capture HTTP traffic

The HTTP protocol has several specific configuration options. Here is a sample configuration for the http section of the packetbeat.yml config file:

```
packetbeat.protocols:
- type: http
  ports: [80, 8080, 8000, 5000, 8002]
  hide_keywords: ["pass", "password", "passwd"]
  send_headers: ["User-Agent", "Cookie", "Set-Cookie"]
  split_cookie: true
  real_ip_header: "X-Forwarded-For"
```

Figure 3.3.6.5 - HTTP traffic configuration (Packetbeat)

Capture TLS traffic

TLS is a cryptographic protocol that provides secure communications on top of an existing application protocol, like HTTP or MySQL.Packetbeat intercepts the initial handshake in a TLS connection and extracts useful information that helps operators diagnose problems and strengthen the security of their network and systems.

```
packetbeat.protocols:
- type: tls
  send_certificates: true
  include_raw_certificates: false
  include_detailed_fields: true
  fingerprints: [ md5, sha1, sha256 ]
```

Figure 3.3.6.6 - TLS traffic configuration (Packetbeat)

Configure the Packetbeat output:

The Elasticsearch output sends events directly to Elasticsearch using the Elasticsearch HTTP API.

```
# ---------------------------- Elasticsearch Output ----------------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]
  pipeline: geoip-info
```

Figure 3.3.6.7 - Packetbeat output to ElasticSearch (direct)

The Logstash output sends events directly to Logstash by using the lumberjack protocol, which runs over TCP. Logstash allows for additional processing and routing of generated events.
If you want to use Logstash to perform additional processing on the data collected by Packetbeat, you need to configure Packetbeat to use Logstash.
To do this, you edit the Packetbeat configuration file to disable the Elasticsearch output by commenting it out and enable the Logstash output by uncommenting the Logstash section:

```
output.logstash:
  hosts: ["127.0.0.1:5044"]
```

Figure 3.3.6.8 - Packetbeat output to ElasticSearch (via Logstash)

Winlogbeat config

Configure Winlogbeat

To configure Winlogbeat, edit the configuration file. The default configuration file is called winlogbeat.yml. The location of the file varies by platform. To locate the file, see Directory layout.[9]

Modules

- PowerShell module
    - The PowerShell module processes event log records from the Microsoft-Windows-PowerShell/Operational and Windows PowerShell logs.
- Security module
    - The security module processes event log records from the Security log

Configure the ElasticSearch output:
The Elasticsearch output sends events directly to Elasticsearch using the Elasticsearch HTTP API.
Example configuration:

```
output.elasticsearch:
  hosts: ["https://myEShost:9200"] 1
```

Figure 3.3.5.1 - Winlogbeat output to ElasticSearch (direct)

Configure the Logstash output:
The Logstash output sends events directly to Logstash by using the lumberjack protocol, which runs over TCP. Logstash allows for additional processing and routing of generated events.If you want to use Logstash to perform additional processing on the data collected by Winlogbeat, you need to configure Winlogbeat to use Logstash.

```
output.logstash:
  hosts: ["127.0.0.1:5044"]
```

Figure 3.3.5.2 - Winlogbeat output to ElasticSearch (via Logstash)

Chapter 4
Result and Analysis

4.1 Results and analysis of Apache Open-source Server log files

Based on the configuration of Logstash and Filebeat, log files are parsed and analyzed and then indexed into Elasticsearch for analytics and aggregation. We utilize the Kibana as the user interface to communicate with Elasticsearch.

Based on the enriched information of the GeoIP plugin, We are able to analyze and figure out the countries that sent the most server requests and which cities in each country sent the most requests. And the specific number of requests sent by these countries and cities. This information is presented directly to the user in the form of a histogram, allowing the user to clearly understand the useful information, as shown following Figure,

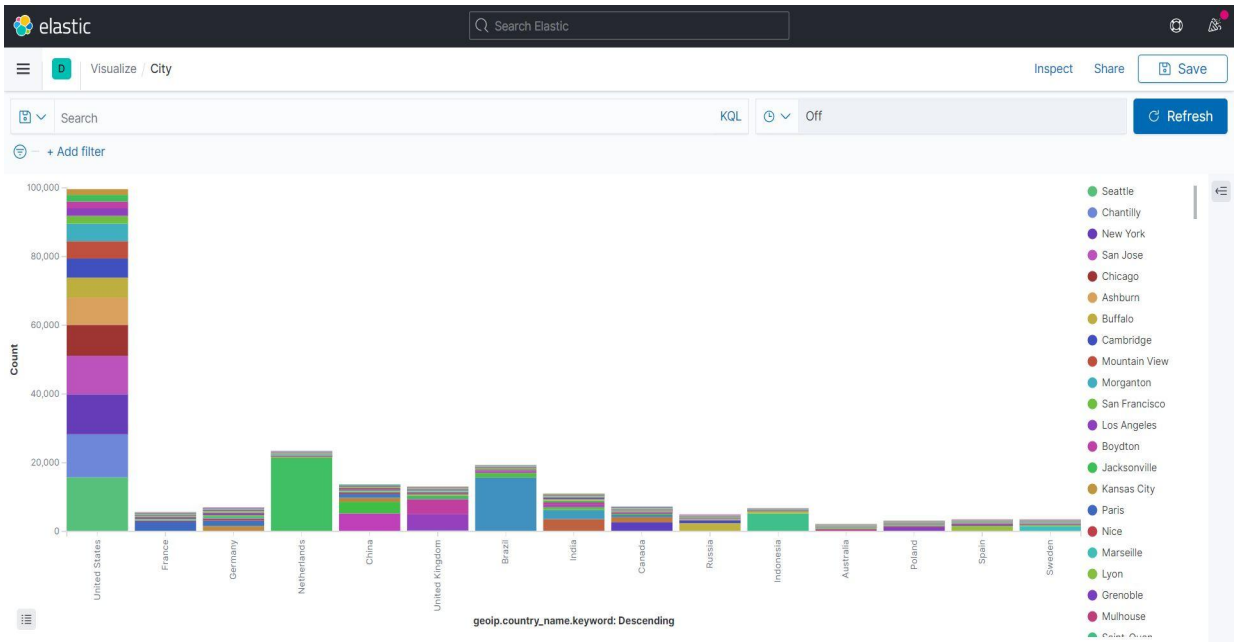

Figure 4.1.1 - Visualization of cities on Apache logs

Furthermore, we are able to analyze the historical graph indicating the day and the time that have the most clients visiting the web server. All the charts and graphs could be arranged to be one dashboard and can be customized to fit the users use cases and requirements. As shown in following Figure,

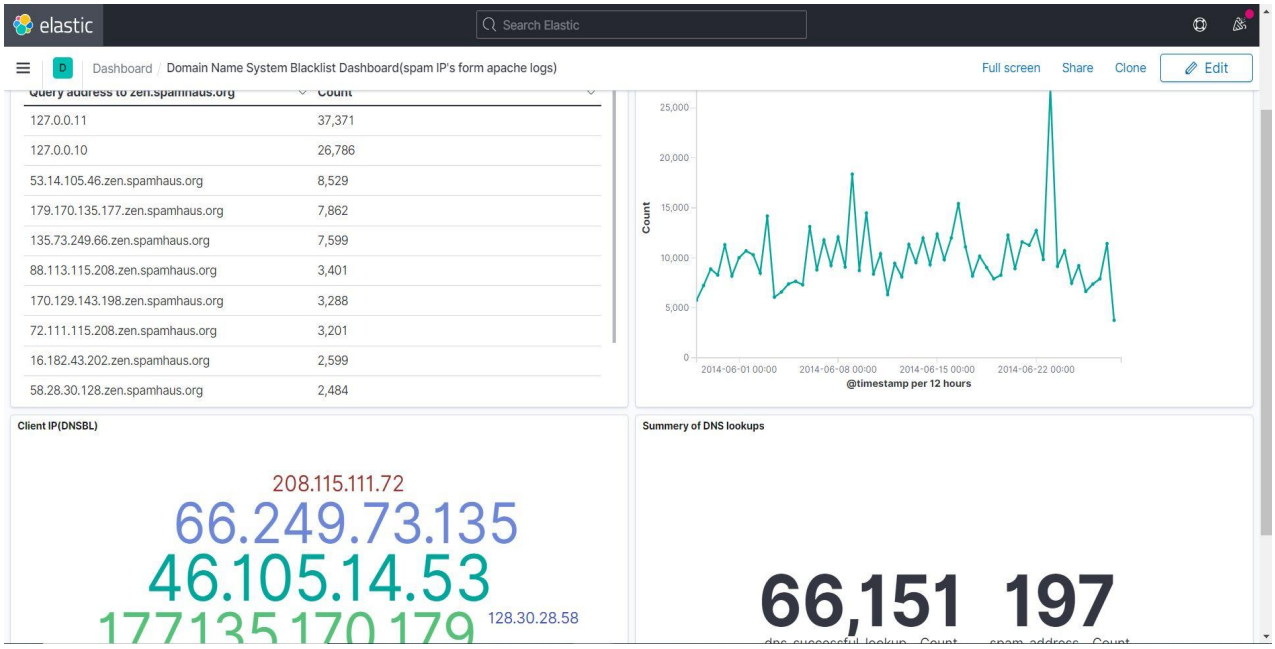

Figure 4.1.2 - Dashboard of Apache log files

4.2 Results and analysis of security related aspects of node.js web server

By using Kibana, we can create and customize dashboards to visualize our data. As we are gathering threat data such as crimeservers data from third parties information provider, we can also visualize that data using Kibana, In the dashboard, we customize it to show the status of crime events, and the types of crime servers, which are all can be seen clearly in our dashboard

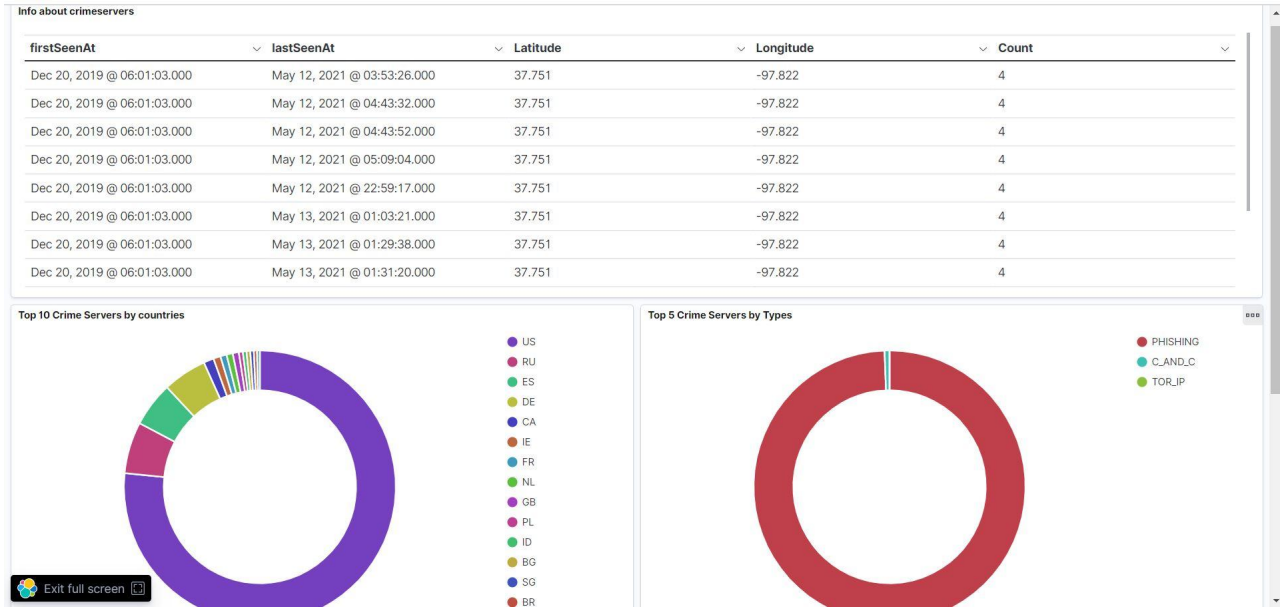

Figure 4.2.1 - Blueliv Dashboard 1

Figure 4.2.2 - Blueliv Dashboard 2

As the configuration we set-up for ELK stack components, we are able to visualize our access logs in Kibana. Based on the access logs we collected and indexed, we can visualization these data and conduct security analytics on it, as shown in the following Figure

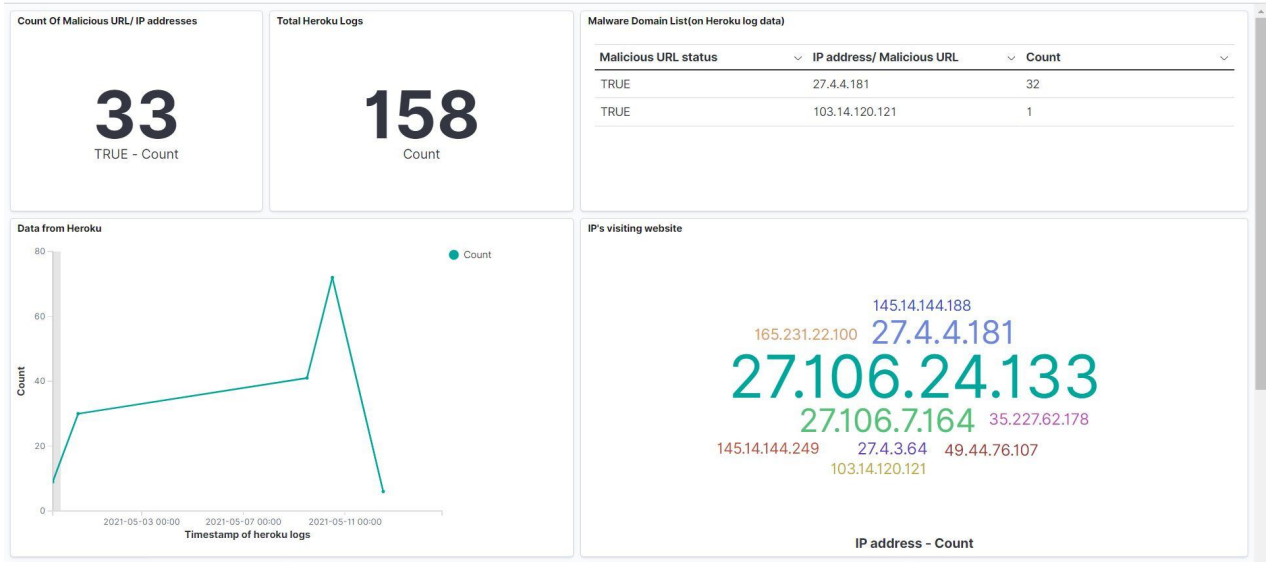

Figure 4.2.3 - Heroku Dashboard 1

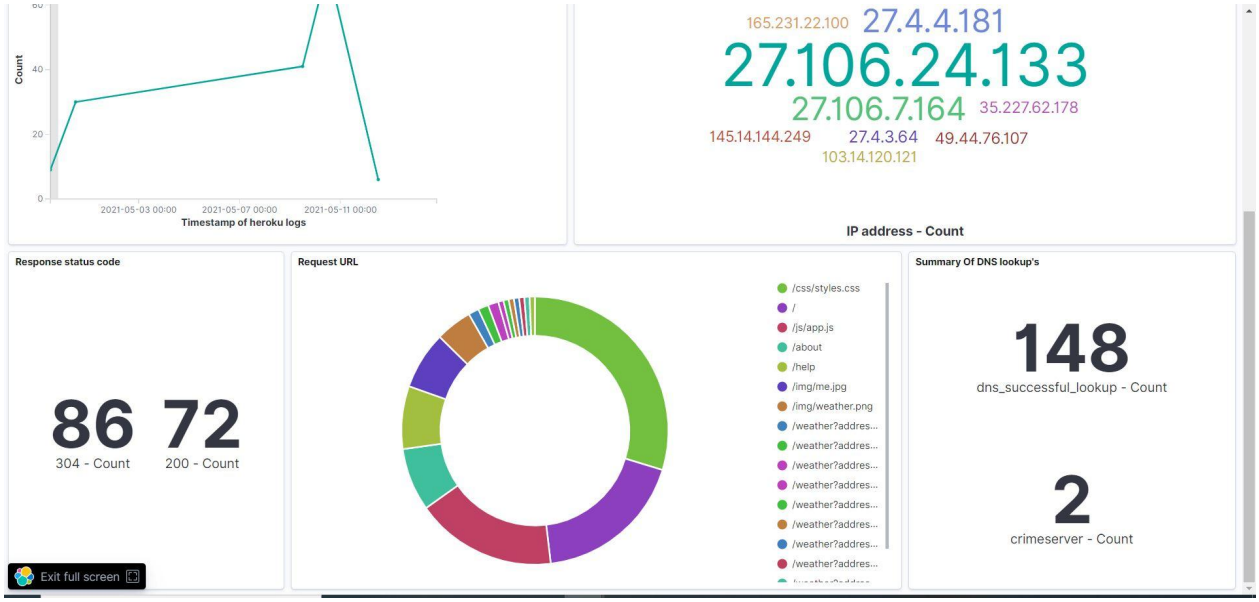

Figure 4.2.4 - Heroku Dashboard 2

As it is shown in the above figure, the crimeservers that hit our web application server are all captured and visualized in our customized dashboard. Based on this information, web application server administrators can be aware of the status of the server they control and take any necessary actions against potential attacks or intrusion.

We can also conduct searches in Kibana, as we configure our Logstash, we are able to tag any malicious IP or Botnet IP that are visiting or have visited our server, these requests would be marked as malicious URL visitings. We can list all the matched results that have the tag, which provides administrators a lot of valuable information regarding these potential intrusions and allows them to conduct further research on the information.

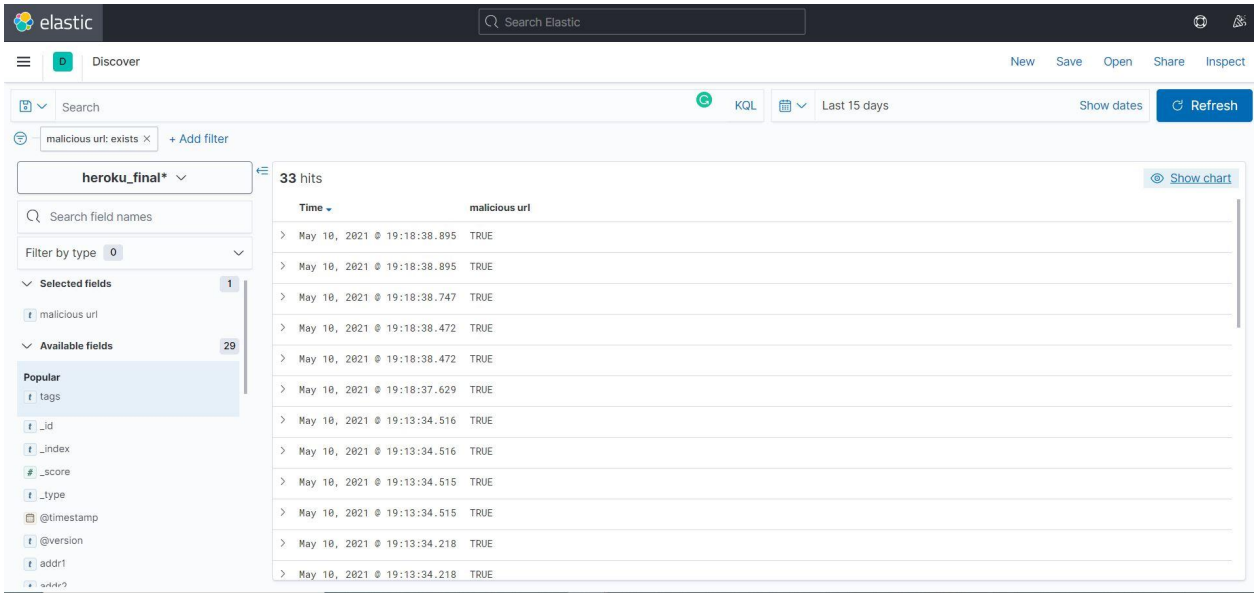

Figure 4.2.5 - Malicious URL Match

4.3 Results of dashboard using Packetbeat

After successfully setting up the packetbeat with ElasticSearch and Kibana, we were able to gather different **1058 fields** in the ElasticSearch and we built visualization using these fields. At this time we have around **75903+ unique documents** which contain the data about the network flows and different events collected from the Packetbeat and stored in the ElasticSearch database.

We have created the following visualizations in Kibana from the data generated from Packetbeat which enables us to monitor the data/events related to the network of the server.

- This dashboard consists of the navigation bar to navigate across the other dashboards and other general information related to the network flow and events like server location from on the coordinate map using the data obtained from the GEO IP processor and other basic visualizations.

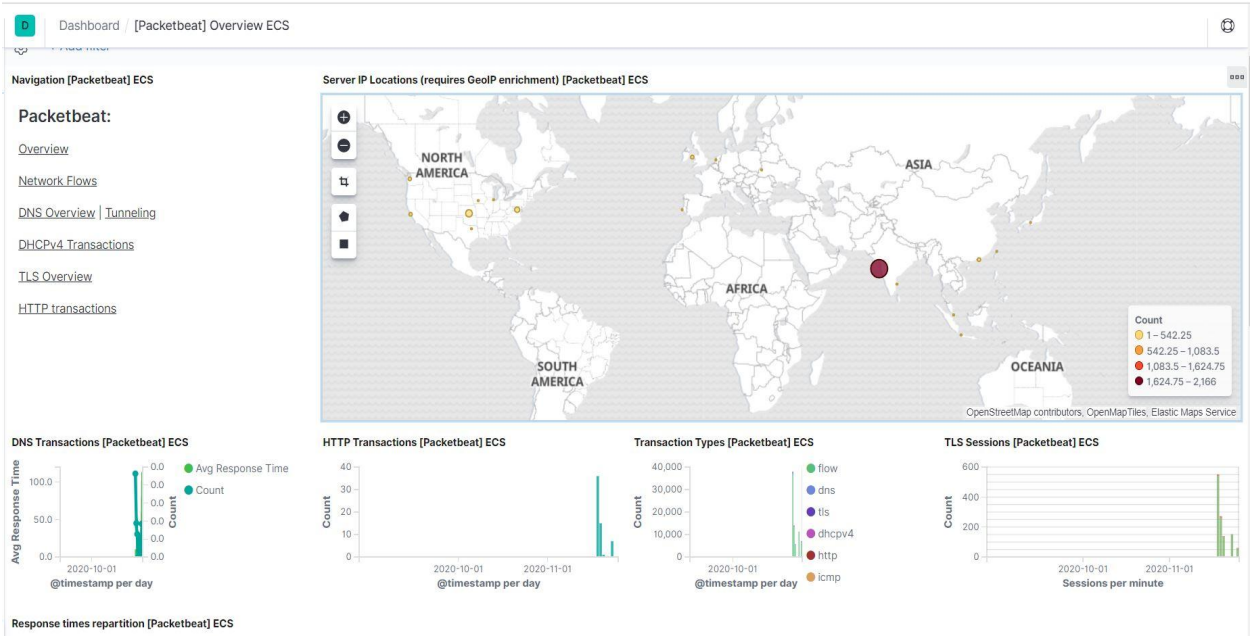

Figure 4.3.1 - Packetbeat dashboard (Overview)

- This dashboard consists of information in the form of the visualization about DNS. It consists of the information about DNS query summary, DNS question type, DNS response codes and information about DNS tunneling.

Figure 4.3.2 - DNS Overview

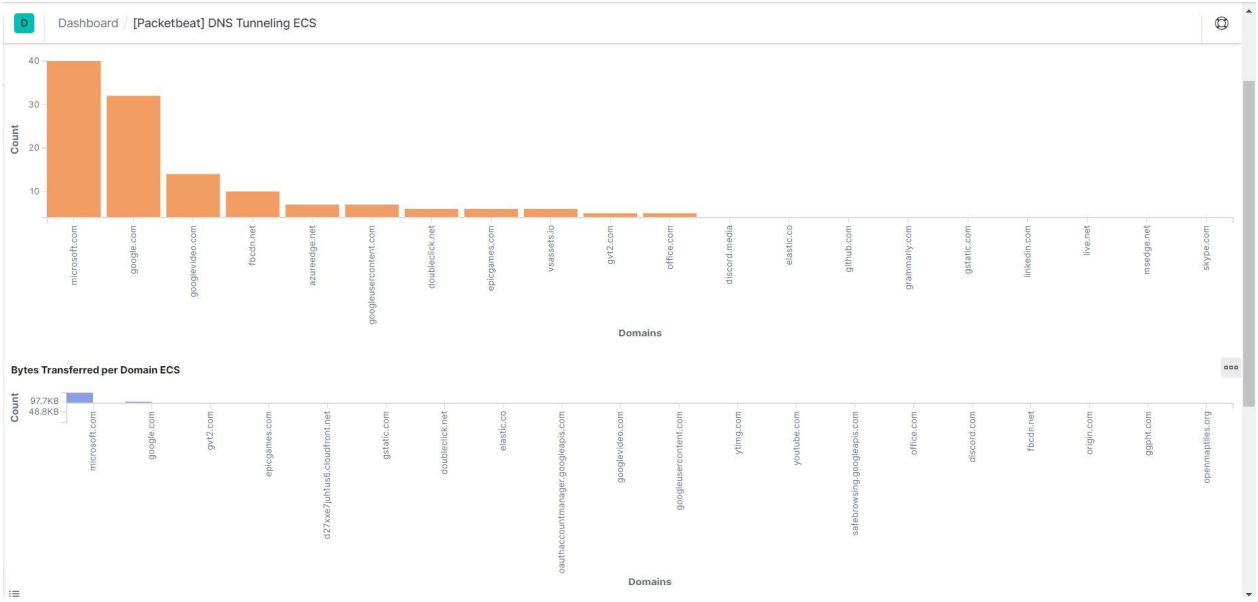

Figure 4.3.3 - DNS Tunneling

- This dashboard consists of information about TLS overview, total number of TLS sessions,TLS server name, TLS server certificates name and other important information about TLS protocol.

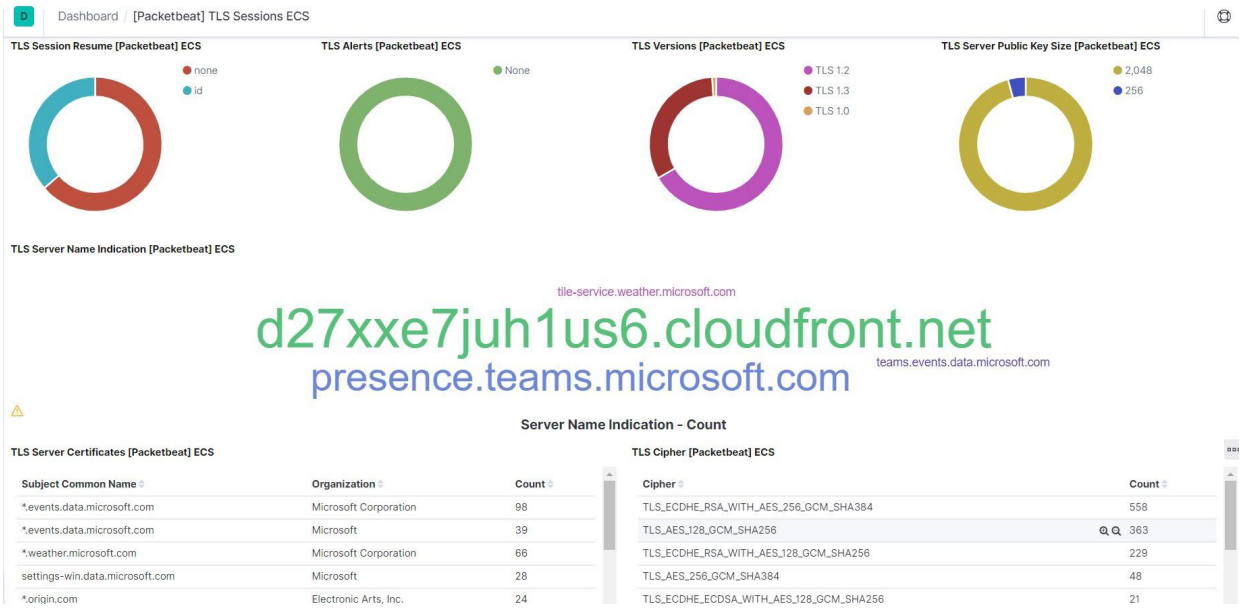

Figure 4.3.4 - TLS sessions Overview

- This dashboard contains information about http transactions i.e total number of HTTP transactions, status code returned for top queries and top 10 HTTP requests.

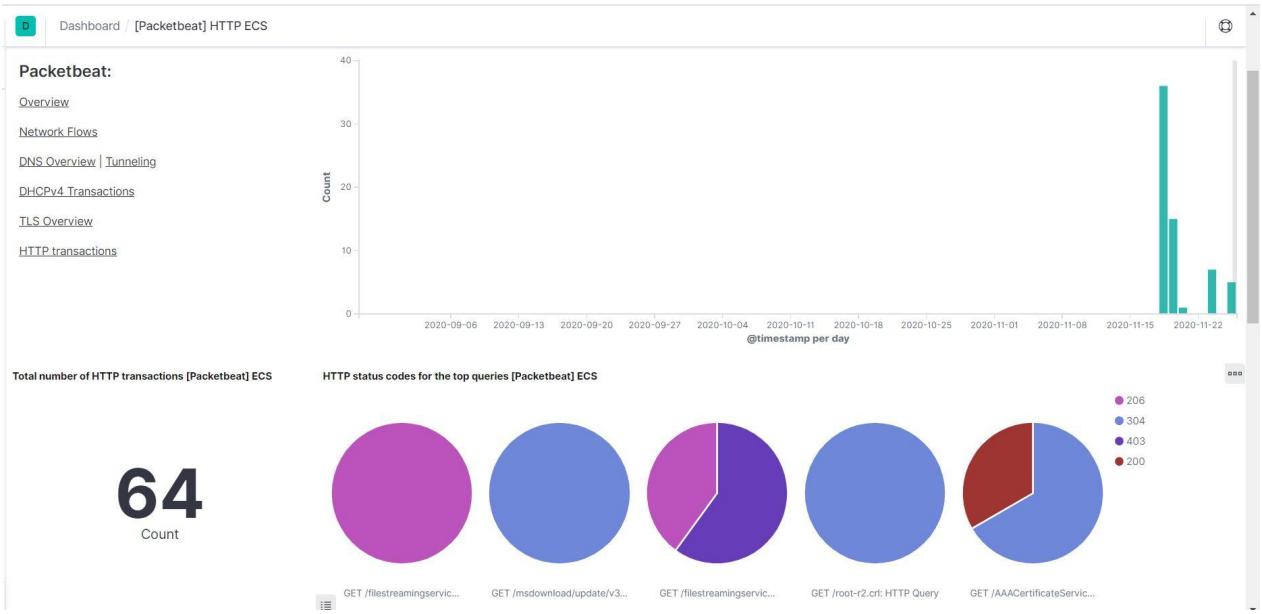

Figure 4.3.5 - HTTP Overview

4.4 Results of dashboard using Winlogbeat

After successfully setting up the Winlogbeat with ElasticSearch and Kibana, we were able to gather different **1066 fields** in the ElasticSearch and we built visualization using these fields. At this time we have around **161488+ unique documents** which contain the data about the Windows event logs like security events,system events and application events collected from the Winlogbeat and stored in ElasticSearch database.

We have created the following visualizations in Kibana from the data generated from Winlogbeat which enables us to monitor the data/events related to the complete windows events.

- This Dashboard helps us to monitor in general windows events realtime. It consists of different visualizations which show the total number of the events and pie-chart according to sources (providers) for windows events.

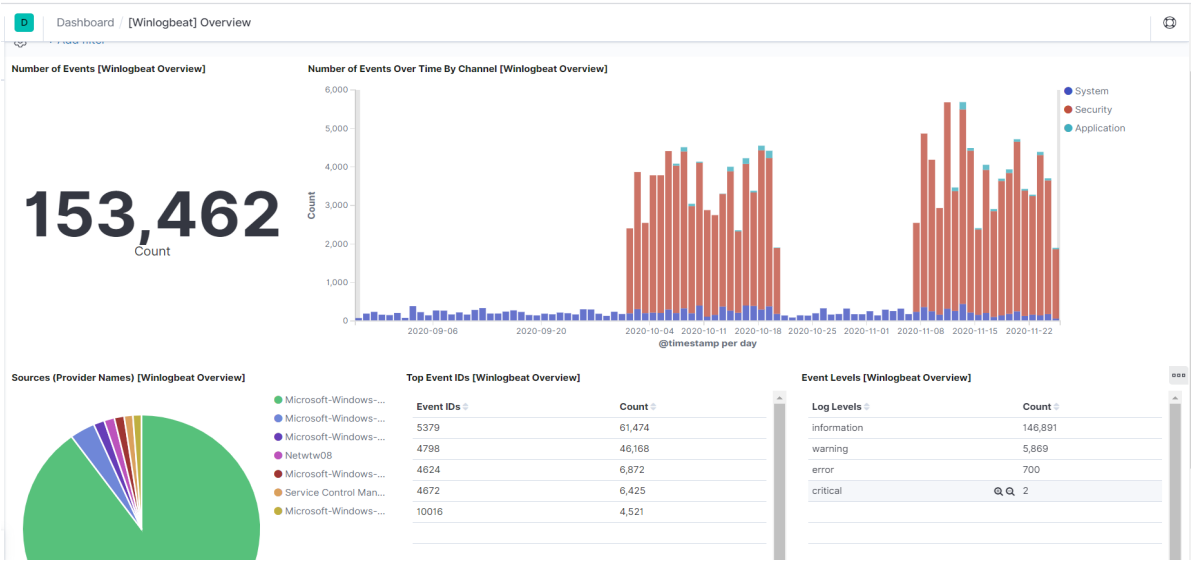

Figure 4.4.1 - Winlogbeat dashboard Overview

- This dashboard visualizes the related to the user management events. Data is visualized using the Metric, Data table, pie-chart, heatmap and markdown.

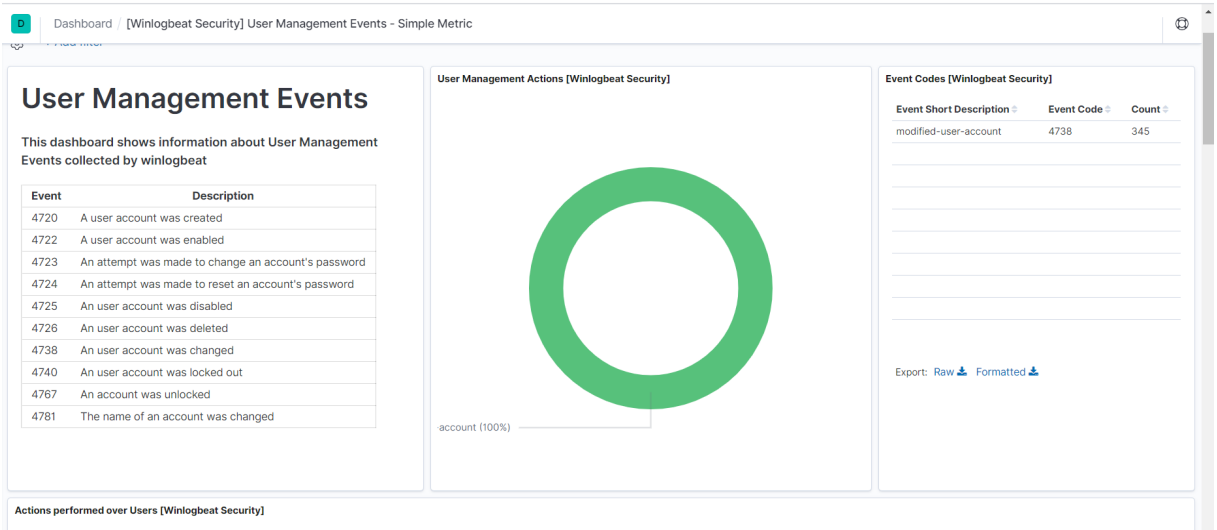

Figure 4.4.2 - User management events

Chapter 5
Conclusion and Future work

5.1 Conclusion

In our project, specifically, we complete several tasks as shown in the following.

1. Log management and Analytics

The server Log files are analyzed and visualized in a clear format and we customize multiple dashboards to present as many as insights that the server log files can provide. This helps web application administrators, security engineers, or devOps engineers to gather helpful information for troubleshooting, knowing the server better and conducting the best way to manage and optimize their servers.

2. Security Analytics

We are able to make use of multiple free open source threat data sources that enable us to conduct different security analytics and gather different insights of our data. Based on this information, a responsive team can step in and conduct any necessary actions to make sure the server is under control.

3. ELK Stack functionality

We also explore multiple components and try different functions of Elk stack, especially Logstash and Elasticsearch, we believe these powerful tools can be utilized in many more use cases in the future.

Using ElasticSearch, Logstash and Kibana along with Packetbeat and Winlogbeat, we were able to gain significant insights about the network traffic as well as for the event in the Windows system on a real-time basis. Building dashboards from these data helped to monitor the network traffic and the Windows system activities through various visualizations such as Pie-chart, Histogram, Wordcloud, Counter, and Geo-location maps.

From the dashboard created using Packetbeat, the information regarding the network that was obtained consisted of Count of HTTP websites visited, Location of the servers to which the requests have been sent, Source and Destination IP addresses for HTTPS and HTTP protocols, Amount of data transferred between source and destination IP addresses, Top 10 DNS questions.

Dashboard created using Winlogbeat monitored the Windows system on the aspects such as Total number of windows events, Top 5 event IDs, Event levels, Number of Events Over Time By Channel.

To conclude, the ELK stack along with Beats acts as a perfect combination to visualize and understand how the network traffic and events in Windows systems work.

5.2 Future Work

Due to the limited of time and resources, there are still many limitations in our capstone projects, and we can have further research on the following directions:

1. More Types of Attacks

DDOS is a widely used attack used by attackers, so we pay most of our attention to it. In fact, there are many other types of attacks such as Smurf or Trojans in the realistic scenario. In future work, we can try to make our system compatible with more types of attacks. In addition to the types of attacks, giving out more analysis reports and countermeasures 41 against different threats is meaningful.

2. Apply Machine Learning Models

Traditional log analysis systems are detecting known or manually identifiable attacks. However, once an attacker slightly modifies some known malware, it poses a major challenge because the above detection methods will fail. If the attacker makes a large modification and changes the feature identifier of the malware, the method of detecting by using the identifier does not detect any abnormal content. At this point, machine learning can take advantage of it, by learning the existing data and building models to predict unknown data, so as to conduct security analysis and predict the newly generated logs.

3. Larger scale of data

Compared with the scale of data used in industry, the data scale we used is little. So Another direction is to make our project support a much larger data scale. Perhaps it will be really difficult to process such a huge amount of data on off-line systems, we can turn to AWS or Azure for help. Taking advantage of the clouds, our project could handle more data and more practical scenarios

The dashboards that we have built, can only monitor the security aspects. But only monitoring is not sufficient, in most of the use-cases where Elastic Stack is used for monitoring, security and alerting as well. To implement an alerting system in our project we can use X-pack services of the Elastic Stack and to implement more security we can implement Machine Learning services of the Elastic Stack to get the alerts on Email, Webhook, Jira and Slack whenever the suspicious activity/patterns or the events are observed.

References

1. http://www.ijcse.com/docs/INDJCSE19-10-05-008.pdf
2. https://www.elastic.co/guide/en/kibana/master/vega-graph.html
3. https://www.elastic.co/guide/en/elasticsearch/guide/current/hardware.html
4. https://github.com/Blueliv/elk-config-examples/blob/master/documentation.pdf
5. https://www.spamhaus.com/resource-center/dns-blocklist-basics/
6. https://www.elastic.co/guide/en/beats/libbeat/current/beats-reference.html
7. https://www.elastic.co/guide/en/elasticsearch/reference/master/pipeline.html
8. https://www.elastic.co/guide/en/beats/packetbeat/current/configuring-howto-packetbeat.html
9. https://www.elastic.co/guide/en/beats/winlogbeat/current/configuring-howto-winlogbeat.html