



Princess Sumaya جامعة
University الأميرة سميرة
for Technology للتكنولوجيا

Design and Implementation of Network analysis and abnormal behavioral detection tool

By

Hamza Al-Quabeh & Mahmoud Al-Shattali

Supervised by

Eng. Farah Abu-Dabaseh

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE

in

Network and Information Security Engineering

at

PRINCESS SUMAYA UNIVERSITY FOR TECHNOLOGY

Amman, Jordan

2023-2024

This is to certify that I have examined
this copy of an engineering documentation by

Hamza Al-Quabeh & Mahmoud Al-Shattali

And have found that it is complete and satisfactory in all respect,
And that any and all revisions required by the final Examining Committee have been made

Eng. Farah Abu-Dabaseh

Acknowledgments

We would like to express our sincere gratitude to our committed supervisor, Eng. Farah Abudabaseh, for her tireless work and steadfast dedication, which greatly aided in the growth and accomplishment of this project. And we would also love to deeply thank Eng. Farah, for her insightful criticism and invaluable direction that helped shape our project.

We also thank Princess Sumaya University for Technology for creating an atmosphere that promotes creativity and academic distinction, laying the groundwork for our initiative's expansion and improvement. We extend our deepest gratitude to Princess Sumaya University for Technology's instructors, students, assistants, and colleagues, and it has been a pleasure to make them a part of our journey.

Lastly, we would like to thank our families and friends for their support. We would never be where we are without their support and encouragement.

Hamza Al-Quabeh

Mahmoud Al-Shattali

Abstract

As far as the cybersecurity threats are concerned, the identification and classification of events in the network is critical and should happen immediately. This graduation project intended at illustrating a new network analysis tool that the authors consider to be suitable for this purpose. The main objective of the project is to develop efficient software that in real-time scans for malicious activities in the network. Besides, vulnerability scanning with depth would also be incorporated into the tool to address new and previous threats correspondingly. It also work in the manner of pulling and feeding external threat intelligence to users, this means that it inform users of the rising threats. In addition, the tool monitor the flow of the network and give constant reports on it to be used in the study of the flow. The concept of such an inclusive approach is to ensure the network has the maximum level of protection by establishing a diverse layered security that will safeguard the network.

It performs another important task of classification of the abnormal behaviors on the network employing some standard algorithms. It also has the ability to deliver these notifications at the exact right time. Besides, it is broader than detection as it provides the detail alert and notification to administrators about potential threats of security breaches at the earliest. It also entails characteristics to assist in the visual representation of the gathered logs for enhanced understanding and analysis.

Table of Contents

Abstract	iv
List of Figures and Tables	7
1 Introduction	8
1.1 Project Objectives	9
1.2 Design Requirements	9
1.3 Design Constraints	9
1.4 Engineering Standards	10
1.5 Preliminary Design	10
1.6 Division of Labor	12
1.7 Document organization	12
2 Background and Literature Review	13
2.1 Background	13
2.1.1 Intrusion detection system (IDS)	13
2.1.2 Network analysis tool	14
2.1.3 Nmap: The Network Mapper	14
2.1.4 Snort	14
2.1.5 Packetbeat	15
2.1.6 ElasticSearch	16
2.1.7 Kibana	16
2.2 Literature Review	17
2.2.1 Network Security Analysis with SnortIDS Using ACID (Analysis Console for Intrusion Databases)	17
2.2.2 Choose Your Best Network Vulnerability Scanning Tool	17
2.2.3 A Framework for Social Media Data Analytics using Elasticsearch and Kibana	18
2.2.4 Towards Efficient Labeling of Network Incident Datasets Using Tcpreplay and Snort	18
2.2.5 Network Monitoring and Enumerating Vulnerabilities in Large Heterogeneous Networks	19
3 Design	20
3.1 Design Requirements	20
3.2 Analysis of Design Requirements	20
3.3 Analysis of Design Constraints	21
3.4 Different Designs Approaches/choices	21
3.5 Developed Design	22
3.6 Create Deployment	23
3.7 Create Fleet Server	23
3.8 Configuring Filebeat	24
3.9 Configuring Packetbeat	25
3.10 Configuring Snort	26
3.11 Real-time Python Script	27
3.12 Python Scan Script	28
3.13 Python Configuration Script	30
3.14 Elastic and Visualization	32

4	Results.....	35
4.1	Accessing The Dashboard	35
4.2	Visualized Data	36
4.3	Validation of design requirements within the realistic constraints.....	39
5	Conclusion and Future Work.....	40
6	References.....	41

List of Figures and Tables

Figure 1.1: Preliminary Design	11
Figure 2.1: Project Diagram	13
Figure 2.2: Snort Processes[1]	15
Figure 2.3: Elasticsearch Procedure	16
Figure 3.1: Final Design	22
Figure 3.2: Elastic's Deployment	23
Figure 3.3: Agent Installation Curl	23
Figure 3.4: Agent System Status	24
Figure 3.5: Filebeat's Input Configuration	24
Figure 3.6: Packbeat's Cloud Configuration	25
Figure 3.7: Packetbeat's Protocol Configuration	25
Figure 3.8: Snort's Network Variable Configuration	26
Figure 3.9: Snort's Output Configuration	26
Figure 3.10: Snort Running	27
Figure 3.12: Real-time Alert Script	27
Figure 3.11: Filebeat Running	27
Figure 3.13: Example Alert	28
Figure 3.14: Port Scan Command	28
Figure 3.15: Host Scan Command	28
Figure 3.16: Script Scan Output	29
Figure 3.17: Intelligence Gathering Script	30
Figure 3.18: Intelligence Gathering Output	30
Figure 3.19: Snort and Scan Configuration	31
Figure 3.20: Main of Configuration Script	31
Figure 3.21: Convert to JSON	32
Figure 3.22: Main of Run Script	32
Figure 3.23: Dashboard Creation	33
Figure 3.24: Lens Entry	33
Figure 3.25: Choosing Indices	34
Figure 4.1: Analytics' Dashboards	35
Figure 4.2: Event Details Table	36
Figure 4.3: Host Details Table	36
Figure 4.4: Event Message Pie Chart	37
Figure 4.5: Event Priority Line chart	37
Figure 4.6: Average Bytes-in	37
Figure 4.7: Average Bytes-out	37
Figure 4.8: Average Packet Length	38
Figure 4.9: DNS Query Details Table	38
Figure 4.10: DNS Answers Details Table	38

1 Introduction

These days, the internet plays a crucial role in our daily lives. It's rare to have a day when you don't need to use the internet. This includes businesses and individuals alike. It has become a crucial part of our infrastructure and economy, which makes it a prime target for attackers trying to cause harm to organizations. Such attacks, if successful, can cause billions of dollars in damages and cost people their jobs. For this reason, we decided to make our graduation project about protecting our confidentiality, integrity, and availability to the internet.

The threat that resides in the project is the security threats that exist today in the form of cyber occurrences, which have become frequent and more severe to inflict major financial losses, compromise individual's data, and destabilize vital systems. In adversarial anomaly based methods, the patterns are clearly prescribed in a way that only helps identify known forms of attack; as a result, they are least effective when it comes to identifying new complex attacks that take advantage of vulnerabilities that are yet to be identified by researchers, commonly referred to as zero day attacks[13]. This goes to show that there is need to develop better methods that are intelligent and elastic enough to detect any indications of irregularity in the networking traffic flow.

Another issue is that of consolidating all this collected information under one roof that disaggregates this information. This does so with the help of significant current software tools like the Nmap and Snort- both of which are already popular for extracting data from the traffic data of a network[1][9]. More codes have been included in this program with the help of the Python script language that can be used to automate most of the processes involved. It was also utilized to create a way for snort alerts to be sent in real time as snort by itself does not have this feature.

In other words, to determine some security threats, it's necessary to join and analyze logs. An open-source solutions worldwide popular is to be introduced and thus Kibana is to appear. Specifically, in the further analysis, we will take advantages of other functions in Kibana which has more superiorities for visualizing log data[7][8], then turn the data that we received into useful information. In achieving its objectives, it is significant and useful in helping security personnel to view trends and perform a search to discover additional information best viewed in total, hence it gives them a complete picture of their network health.

We do not only limit our work to only data collection and analysis just like most of the research activities. For this reason, we have adopted scripting using Python to control more of the output that comes out of the Nmap scan. This addition could entail elements such as providing Nmap with an ability to perform repetitive work such as periodic scanning and allows us to change the output formatting in whatever way fits best.

1.1 Project Objectives

The main aim of this project is to develop a strong network security system using Snort IDS so that it can monitor the network traffic and detect any threats in real time. The system use of these and other more sophisticated detection techniques will allow the enterprise to monitor live network traffic in real time; if for example, a malware attack or protocol violation is launched. Additionally, the system will be equipped with powerful vulnerability scanning tools that track and remove network security vulnerabilities. The system will also monitor and analyze network traffic flow to help us for our further development. Everything will be accessible from a single point for the user to use. The overall goal of this project is to fortify network security and shield it from outside threats so that it remains dependable and safe.

1.2 Design Requirements

- The tool shall conduct performance analysis, including latency measurement, packet loss assessment, and bandwidth usage tracking.
- The tool shall integrate with external threat intelligence feeds and services to stay updated on the latest cyber threats and vulnerabilities.
- The tool shall collect and store network data, including connections, and traffic patterns.
- The tool shall have the capability to perform network discovery to identify active devices and services.
- The tool shall conduct vulnerability scans to identify potential security weaknesses in network devices and services.
- The tool shall generate alerts and notifications for critical events, such as security breaches or performance anomalies.
- The tool shall detect abnormal behavior or patterns in network traffic, potentially indicating security incidents or performance issues.

1.3 Design Constraints

- Economic Constraint:
 - It will cost us the price of paid Cloud services.
- Manufacturability and Sustainability Constraint:
 - We need to ensure that the network that we will analyze is working well.
 - Consider releasing the software as open-source, allowing a community of contributors to maintain and improve it over time.
 - Rigorously test the software to identify and resolve any bugs or security vulnerabilities. Continuous testing and quality assurance are crucial for a reliable network analysis tool.

1.4 Engineering Standards

- IEEE 802: This series of standards covers local area network (LAN) and metropolitan area network (MAN) technologies, including Ethernet (802.3), Wi-Fi (802.11), and more.
- IEEE 802.11: A family of standards for wireless local area networking (Wi-Fi).

1.5 Preliminary Design

The purpose of this project is to create a tool that protects the network from outside threats and provide the necessary information to expand and improve said network. Our approach aims to provide everything the tool offers in one place where the user can easily access and understand the data.

The flowchart below describes how the tool works and how the data reaches the user. First when we start the operations, an intrusion detection system will start monitoring the network and producing logs, these logs will be sent to a service called Kibana which will visualize out logs and data. A network flow tool keeps track of the network flow and consistently sends data to Kibana to visualize it. Vulnerability scan occurs periodically to keep track of any open vulnerabilities on the network and its output is then also will be visualized in Kibana.

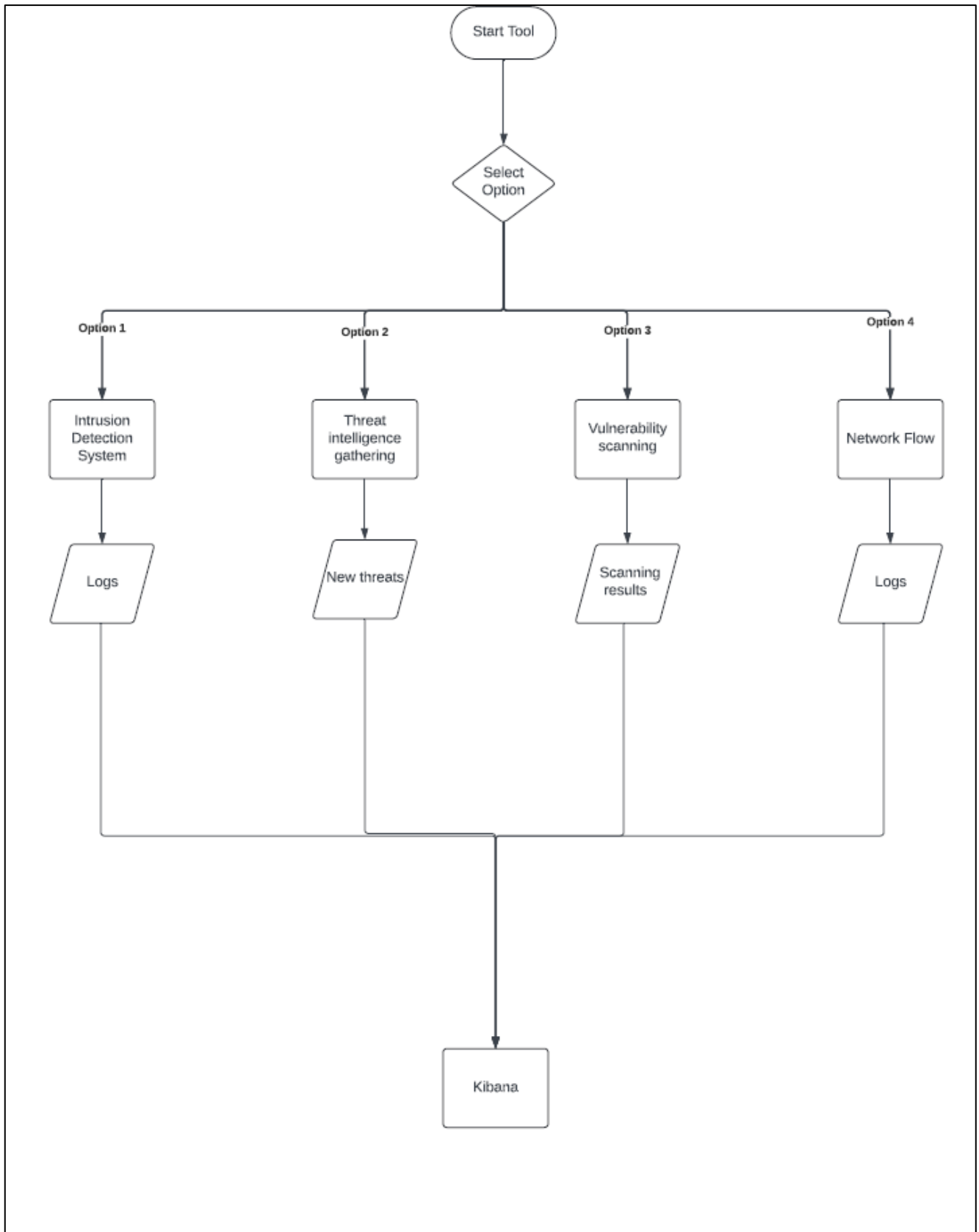


Figure 1.1: Preliminary Design

1.6 Division of Labor

Table 1: Division of Labor

	MAHMOUD	HAMZA
DOCUMENTATION	✓	
CONFIGURING AGENTS	✓	✓
ELASTICSEARCH FLEET AND KIBANA	✓	
PYTHON SCRIPTS FOR SCANNING AND ALERTING		✓
CONFIGURING SERVICES		✓

1.7 Document organization

The following chapter provides an overview of the major technologies that will be utilized in the implementation of this project, as well as a similar approach to that stated in the preliminary design. Where Chapter 3 explains the project design, requirements, methods, and technologies. While Chapter 4 shows the results and compare them with previous projects. Finally, Chapter 5 includes the conclusions, improvements, and future work consideration.

2 Background and Literature Review

2.1 Background

There are two primary parts for our system; First is the Collection of Network Data, this component may use network sniffing tools or API integrations to extract the network traffic information from sources like switches, firewalls, routers, and other relevant equipment as it shows in **Figure 2.1**. The second component is Connecting and Visualizing: For the logs and data visualization Elasticsearch and Kibana actually with the hosts and the agent depends on the infrastructure and some features about the project that we have.

2.1.1 Intrusion detection system (IDS)

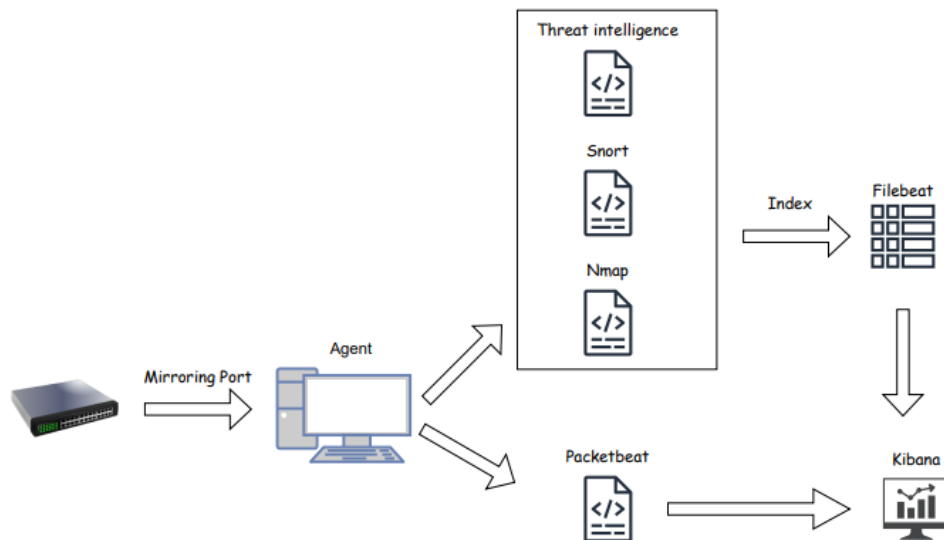


Figure 2.1: Project Diagram

IDS is a security system created for detecting improper actions onto a network or computer system. It has a function of sentinel, the one constantly scanning the interactive space for forms of breach, of denial of service attacks, of worms, viruses, or any form of upset.[13]

An IDS permanently screens all the processes happening in the network or concerning the systems, which is about the examination of the incoming and the outgoing data packages or the system calls.

Furthermore it is important to mention that it is evident that most IDS technologies use anomaly-based detection with the detection based on signatures. This process tries to find a traffic or system and identify whether the traffic/system is normal or not. this could be in a social way imply that there is some vice activities that are going on in the area of interest that warrants investigation.

Afterwards, based to the messages it discerns, the IDS often sends an alarm to the security team with regards to a probable menace. IDS could also acquire more detailed

information about the identified activity for analysis, as well as investigation.

2.1.2 Network analysis tool

Network analysis tool is a tool that can be used for the identification and diagnosis of faults within the network. the rate of traffic flow in the network can also be established and the type of traffic that is being fed to the network is also distinguished[5].

It can scan the network for any violators and brief the security personnel in the event of any upcoming threat.

It can perform an analysis of the parameters such as the latency or delay, the packets drop and the bandwidth on the network. It also aids in identifying and even magnifying areas that perhaps could be considered less effective or the networks are not as swift as other networks.

Most tools enable the graphical depiction of the network topology, the devices studied in the network, as well as the connection and the traffic flow patterns[9]. It is beneficial in displaying the topology of the network and more to that, it can be useful in diagnosing potential problems in the process.

2.1.3 Nmap: The Network Mapper

Nmap is a graphical mapped network discovery tool developed for security assessment and it is web based thus no download is needed. It is the useful instrument used by the network administrators and it security officers. Nmap has evolved to become a rather command-line based tool, and while there aren't too few graphical interfaces available for it, there definitely aren't too many. In targets, there are generic targets which can be IP addresses and address ranges or specific options depending on the selected scan type. There are different types of scanning methods in Nmap, mainly designed for these purposes [10]. Below are some of the most common scans:

- VULN scan: Nmap scans for network vulnerabilities and misconfigurations.
- UDP scan: It was also used in the confirmation of the open/available UDP ports.
- OS detection scan: To profile the computer and ascertain the operating system used.

We chose Nmap for the process of scanning and discovery of the network, and the fact that it is open source as well as having cross-performance the program has proved to be very efficient. There are other tools that are somewhat comparable to Nmap, but the advantages of flexibility, usability, and reliability make Nmap our tool of choice.

2.1.4 Snort

Snort is another robust and highly developed open source NIDS which has massive importance to the project and its security amplification. Remarkably, Snort is reliable and flexible. It is, therefore, a wise guardian in a way that it investigates the current connections and network traffic to fending off security breaches.

Snort based on the detection or the signature-driven model works by inspecting all the packets that is going through the network , and blinks in the instance that it recognizes any form of deviation or an attack signature. The given approach also enables Snort to quickly identify rather various forms of suspicious actions starting with the port scans and ending with intrusion attempts thus preserving the availability and integrity of the network.

As with most of its features, one of Snort's strongest aspects which are an obvious outcome of its design is the ability to incorporate into Snort complex rules optimized for a certain network's needs. These flexible arrangements allow the security profession to address the required characterization of the detection strategy when needed while maintaining congruency with the constantly mutating threat landscape.

There are multiple reasons we picked Snort for our choice of IDS. It is open source, hence customizable and is a stable environment for hosting websites. Rule sets can also be customized and dealing with notifications and alerts as depicted in the **Error! Reference source not found.** real time detection, which makes it suitable for organizations of different sizes. Snort also possess large user base where resources can be published and information can be shared between its users. Overall, Snort implementation makes organizations to mitigate and prevent cyber threats, and adequately safeguard it TIC within efficient manner. [1]

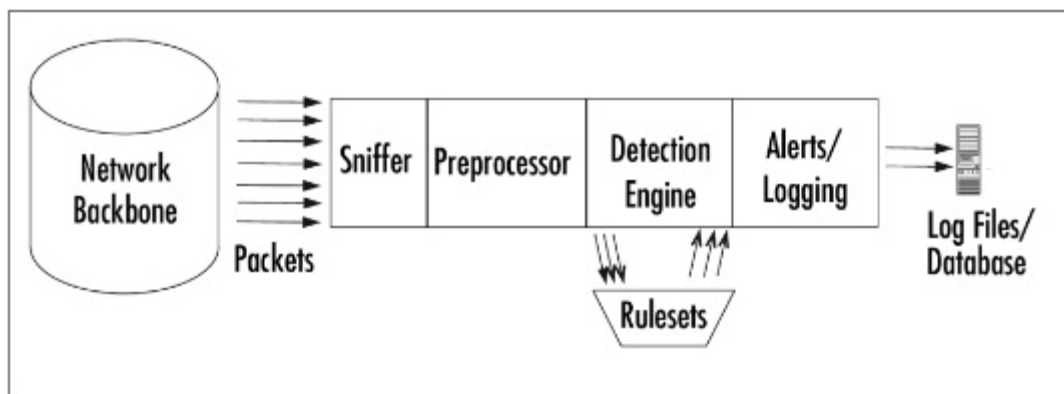


Figure 2.2: Snort Processes[1]

2.1.5 Packetbeat

Is one of the components of Elastic as a real-time network packet analyzer. It captures the network traffic, and also analyzes several applications protocols like DNS, MySQL, and HTTP at the application layer. Then the data is passed to Elasticsearch so that visualization tool like Kibana can be employed in processing it. As an example, you can use Packetbeat for logging your application's health and growth using data like the number of responses, rate of error occurrences, and the amount of network delay. Pulled together, this provides base-level and detailed understanding of network traffic, behavior of applications, and helps in diagnosis of issues, fine-tuning of applications, and improving general performance and utilization of your networks.

Packetbeat comes with very basic and useful visualization of data, and it integrates well with the Elastic Stack. It covers a wide variety of protocols. These include DNS, MySQL, and HTTP among others, thus it is possible to have a detailed look at the network service at the application level. Using the Packetbeat tool, one can easily address the security threats and the performance issues as the tool provides real time analysis. All these reasons together made us use Packetbeat as our choice for a network analysis tool.

2.1.6 Elasticsearch

Is an open source, highly scalable and distributive real-time web search engine for applications. Using inverted indices and finite-state transducers, it uses full text querying [7]. enables a general-purpose approach to building a distributed computing environment that can execute efficient and immediate data management and analysis. These responsibilities are well-handled by Elasticsearch, which is designed using distributed structure and has embedded search engines that use Lucene and are designed for document search and indexing. Some of the benefits that can be derived from it include versatile querying such that it can do highly helpful operations like searching, creating indexes, and calculating. Kibana as a visualization tool for Elasticsearch is recommended since it has interfaceable APIs that allow the easy integration of the platform and applications. It also provides the service of a fleet server. A fleet server allows you to create and manage elastic-agents, set specific policies for said agents, and a central place to configure your cloud deployment.

2.1.7 Kibana

Kibana is one of the amazing tools that is more capable to visualize data that is based on Elasticsearch and Elastic Stack. It has an integrated web front-end for report creation, dashboard and visualization that must be customized depending with the requirements of an organization. This implies that, through responding to the querying and aggregation functionalities of Elasticsearch, Kibana enhances the analytics of smart data in real-time. The fact that it does not rely on a fixed structure but can be extended through plugins, and has a tutorial-like graphical interface, means that people at different levels of technical proficiency can utilize all of its potential and make enterprises analyze their data more efficiently. In Figure 2.3 it shows the procedure of connecting all together until we reach Kibana and visualize logs. [8]

We chose Kibana instead of the other solutions due to features like Seamless integration with Elasticsearch, high quality of visualization, powerful query and aggregation functions, a clear and user friendly interface, option to customize and flexibility of creating centralized dashboards. These peculiarities make Kibana one of the most desirable solutions for businesses that struggle to work with and analyze their data. Moreover, Kibana does not like relational SQL that, in turn, enables relations to be placed in rows and the view of fields altered.

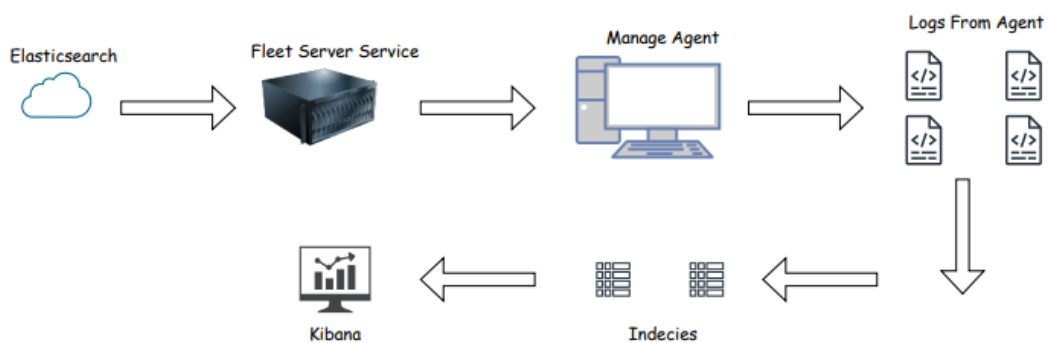


Figure 2.3: Elasticsearch Procedure

2.2 Literature Review

2.2.1 **Network Security Analysis with SnortIDS Using ACID (Analysis Console for Intrusion Databases)**

The paper analyzes the field of network security and investigates how it plays a vital today and will continue despite constant evolution of threats.

This highlights the need to incorporate strong security architectures within the tested network, and integrates tools such as Snort IDS to detect intrusions and architectures such as ACID, to analyze databases of intrusions to increase the levels of protection against malicious undertakings.

The paper explains the need to apply IP Tables on Ubuntu Server as a firewall to effectively combat any attacker before they can breach the system security and attain unauthorized access to the organizations' resources.

The paper also looks at the need for the development of more preventive and monitoring network systems, which have recommendations that include a method for effectively creating and implementing a network system; this method includes real time packet checking and detection sensor as well as the systems alert mechanism to block threats.

Also, the conclusion was made about VLAN for the purpose of network segmentation and the improvement of network quality and security and the effectiveness Of an IDS system in network security and intrusion detection.[9]

2.2.2 **Choose Your Best Network Vulnerability Scanning Tool**

Hackers' directions are analogous to vulnerability scans in the fields of cybersecurity implementing the network defense. Identifying the vulnerabilities on the computer systems is an important step towards protecting the real life intrusions and a good secure System.

It is suggested that the prescription to conduct the vulnerability scanning is raised to increase the awareness and knowledge of ethical hacking and system security in the cybersecurity program. The traditional paradigm contains an outline for a lab base for the work presented in the article and an appraisal of the state in open source vulnerability detection technologies.

Approximately, the discussion is focused on the elements of vulnerability scanning in networks and based on this, it is stated that it is crucial to define vulnerabilities in computers. The role of vulnerability detection in case of education is stressed in The best open source tools are presented with great details here.

This is an opportunity to try and understand their system vulnerability in a bid to fight back the hackers' attacks as has been demonstrated in the study. It is considered as one of the crucial curricula while teaching network security and ethical hacking since it enables the students to defend computer and participate in lawful hacking activities in reality.

To provide an understanding of vulnerability scanning the article begins with an explanation of context on practice before introducing the reader to specifications of open source tools in practice, building up to practical tutorial along with OpenVAS and retrospective assessment of the tool for effectiveness in learning cybersecurity.

Discussing the outcomes of the practical labs, the authors recall the opportunities for the additional studies that might contribute to the further improvement of the future thinking about the vulnerability scanning at the sphere of cybersecurity education. [4]

2.2.3 A Framework for Social Media Data Analytics using Elasticsearch and Kibana

The research paper underscores the importance of the real-time, online data processing and in social media analysis; more so in the situation where data needs to be continuously fetched and analyzed for current business intelligence .

Elasticsearch is presented as an effective tool to address the issues arising with real-time analysis, based on its innately adaptable and micro-segmented search engine, distributed data storage index, and standard, pre-made indexing for large scale text mining .

The framework described in the paper again deals with the utilization of Elasticsearch and Kibana for monitoring Twitter data and demonstrates the development of the dynamic dashboard reflecting different metrics like source of the tweets, languages, the flow of the data over time, the most used words in the tweets, etc. , and finding out the most influential and active users among those mentioned in the dataset.

The functionality of the system is supported by its ability to accomplish tasks swiftly; data retrieving, descriptive analysis, and generation of graphs, including for the given company with a sufficient scale, Gross 250M tweets are within 15 seconds, which makes it suitable for online analyzes of large textual datasets.

The study also stresses the need to properly optimize Elasticsearch and Kibana for big data analysis as well as other real-time analysis functions to achieve near-instantaneous visualization of large and complex datasets in a format that can easily be utilized to inform decision making. [14]

2.2.4 Towards Efficient Labeling of Network Incident Datasets Using Tcpreplay and Snort

In this research, emphasis will be given to the proper classification of network traffic data as NIDS research is improved. It is customary to mark this data according to the type of attack it is believe to exemplify the network packets. This manual approach is time consuming, boring and prone to human error, than the programmed mode from the foregoing analysis.

It suggests utilizing Tcpreplay and Snort to label rules automatically. Tcpreplay is a tool used in network analysis as its primary function involves retransmitting the packets captured from the network in several network environments. For example, Snort system is a NIDS that can detect the malicious traffic based on the attack signatures in its pre-defined database.

The system utilizes network flowing, which is input of packets of traffic data that can include both normal traffic and packets of attack. Tcpreplay then replays this traffic in a packet cap environment. At the same time, Snort has to scan the packet against its signature database in order to detect the presence of any matches to known intrusions. If a packet matches any the signatures that are set in Snort, then it is seen to be carrying out a malicious attack.

The system will also be able to locate back the particular packet associated with the attack that Snort has been able to detect in the originally captured data. The packet is then labeled to show the specific type of an attack intended out of the network.

First, it enhances the efficiency of the rater in that they do not have to spend a lot of time labeling since most of it is done automatically. Consequently, it decodes the labeling of network traffic data with a considerable reduction in time and efforts relative to the traditional manual approach. Furthermore, because the system relies on Snort's specific signature check, it can reasonably offer higher labeling accuracy. Further, the system is quite elastic and feasible to deal with large sizes of data, making it suitable for functional NIDS investigation projects.

The researchers intend to make a further improvement on the system and also probe into possibility on And they aim at exploring ways and means such as improving the labeling accuracy and broadening the applicability of the system in analyzing various data of network traffic originated from different sources. Regulated billing would thus considerably extend the research and development areas of NIDS.

In conclusion, this study has proposed a promising approach to the classification of network traffic data and has therefore brought into the discussion opportunities to design a better, accurate and efficient NIDS. [6]

2.2.5 Network Monitoring and Enumerating Vulnerabilities in Large Heterogeneous Networks

The present work focuses on the main topic, that is, the consideration of vulnerability enumeration in computer networks tied to the use of the commonly accessible network scanning and sniffing tools and the challenges that relate to large-scale and heterogenic computer networks.

During the active and passive network scan, all network devices, including switch and routers, shall be detected the type, and version of computer systems and software resources shall also be detected in the network. This data is then matched with the currently available, recent vulnerabilities which are obtained from the NVD by using common database feature such as CPE.

In the current research, the main focus is to correlate the active method with the passive method to see which is most efficient in identifying more devices in the network and accurately tagging them accordingly. The analysis of traffic data derived from passive networking is acknowledged to produce more data than active mapping of the network even though the process has lower accuracy. Passive network traffic analyzing seems as the technique that creates greater traffic amounts but sometimes with lesser precision compared to the active network scanning.

The paper also illustrates the usefulness of these methodologies by an example area that explains how to analyze hosts that have different CPE identifiers and relate them with susceptibilities within a campus network. To achieve this, there is usually a discovery scan carried out, and this aids in identifying other hosts in the network that might also be infected; this creates room for the next phase, which is vulnerability analysis and remedy.

Furthermore, the research work indicates that it is imperative to ensure that an active and passive monitoring method is adopted in order to arrive at a comprehensive list of vulnerabilities relevant for big networks Finally, the study reveals that it is important to note that while using tool like Shodan can be useful in gradually fortifying a network, it is not enough. It served to justify the positive reason for the need to embrace the Systematic approach in the management of network risks. [5]

3 Design

3.1 Design Requirements

- The tool shall conduct performance analysis, including latency measurement, packet loss assessment, and bandwidth usage tracking.
- The tool shall integrate with external threat intelligence feeds and services to stay updated on the latest cyber threats and vulnerabilities.
- The tool shall collect and store network data, including connections, and traffic patterns.
- The tool shall have the capability to perform network discovery to identify active devices and services.
- The tool shall conduct vulnerability scans to identify potential security weaknesses in network devices and services.
- The tool shall generate alerts and notifications for critical events, such as security breaches or performance anomalies.
- The tool shall detect abnormal behavior or patterns in network traffic, potentially indicating security incidents or performance issues.

3.2 Analysis of Design Requirements

Everything inside our will tool will be linked to Elastic search in order to be able to monitor everything our tool offers from one singular place. A fleet server would be required to receive and store all the information on. As a network administrator you will be able to view information about the current network flow, real-time security threats , RAM and CPU usage of our agents. By that we achieved the third and fourth requirement.

Our elastic-agents will be deployed on whatever machine we wish to monitor and by that we achieved the first requirement. The agents are configured to establish a connection to our fleet server and start sending data. Everything used for monitoring must be installed on these agents such as Snort and Packetbeat. As the elastic-agent is prebuilt to send system metrics to the fleet server we use Filebeat in order to read our snort logs and send them through the elastic-agent [11]. Which are then filtered and visualized using Kibana.

The vulnerability scan is designed to occur periodically to discover all live hosts and find any new vulnerabilities on the system, it utilizes Nmap's NSE to keep up with new threats and attacks. By that we achieved second and fifth requirement.

We utilize Elastic's Packetbeat packet analyzer to send data about the current network flow on our elastic-agent, It can be easily integrated with the fleet server as it is a service provided by Elastic themselves. We then use Kibana to visualize this information.

Security breaches are detected using the Snort IDS, Snort can be configured to suit any and every network its ran on. It provides much flexibility for the types of attacks depending on the nature of the network. Any urgent alerts are sent using a python script built by us that searches the log's files for high a priority report and then sends said report to the network administrator via E-Mail[9]. By that we achieved the sixth and seventh requirement.

3.3 Analysis of Design Constraints

It will use fee based cloud service in the project hence implying some cost. This also defines the place to host the project and the extent of computational resources to be employed in the process. Here, it should be noted the difference between the cost of applying cloud solutions as well as providing on-site facilities. It can also mean for example that expanding the system and adding new features to the application in the future will be equivalent to the rudimentary to only the basic functions, which will consume most of the resources. Its mainly designed for use in LAN for instance in home and or office and it may not be very well equipped to handle complex networks. This constraint mean that the tool has to be efficient for the smaller networks thus lies it's efficiency. This will affect the discovery and monitoring of the network hence reduced extent of network coverage.

Thus, when analyzing the network, which is to remain efficient, our code can only be done by developing sophisticated packages that are used to scan the network and ascertain its state before it is diagnosed. Open sourcing the software also has several implications relative to the future maintenance of the code, its documentation, and social consequences. This also implies that the code standards must be good, and the code must be written in such a way that all the contributors must be able to comprehend it well apart from it being extensible. Automated integration and deployment should also be made and tested time to time to make sure the testing and quality assurance are efficiently done.

3.4 Different Designs Approaches/choices

This section reveals several design techniques that can be applied for this project in order to be accomplished.

- First approach: The initial strategy considered in designing our tool was creating our own custom AI powered NIDS. This would help the tool identify any new or zero-day attacks performed on the network and overall prove better security performance. The issue with this approach is integrating the NIDS with new networks as the environment will not be compatible with the trained model.
- Second approach: Installing our elastic-agents on cloud machines. This approach calls for the use of cloud services on which to deploy our monitoring agents which include snort, Scanner and Packetbeat. Although this may extend the flexibility of our tool at large networks, we are interested in small to medium scale networks where a single or at most two agents will be able to adequately monitor the network. Also, it will be detrimental to our real-time alert because it will prolong its production.

3.5 Developed Design

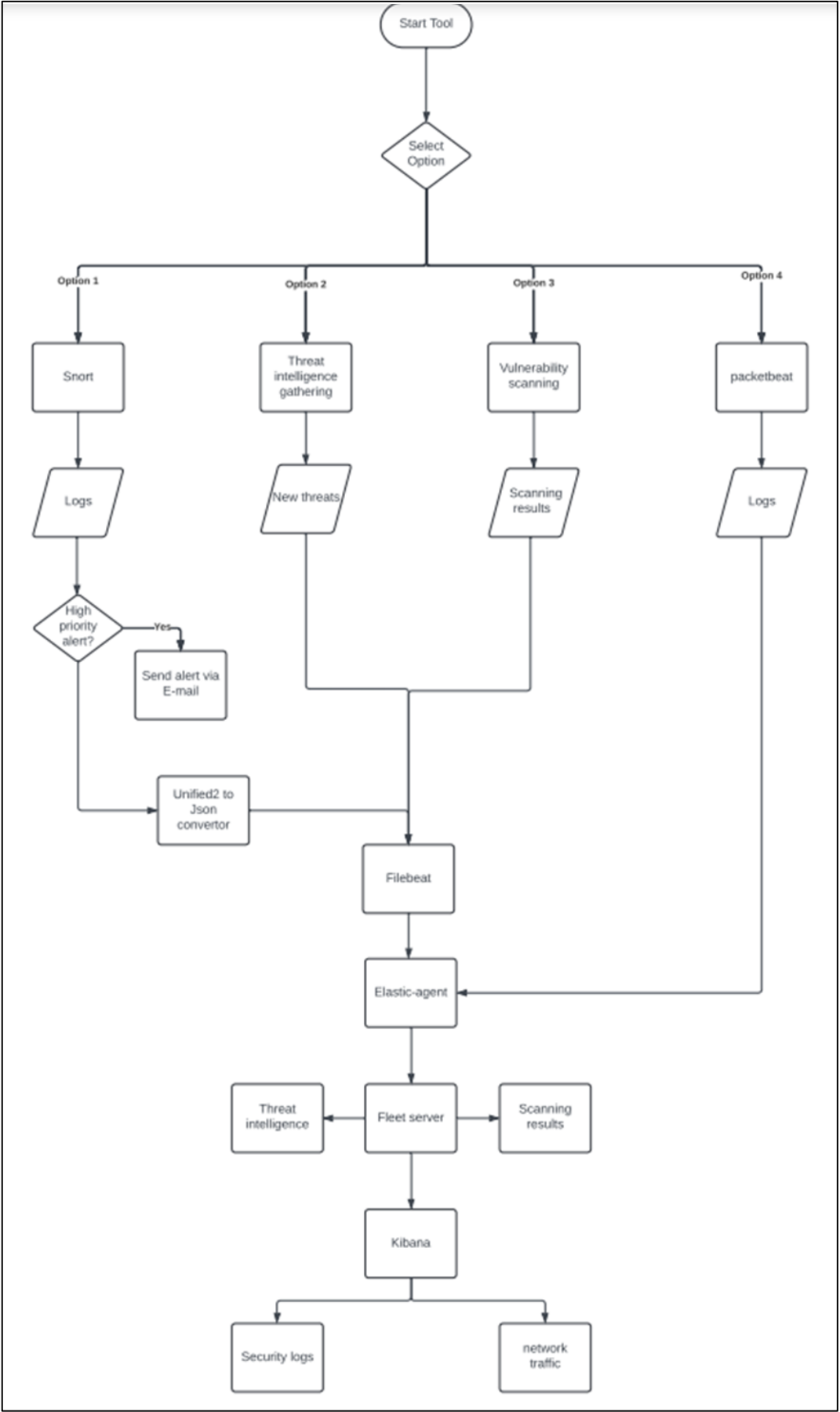


Figure 3.1:Final Design

3.6 Create Deployment

In the developed design we start by first creating an instance of the fleet server on elastic's cloud service, all our nodes and agents will be collected in this server, and it will be the center of our operations. Figure 3.2 shows our deployment on the elastic cloud which includes the fleet server and Kibana.

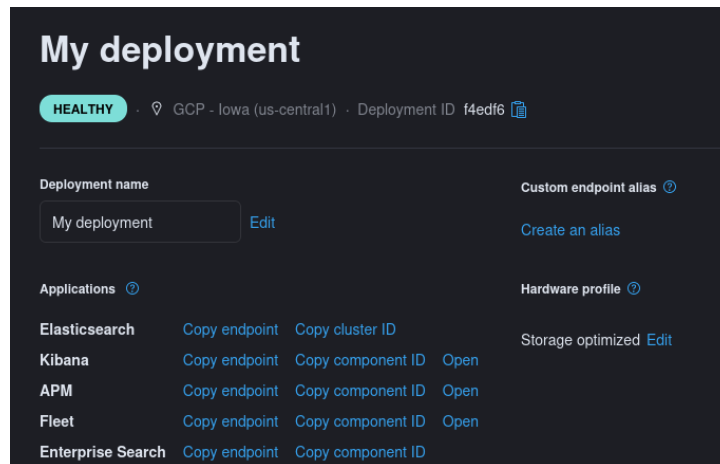


Figure 3.2:Elastic's Deployment

3.7 Create Fleet Server

After installing the fleet server, we can start adding agents to our nodes. The agent installation is as simple as copying your fleet server specific curl and executing it on the node. Figure 3.3 shows the elastic-agent installation curl.

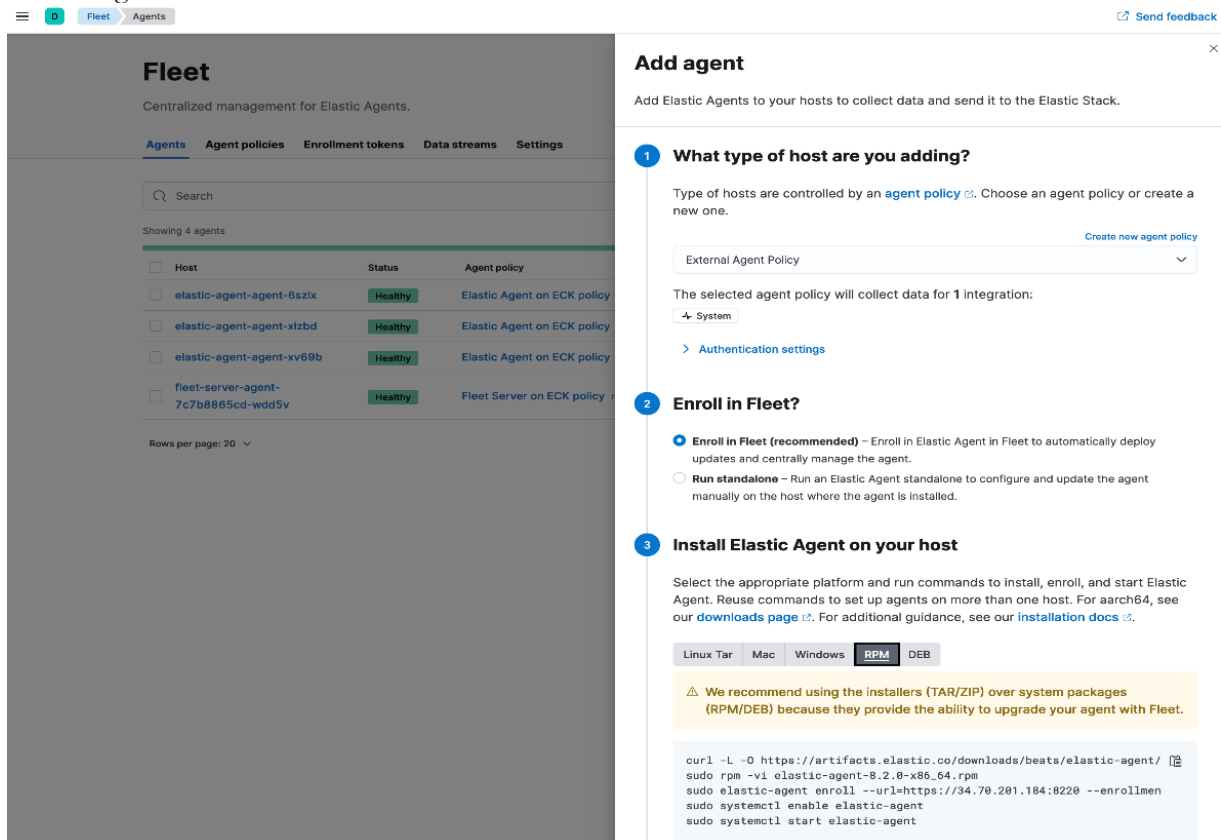


Figure 3.3:Agent Installation Curl

The agent will be installed and automatically enrolled into the fleet server. The connection between the agent and the fleet server is over https, meaning our data's integrity and confidentiality are both protected. Figure 3.4 shows the elastic-agent service running.

```
doffy@doffy-HP-Laptop: ~/filebeat-8.13.4-linux-x86_64
doffy@doffy-HP-Laptop:~/filebeat-8.13.4-linux-x86_64$ sudo systemctl status elastic-agent
● elastic-agent.service - Elastic Agent is a unified agent to observe, monitor and protect your system.
   Loaded: loaded (/etc/systemd/system/elastic-agent.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2024-05-18 01:20:43 +03; 5min ago
     Main PID: 865 (elastic-agent)
        Tasks: 162 (limit: 9292)
       Memory: 952.6M
      CGroup: /system.slice/elastic-agent.service
              └─ 865 /opt/Elastic/Agent/elastic-agent
                 1089 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1095 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/osquerybeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1101 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/filebeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1107 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1133 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1156 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1163 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/filebeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1185 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1192 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/metricbeat -E setup.ilm.enabled=false -E setup.template.enabled=false
                 1214 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/osqueryd --utc=true --pidfile=osquery/osquery.pid --disable_tables=
                 1216 /opt/Elastic/Agent/data/elastic-agent-8.13.4-a2e31a/components/osquery-extension.ext --socket /var/run/252760582/osquery.sock --tl

May 18 01:20:43 doffy-HP-Laptop systemd[1]: Started Elastic Agent is a unified agent to observe, monitor and protect your system..
May 18 01:20:46 doffy-HP-Laptop osqueryd[1214]: osqueryd started [version=5.10.2]
lines 1-22/22 (END)
```

Figure 3.4: Agent System Status

Now that the elastic-agent and fleet server are connected we can start running our services, but before that we must configure them to meet our needs.

3.8 Configuring Filebeat

Inside the Filebeat configuration file we create a new input for snort of type “logs”, this tells Filebeat that the data received will be in Json format. We also notice that we must enable Json for this input to work as intended.

```
home > doffy > filebeat-8.13.4-linux-x86_64 > ! filebeat.yml
23 - type: log
24
31
32 # Paths that should be crawled and fetched. Glob based paths.
33 paths:
34   - /var/log/snort/snort.json
35   # - /home/doffy/Desktop/test.txt
36   #- c:\programdata\elasticsearch\logs\*
37 json.keys_under_root: true
38 json.overwrite_keys: true
39 json.add_error_key: true
40 json.expand_keys: true
41
42 - type: filestream
43
44   id: scan-results
45
46
47   enabled: true
48
49
50   paths:
51     - /home/doffy/Desktop/GP/scan_results.txt
52
53 - type: filestream
54
55   id: int-results
56
57
58   enabled: true
59
60
61   paths:
62     - /home/doffy/Desktop/GP/int_results.txt
63
64
```

Figure 3.5: Filebeat's Input Configuration

Both scan results and the threat intelligence results are sent with the type “filestream” since they are being sent as plaintext. Filebeat’s input configuration is shown in **Error! Reference source not found.**.

3.9 Configuring Packetbeat

Before running the Packetbeat service we must configure it to connect to our fleet server, This is done by setting the `cloud.id` and `cloud.auth` variables. Make sure to protect your `cloud.id` as it is considered sensitive data.

```

194 # ===== Elastic Cloud =====
195
196 # These settings simplify using Packetbeat with the Elastic Cloud (https://cloud.elastic.co/).
197
198 # The cloud.id setting overwrites the 'output.elasticsearch.hosts' and
199 # 'setup.kibana.host' options.
200 # You can find the 'cloud.id' in the Elastic Cloud web UI.
201 cloud.id: "f4edf60c28f647bb9049fd600de5fb55:dXmTY2VudHJhbDEuZ2NwLmNsY3VkbWVzLmV0jQ0MyQyMjk3MmVkbWUyZWMDNDh0"
202
203 # The cloud.auth setting overwrites the 'output.elasticsearch.username' and
204 # 'output.elasticsearch.password' settings. The format is '<user>:<pass>'.
205 cloud.auth: "packetbeat: [REDACTED]"
206

```

Figure 3.6: Packbeat's Cloud Configuration

Also, inside the Packetbeat's configuration file is a list of protocols to capture and their specified ports. If your network uses a service on a port different than the default, then it should be added in here to be captured. Figure 3.7 shows the Packetbeat's protocol configuration.

```

55 # ===== Transaction protocols =====
56
57 packetbeat.protocols:
58   - type: icmp
59     # Enable ICMPv4 and ICMPv6 monitoring. The default is true.
60     enabled: true
61
62   - type: amqp
63     # Configure the ports where to listen for AMQP traffic. You can disable
64     # the AMQP protocol by commenting out the list of ports.
65     ports: [5672]
66
67   - type: cassandra
68     # Configure the ports where to listen for Cassandra traffic. You can disable
69     # the Cassandra protocol by commenting out the list of ports.
70     ports: [9042]
71
72   - type: dhcpv4
73     # Configure the DHCP for IPv4 ports.
74     ports: [67, 68]
75
76   - type: dns
77     # Configure the ports where to listen for DNS traffic. You can disable
78     # the DNS protocol by commenting out the list of ports.
79     ports: [53]
80
81   - type: http
82     # Configure the ports where to listen for HTTP traffic. You can disable
83     # the HTTP protocol by commenting out the list of ports.
84     ports: [80, 8080, 8080, 5000, 8082]
85
86   - type: memcache
87     # Configure the ports where to listen for memcache traffic. You can disable
88     # the Memcache protocol by commenting out the list of ports.
89     ports: [11211]
90
91   - type: mysql
92     # Configure the ports where to listen for MySQL traffic. You can disable
93     # the MySQL protocol by commenting out the list of ports.
94     ports: [3306, 3307]
95

```

Figure 3.7: Packetbeat's Protocol Configuration

3.10 Configuring Snort

With the much flexibility snort provides we can configure it so suit any network. part of the configuration is to set the network variables such as the home network and external network, it is recommended to configure the external network as “any” since an attack can be initiated from within your own home network. Figure 3.8 shows the HOME_NET and EXTERNAL_NET values in the snort configuration.

```
51 ipvar HOME_NET 192.168.1.0/24
52
53 # Set up the external network addresses. Leave as "any" in most situations
54 ipvar EXTERNAL_NET any
```

Figure 3.8: Snort's Network Variable Configuration

the snort configuration must be set to create an output for snort to dump plaintext alerts into for the real-time python script to read, and a unified2 output file to be converted into Json in the next step as shown in Figure 3.9 .

```
534 #####
535 # Step #6: Configure output plugins
536 # For more information, see Snort Manual, Configuring Snort - Output Modules
537 #####
538
539 output alert_full: stdout
540 output alert_full: alert.log
541 output alert_fast: fast.log
542 output unified2: filename snort.log, limit 128, nostamp, mpls_event_types, vlan_event_types
543
544 # pcap
545 # output log_tcpdump: tcpdump.log
```

Figure 3.9: Snort's Output Configuration

The data supplied as logs needs to be converted to Json format before they are sent to Filebeat. This is due to multiple factors. In this format, log data is organized using JSON to get key-value pairs for the identification of structured data. In Elasticsearch, log data is stored in well-formatted structures, making it possible for the system to easily parse and search for the logs. On the same note JSON makes it easier to separately isolate certain information for analysis from log entries hence taking less time. To sum up, the format used in the logs consists of JSONspace separated items where each item can include much meta information. This metadata could extend to more than just the time stamp and the message types; it can include user identification number, IP address, or time stamp among others. Having that sort of material available has several benefits for security professionals, including the ability to conduct more detailed investigations into security events and network traffic. The conversion process is done using the u2Json tool from idstools. Figure 3.10 and Figure 3.11 show Snort and Filebeat running on the system.

```

--== Initialization Complete ==--

-.*- Snort! <*-
Version 2.9.7.0 GRE (Build 149)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 2.4 <Build 1>
Preprocessor Object: SF_POP Version 1.0 <Build 1>
Preprocessor Object: SF_FIPELNET Version 1.2 <Build 13>
Preprocessor Object: SF_DNP3 Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 9>
Preprocessor Object: SF_SSH Version 1.1 <Build 3>
Preprocessor Object: SF_DNS Version 1.1 <Build 4>
Preprocessor Object: SF_SDF Version 1.1 <Build 1>
Preprocessor Object: SF_INAP Version 1.0 <Build 1>
Preprocessor Object: SF_REPUTATION Version 1.1 <Build 1>
Preprocessor Object: SF_MODBUS Version 1.1 <Build 1>
Preprocessor Object: SF_SIP Version 1.1 <Build 1>
Preprocessor Object: SF_DCERPC2 Version 1.0 <Build 3>
Preprocessor Object: SF_GTP Version 1.1 <Build 1>
Preprocessor Object: SF_SSLLP Version 1.1 <Build 4>

Commencing packet processing (pid=3142)
[**] [1:527:8] BAD-TRAFFIC same SRC/DST [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
05/18-01:21:35.179000 0.0.0.0:68 -> 255.255.255.255:67
UDP TTL:64 TOS:0x0 ID:0 Iplen:20 DgnLen:310 DF
Len: 282
[Xref => http://www.cert.org/advisories/CA-1997-28.html][Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0016][Xref => http://www.securityfocus.com/bid/2666]

```

Figure 3.10: Snort Running

```

doffy@doffy-HP-Laptop: ~/filebeat-8.13.4-linux-x86_64
doffy@doffy-HP-Laptop:~/filebeat-8.13.4-linux-x86_64$ sudo ./filebeat run -e
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.300+0300","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.(*Beat).configure","file.name":"instance/beat.go","file.line":811},"message":"Home path: [/home/doffy/filebeat-8.13.4-linux-x86_64] Config path: [/home/doffy/filebeat-8.13.4-linux-x86_64] Data path: [/home/doffy/filebeat-8.13.4-linux-x86_64/data] Logs path: [/home/doffy/filebeat-8.13.4-linux-x86_64/logs]","service.name":"filebeat","ecs.version":"1.6.0"}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.301+0300","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.(*Beat).configure","file.name":"instance/beat.go","file.line":819},"message":"Beat ID: c9091b8d-67dc-4649-ac67-9e637d4d6dac","service.name":"filebeat","ecs.version":"1.6.0"}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.304+0300","log.logger":"seccomp","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/common/seccomp.loadFilter","file.name":"seccomp/seccomp.go","file.line":125},"message":"Syscall filter successfully installed","service.name":"filebeat","ecs.version":"1.6.0"}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.304+0300","log.logger":"beat","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.logSystemInfo","file.name":"instance/beat.go","file.line":1365},"message":"Beat info","service.name":"filebeat","system_info":{"beat":{"path":{"config":"/home/doffy/filebeat-8.13.4-linux-x86_64","data":"/home/doffy/filebeat-8.13.4-linux-x86_64/data","home":"/home/doffy/filebeat-8.13.4-linux-x86_64","logs":"/home/doffy/filebeat-8.13.4-linux-x86_64/logs"},"type":"filebeat","uuid":"c9091b8d-67dc-4649-ac67-9e637d4d6dac"},"ecs.version":"1.6.0"}}}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.304+0300","log.logger":"beat","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.logSystemInfo","file.name":"instance/beat.go","file.line":1374},"message":"Build info","service.name":"filebeat","system_info":{"build":{"commit":"b24ddd14c936c216817afed0cc7d0b23fd920194","libbeat":"8.13.4","time":"2024-05-06T06:34:12.000Z","version":"8.13.4"},"ecs.version":"1.6.0"}}}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.304+0300","log.logger":"beat","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.logSystemInfo","file.name":"instance/beat.go","file.line":1377},"message":"Go runtime info","service.name":"filebeat","system_info":{"go":{"os":"linux","arch":"amd64","max_procs":8,"version":"go1.21.9"},"ecs.version":"1.6.0"}}}
{"log.level":"info","@timestamp":"2024-05-18T01:24:12.306+0300","log.logger":"beat","log.origin":{"function":"github.com/elastic/beats/v7/libbeat/cmd/instance.logSystemInfo","file.name":"instance/beat.go","file.line":1383},"message":"Host info","service.name":"filebeat","system_info":{"host":{"architecture":"x86_64","boot_time":"2024-05-18T01:20:37+03:00","containerized":false,"name":"doffy-hp-laptop","ip":["127.0.0.1","::1","192.168.1.49"],"fe80::571d:70e3:be7a:d998"},"kernel_version":"5.15.0-107-generic","mac":["9c:7b:ef:15:c0:15","a4:fc:77:58:7e:af"],"os":{"type":"linux","family":"debian","platform":"ubuntu","name":"Ubuntu","version":"20.04.6 LTS (Focal Fossa)","major":20,"minor":4,"patch":6,"codename":"focal"},"timezone":"+03","timezone_offset_sec":10800,"id":"1ba04a91ac894952b11aa8d651f4cb8d"},"ecs.version":"1.6.0"}}}

```

Figure 3.11: Filebeat Running

3.11 Real-time Python Script

The real-time alert python script keeps reading from snort's plaintext output [9], and looks for any

```

8
9 f = open(PATH, 'r')
10 while True:
11     line = f.readline()
12     if line == "":
13         sleep(5)
14         continue
15     prio = int(line.split(" ")[13][0])
16     if prio <= 1:
17
18         yag = yagmail.SMTP(sender_addr, sender_password)
19         subject = 'Snort Alert'
20         body = line
21         yag.send(
22
23         to = receiver_email,
24         subject= subject,
25         contents=body
26
27

```

Figure 3.12: Real-time Alert Script

high priority reports, if found the script will send an E-mail alert to the network administrator, as shown in Figure 3.12.

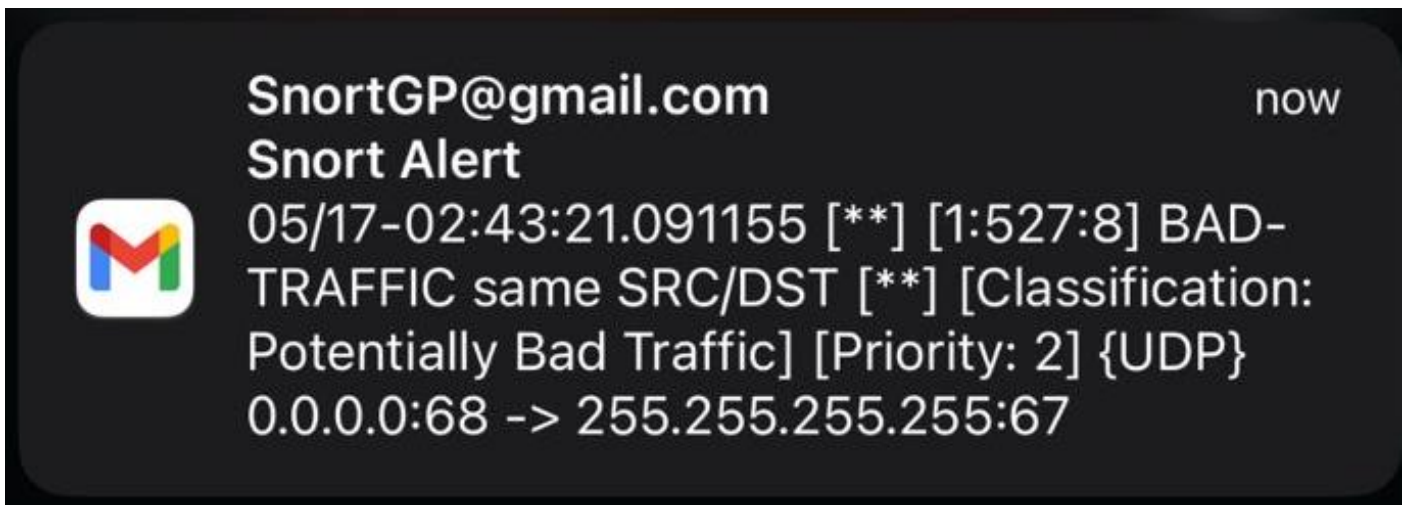


Figure 3.13: Example Alert

Figure 3.13 shows the alert received in case of a high priority threat (demo alert).

3.12 Python Scan Script

The scan script works by first scanning the network for all live hosts without any port-scanning, using different scanning techniques specified by the user. After which each host is sent to the port_scan function, inside this function we run the vulnerability scan and filter out the output. In the end of the scan we get the total number of live hosts, all the services available on them and if any vulnerabilities were found. Figure 3.14 shows the port scan performed on the live hosts.

```
9 def port_scan(host):
10     scanner = nmap.PortScanner()
11     results=scanner.scan(hosts = host, arguments= f"-vv -sN --host-timeout 5m -script vuln")
12     data = scanner.csv()
```

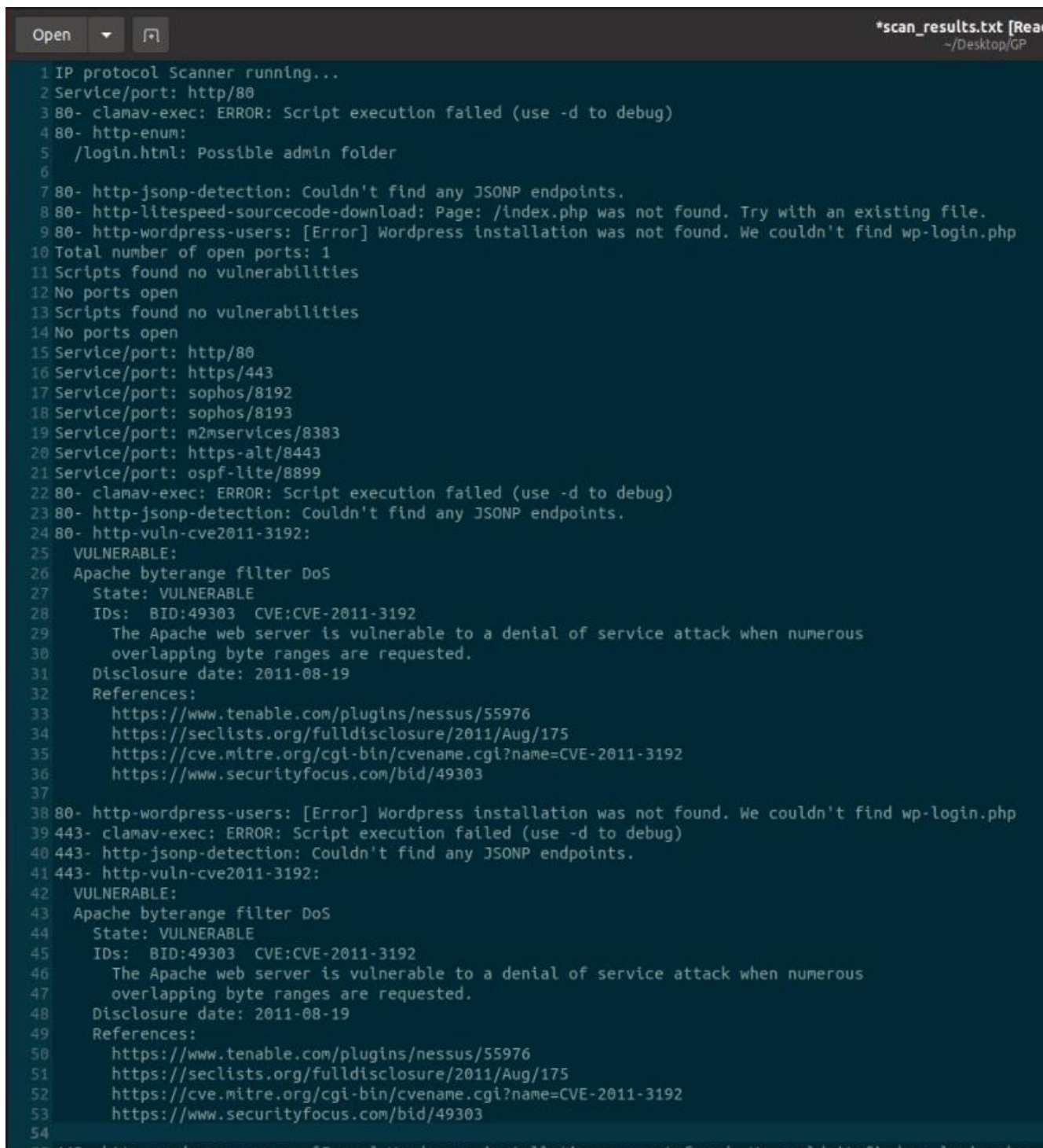
Figure 3.14: Port Scan Command

Figure 3.15 shows the host scan function and the different types of scans supported.

```
58 #Network host discovery scan that takes in one of 4 options (IP, ICMP, Timestamp or Netmask) and returns list of alive hosts and their total
59 def scan(mode = ''):
60     f = open(outputfile, "a")
61     opt = ""
62     if mode.lower().startswith("ip"):
63         opt = "-PO"
64         f.write(f"IP protocol Scanner running...\n")
65     elif mode.lower().startswith("net"):
66         opt = "-PM"
67         f.write(f"Netmask Scanner running...\n")
68     elif mode.lower().startswith("time"):
69         opt = "-pp"
70         f.write(f"Timestamp Scanner running...\n")
71     else:
72         f.write(f"ICMP Scanner running...\n")
73
74     scanner = nmap.PortScanner()
75     scanner.scan(hosts = TARGET_SUBNET, arguments=f'-sn -vv {opt}')
```

Figure 3.15: Host Scan Command

The script then outputs the scan results in a way to show each live host, what services they have running, and any vulnerabilities found in these services. The output text file can be seen in Figure 3.16 .



```
1 IP protocol Scanner running...
2 Service/port: http/80
3 80- clamav-exec: ERROR: Script execution failed (use -d to debug)
4 80- http-enum:
5   /login.html: Possible admin folder
6
7 80- http-jsonp-detection: Couldn't find any JSONP endpoints.
8 80- http-litespeed-sourcecode-download: Page: /index.php was not found. Try with an existing file.
9 80- http-wordpress-users: [Error] Wordpress installation was not found. We couldn't find wp-login.php
10 Total number of open ports: 1
11 Scripts found no vulnerabilities
12 No ports open
13 Scripts found no vulnerabilities
14 No ports open
15 Service/port: http/80
16 Service/port: https/443
17 Service/port: sophos/8192
18 Service/port: sophos/8193
19 Service/port: m2mservices/8383
20 Service/port: https-alt/8443
21 Service/port: ospf-lite/8899
22 80- clamav-exec: ERROR: Script execution failed (use -d to debug)
23 80- http-jsonp-detection: Couldn't find any JSONP endpoints.
24 80- http-vuln-cve2011-3192:
25  VULNERABLE:
26  Apache byterange filter DoS
27  State: VULNERABLE
28  IDs: BID:49303 CVE:CVE-2011-3192
29  The Apache web server is vulnerable to a denial of service attack when numerous
30  overlapping byte ranges are requested.
31  Disclosure date: 2011-08-19
32  References:
33  https://www.tenable.com/plugins/nessus/55976
34  https://seclists.org/fulldisclosure/2011/Aug/175
35  https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
36  https://www.securityfocus.com/bid/49303
37
38 80- http-wordpress-users: [Error] Wordpress installation was not found. We couldn't find wp-login.php
39 443- clamav-exec: ERROR: Script execution failed (use -d to debug)
40 443- http-jsonp-detection: Couldn't find any JSONP endpoints.
41 443- http-vuln-cve2011-3192:
42  VULNERABLE:
43  Apache byterange filter DoS
44  State: VULNERABLE
45  IDs: BID:49303 CVE:CVE-2011-3192
46  The Apache web server is vulnerable to a denial of service attack when numerous
47  overlapping byte ranges are requested.
48  Disclosure date: 2011-08-19
49  References:
50  https://www.tenable.com/plugins/nessus/55976
51  https://seclists.org/fulldisclosure/2011/Aug/175
52  https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-3192
53  https://www.securityfocus.com/bid/49303
54
55 443- http-wordpress-users: [Error] Wordpress installation was not found. We couldn't find wp-login.php
```

Figure 3.16: Script Scan Output

We gather threat intelligence using a python script in Figure 3.17 integrated with an API from here [21].

```

1 import requests
2
3 output_file = "int_results.txt"
4 f = open(output_file, 'w')
5
6 url = "https://pulsedive.com/api/info.php"
7
8 threat = input("what type of threat are you looking for?")
9 params = {
10     "threat": "zeus",
11     "get": "links",
12     "summary": "1",
13     "pretty": "1",
14     "key": "3ca73c270355958527af975e1e58415bc7630b5a5eb29fe7020cc638c41e0e"
15 }
16 params["threat"] = threat
17
18 url = url + "?threat="+params["threat"] + "&summary="+ params["summary"] + "&pretty=" + params["pretty"] + "&key="+ params["key"]
19 def get_api(url):
20     response = requests.get(url)
21     if response.status_code != 200:
22         f.write(f"Error connecting to api\n")
23         return
24
25     response_json = response.json()
26     f.write(f"Threat name: {response_json['threat']}\n")
27     f.write(f"Category: {response_json['category']}\n")
28     f.write(f"Risk: {response_json['risk']}\n")
29     f.write(f"summary: {response_json['wikisummary']}\n")
30
31 get_api(url)

```

Figure 3.17: Intelligence Gathering Script

Figure 3.18 shows the threat intelligence received.

```

1 Threat name: Zeus
2 Category: malware
3 Risk: high
4 summary: Zeus is a Trojan horse malware package that runs on versions of Microsoft Windows. It is often used to steal banking information by man-in-the-browser keystroke logging and form grabbing. Zeus
is spread mainly through drive-by downloads and phishing schemes. First identified in July 2007 when it was used to steal information from the United States Department of Transportation, it became more
widespread in March 2009. In June 2009 security company Prevx discovered that Zeus had compromised over 74,000 FTP accounts on websites of such companies as the Bank of America, NASA, Monster.com, ABC,
Oracle, Play.com, Cisco, Amazon, and BusinessWeek. Similarly to Koobface, Zeus has also been used to trick victims of technical support scams into giving the scam artists money through pop-up messages
that claim the user has a virus, when in reality they might have no viruses at all. The scammers may use programs such as Command prompt or Event viewer to make the user believe that their computer is
infected.
5
6
7

```

Figure 3.18: Intelligence Gathering Output

3.13 Python Configuration Script

A configuration script is used the first time the tool is started on a new system. It automatically does the necessary configuration and parameter setting for Snort, Filebeat, Packetbeat, and the scan script needed for the tool to run as expected. The user will be asked to enter required variables such as the home network, the cloud ID and the cloud user authentication created for the service. It will also ask the user to enter what Email address should the alerts be sent to. Figure 3.19 shows the snort and scan configuration part of the script.

```

34 def configure_snort_and_scan(home_ip):
35     #set the home network variable
36     #set the fast alert file
37     #set the network variable in the scan script
38     f = open("/etc/snort/snort.conf","a")
39     config = f'''
40
41 ipvar HOME_NET {home_ip}
42 output alert_fast: fast.log
43
44     ...
45     f.write(f"{config}\n")
46     f.close
47
48     f = open("/home/doffy/Desktop/GP/nmap_scan.py", 'r')
49     temp = f.read()
50     f.close()
51     f = open("/home/doffy/Desktop/GP/nmap_scan.py", 'w')
52     f.write(f"TARGET_SUBNET = '{home_ip}'\n")
53     f.write(temp)
54     f.close()

```

Figure 3.19: Snort and Scan Configuration

The main function, as shown in Figure 3.20 , is the function responsible for getting the variable data from the user and starting each configuration of the services.

```

def main():
    try:
        print(".....Configuration.....")
        print("#####\n")

        print("-----Filebeat configuration-----")
        cloud = input("Please Enter your cloud ID\n")
        user = input("Please Enter your user\n")
        password = input("Please Enter your password\n")
        configure_filebeat(cloud, user, password)
        print("#####\n")

        print("-----Packetbeat configuration-----")
        cloud = input("Please Enter your cloud ID\n")
        user = input("Please Enter your user\n")
        password = input("Please Enter your password\n")
        configure_packetbeat(cloud, user, password)
        print("#####\n")

        print("-----Snort configuration-----")
        ip = input("Please enter your home network with the subnet mask\n")
        configure_snort_and_scan(ip)
        print("#####\n")

        print("-----alerts configuration-----")
        email = input("Please enter the E-mail address you wish to receive alerts on\n")
        configure_alert(email)

        print(f"-----Configuration complete-----\n")
    except Exception as e:

```

Figure 3.20: Main of Configuration Script

Starting the tool will be done by using the start_tool script. It starts by giving the user the choice of which services they want to run: IDS, network monitoring, vulnerability scan, threat intelligence or any combination of the four. For each option selected, the script will run the necessary services and files. It also gives the user the ability to perform a vulnerability scan whenever they wish to, on top of the scan that occurs periodically. The u2json service required to convert unified2 logs into JSON format can be seen in Figure 3.21.

```

29
30 def u2json():
31     try:
32         subprocess.Popen(["idstools-u2json", "--snort.conf", "/etc/snort/snort.conf", "--directory", "/var/log/snort",
33             "--prefix", "snort.log", "--follow", "--output", "/var/log/snort/snort.json"])
34     except Exception as e:
35         print(f"Error occured while running U2json {e}")
36

```

Figure 3.21: Convert to JSON

Figure 3.22 shows the main function responsible for starting the necessary services based on the user's requested services. It also allows the user to perform an on-demand scan of the network for any vulnerabilities. If an error occurs while running the services an exit message will be displayed.

```

def main():
    print("#####")
    print(f'''
1-IDS
2-Network monitor
3-Vulnerability scan
4-Threat intelligence
    ''')
    choices = input("Please select which options you would like to run\n")
    for choice in choices.split():
        success = execute(choice)
        if not success:
            print("Exiting")
            return
    while True:
        check = input("Run scan? (y/n)")
        if check == 'y':
            scan()

main()

```

Figure 3.22: Main of Run Script

3.14 Elastic and Visualization

After reaching the fleet server, all the data is sent to Elastic's Kibana service which is used to filter the data and then visualize it in a meaningful way for the user to understand.

Inside Kibana we first must create a dashboard that holds all the visualizations as shown in Figure 3.23.

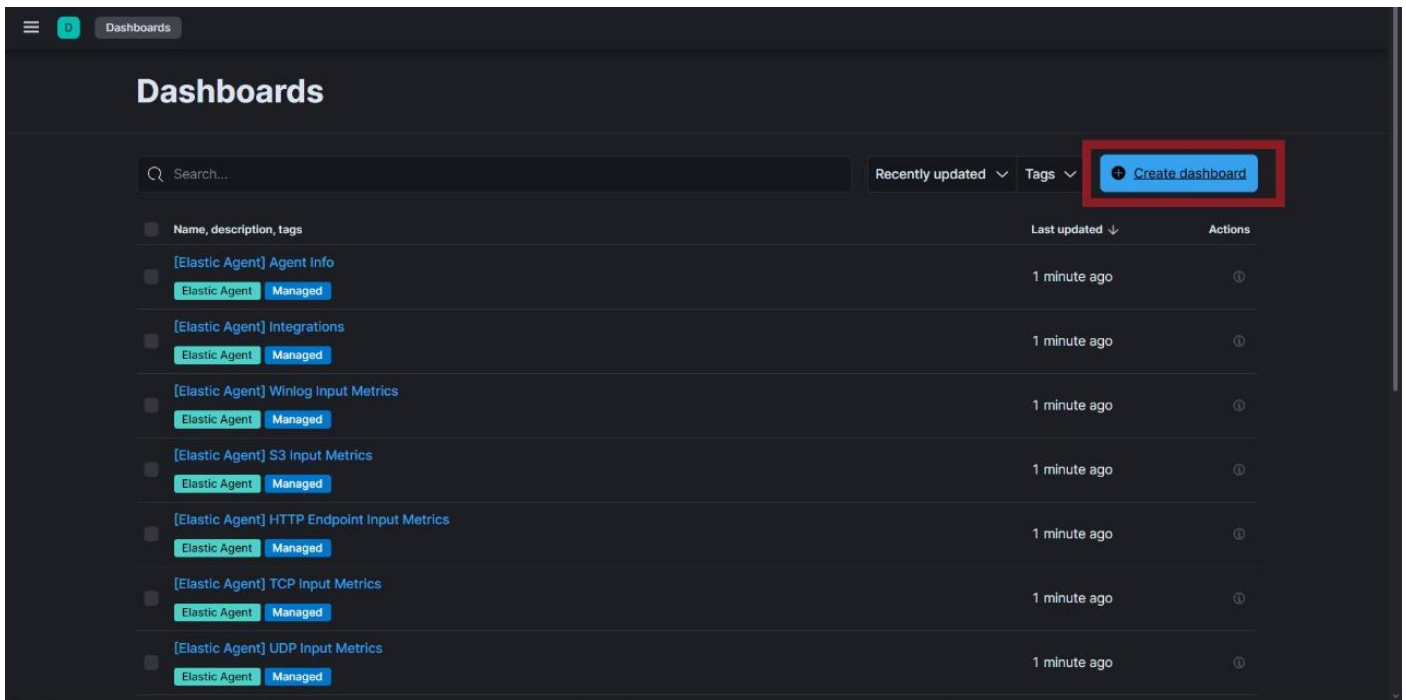


Figure 3.23:Dashboard Creation

After successfully creating our dashboard, we must make our own visualizations by pressing on ‘Create visualization’ as in Figure 3.24 to access all the fields and the graphs.

Inside this dashboard we navigate to the lens section where we can pick the indices we want to work on. In our case we are working on both the “.filebeat*” and “.packetbeat*” indices.

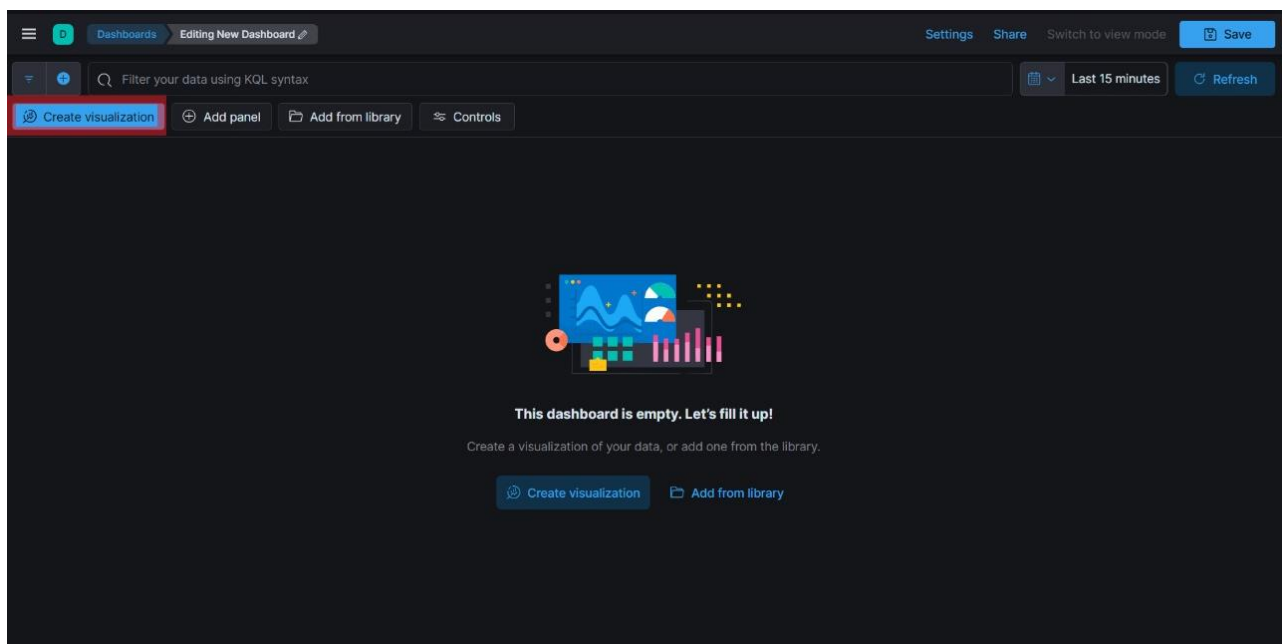


Figure 3.24:Lens Entry

Inside the “.filebeat*” index we find our snort logs where we can visualize them based on the received fields as in the Json format. There are many possible options depending on what you want to monitor/visualize. Figure 3.25 shows the lens page.

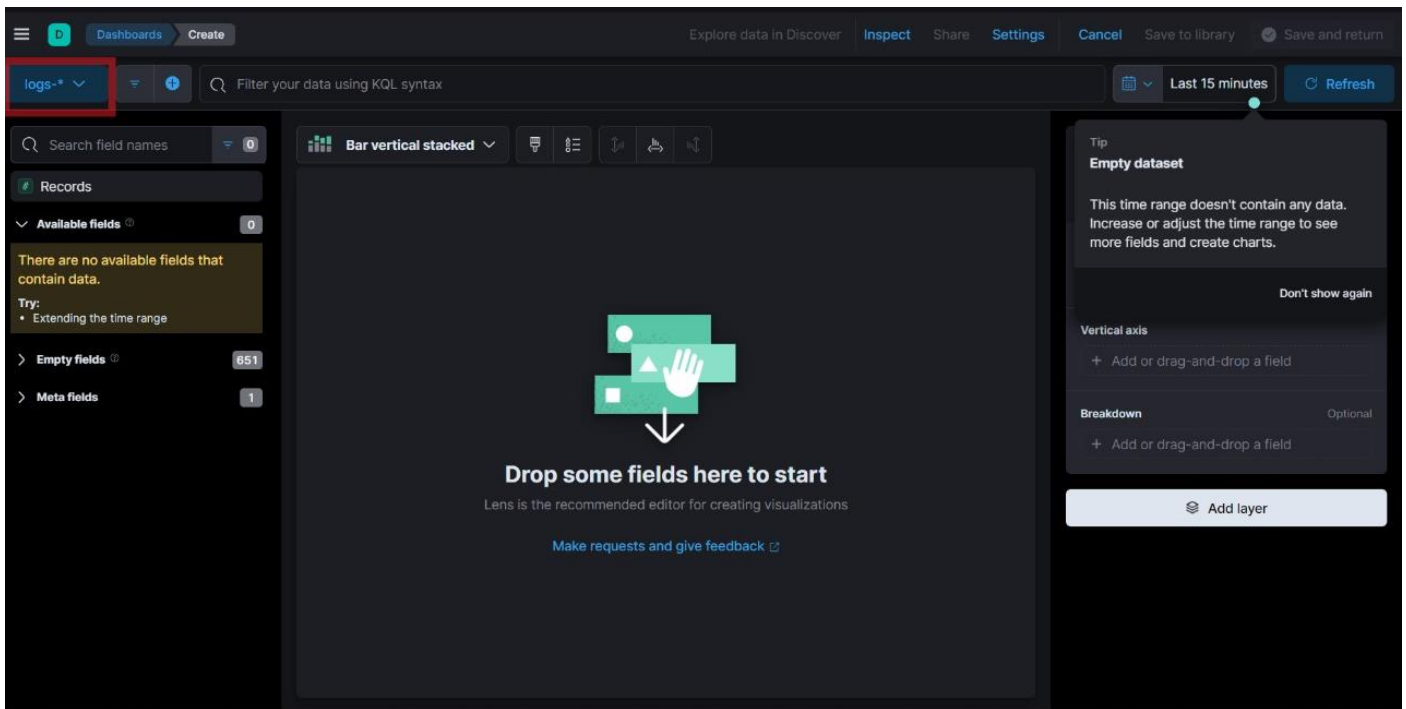


Figure 3.25: Choosing Indices

4 Results

It articulates the progression and achievements made while in the process of creating the tool. Here, the advantages of the application besides the performance in visualizing and analyzing Security & Networks are revealed. When making the analysis from the later data sources that the system has incorporated shall now be brought into consideration.

In the first place, in this chapter, it is described as fundamental principle starting how security data in the tool and how they can be evaluated. We are going to end the work by explaining the varieties of data that are loaded into the tool and then we will analyze the Kibana dashboards for the visualization of data. There are several types of the visualizations, which are included to these dashboards and their purpose is to reveal specific details of networks activity, security events, and threats. Finally, we will summarize the view on the validity of obtained results in terms of determining the overall effectiveness of the tool in case of occurrence identification and classification and in alerting to potential dangers besides reinforcing the comprehending about network security incidences. As demonstrated in this final comprehensive evaluation, the tool will indeed be useful in strengthening your organization's security posture.

4.1 Accessing The Dashboard

First the user will need to login into the elastic cloud account associated with the agent(s). From there to access the visualized data they must navigate to the analytics section and specify the correct dashboard, as shown in Figure 4.1, then the user will be able to access all the tables and graphs available.

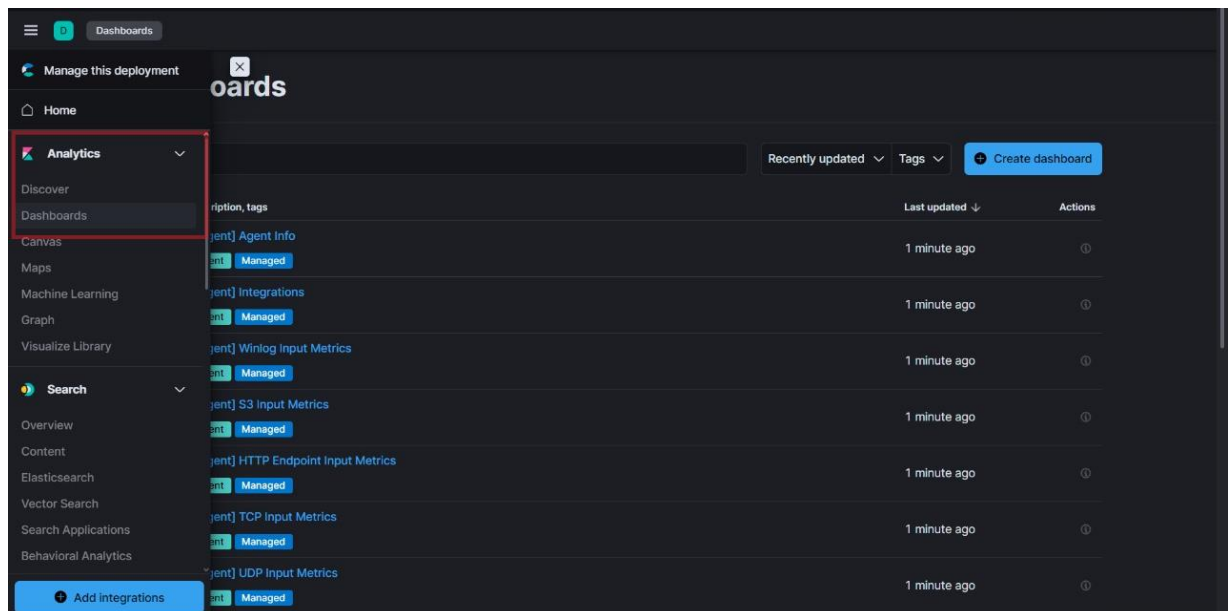


Figure 4.1: Analytics' Dashboards

4.2 Visualized Data

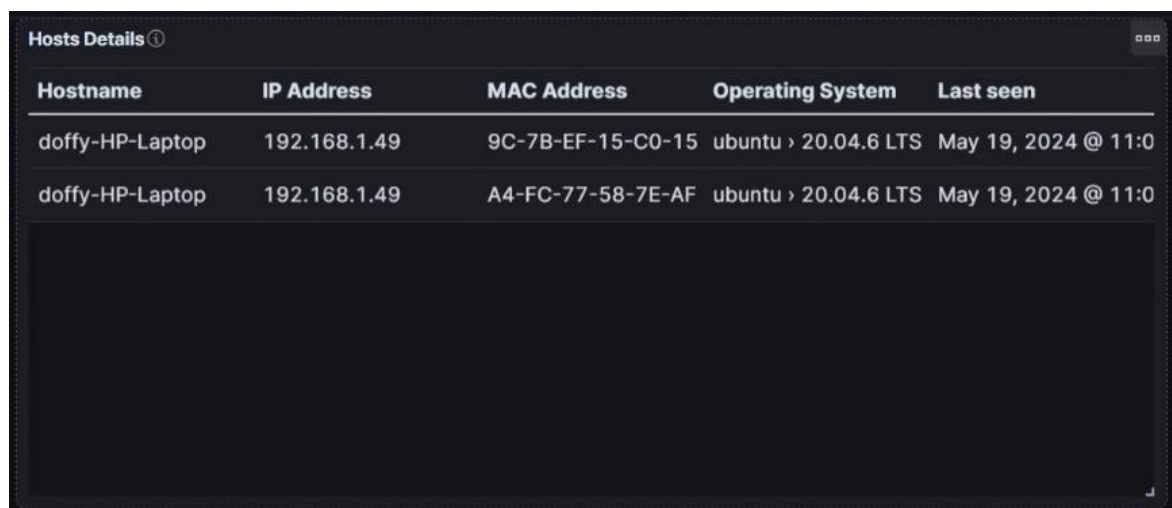


Classification	Destination IP	Source IP	Vlan ID	Time of Event
Potentially Bad Traffic	255.255.255.255	0.0.0.0	0	May 19, 2024 @ 11:02:57.786
Potentially Bad Traffic	ff02:0000:0000:0000:0001:ffed:d5ab	0000:0000:0000:0000:0000:0000:0000:0000	0	May 19, 2024 @ 11:02:57.781
Potentially Bad Traffic	224.0.0.1	0.0.0.0	0	May 19, 2024 @ 11:02:12.772
Potentially Bad Traffic	192.168.1.49	52.168.117.169	0	May 19, 2024 @ 11:00:37.753
Potentially Bad Traffic	ff02:0000:0000:0000:0001:ff05:4e06	0000:0000:0000:0000:0000:0000:0000:0000	0	May 19, 2024 @ 10:55:52.690
Potentially Bad Traffic	44.227.5.207	192.168.1.49	0	May 19, 2024 @ 10:37:05.533
Potentially Bad Traffic	192.168.1.160	192.168.1.49	0	May 19, 2024 @ 10:06:05.319

Figure 4.2: Event Details Table

Since it is important to quickly decide whether the identified event is dangerous, the event classification message defines the type of a security event. The fact that the destination and source IP addresses are used in investigation and mitigation protocols make it easier in identifying the source and target of the suspicious act. The timestamp is beneficial when determining the time of the security event while the VLAN ID assists in identify the affected segment of the network as we see in Figure 4.2. It will be easier to identify, investigate and prioritize events as security staff have the ability to comprehend details such as kind, source and duration of the security events because of the application of this combination of information.

The use of Kibana for analyzing results in Snort logs and a host details table. This network inventory table is practical when it comes to sustaining a list of identified devices since the critical information is available in one location. Specific data provided are: Hostname , MAC address of the NIC, TCP/IP address for networking. The most recent activity of the host picked from this table also as shown in Figure 4.3. The aggregated perspective of this proceeding helps security staff to be more prepared and capable of manipulating network devices and examine possible threats.



Hostname	IP Address	MAC Address	Operating System	Last seen
doffy-HP-Laptop	192.168.1.49	9C-7B-EF-15-C0-15	ubuntu > 20.04.6 LTS	May 19, 2024 @ 11:0
doffy-HP-Laptop	192.168.1.49	A4-FC-77-58-7E-AF	ubuntu > 20.04.6 LTS	May 19, 2024 @ 11:0

Figure 4.3: Host Details Table

The dashboard does not only include the Snort Log Data and host inventory table but also has additional widgets for core visualizations. Pie chart in Figure 4.4 can obtain security event types as well as their proportion and line graph in Figure 4.5 can illustrate priorities of the messages which can help in rapid assessment of presently active threats and changes in network security.

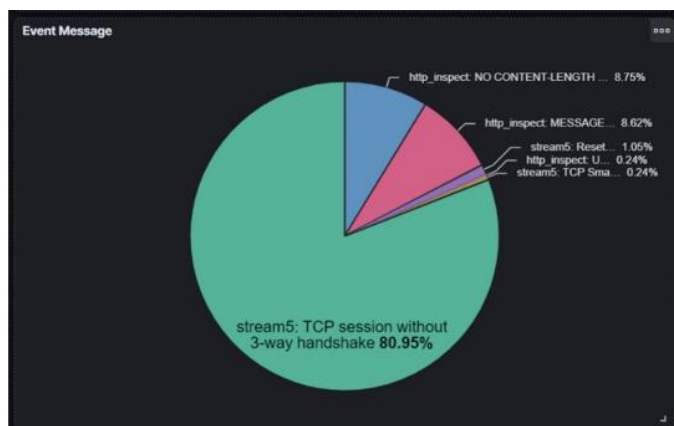


Figure 4.4: Event Message Pie Chart

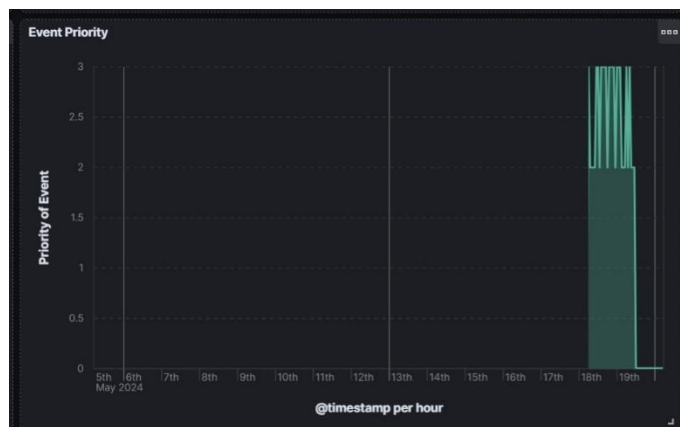


Figure 4.5: Event Priority Line chart

For traffic flow, traffic in and out both are shown through Bytes-in in Figure 4.6 and Bytes-out in Figure 4.7 line graphs. By the Y axis, the amount of data transported in bytes is depicted, and by the X axis, time in seconds, minutes, or hours might be depicted. The user could gain more insight on such configurations by looking at these graphs, and help him or her detect certain issues with the usage of the network.

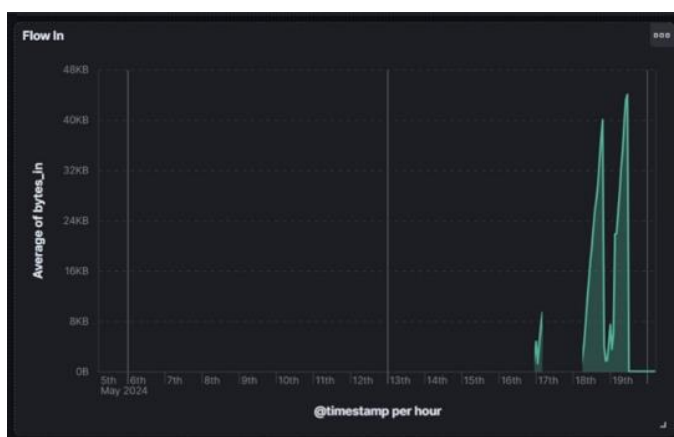


Figure 4.6: Average Bytes-in

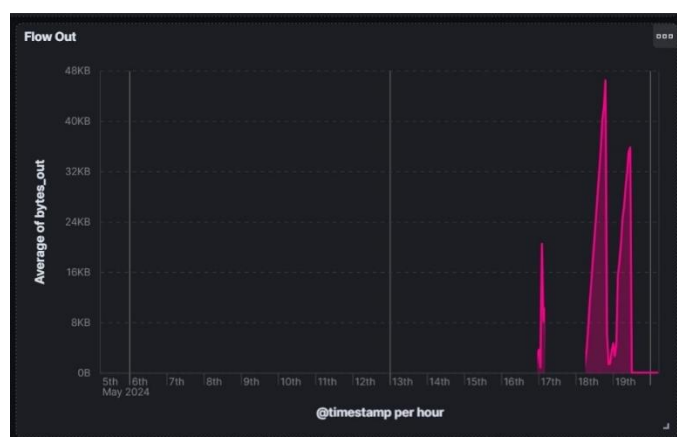


Figure 4.7: Average Bytes-out

The Kibana dashboard expands its network traffic analysis from the total bytes transmitted to the Average Packet Size as we see in Figure 4.8. This extra level of understanding can be useful for finding existing network problems. For example, changes in average packet size might suggest the use of large file transfers or changes in the type of protocols being used.

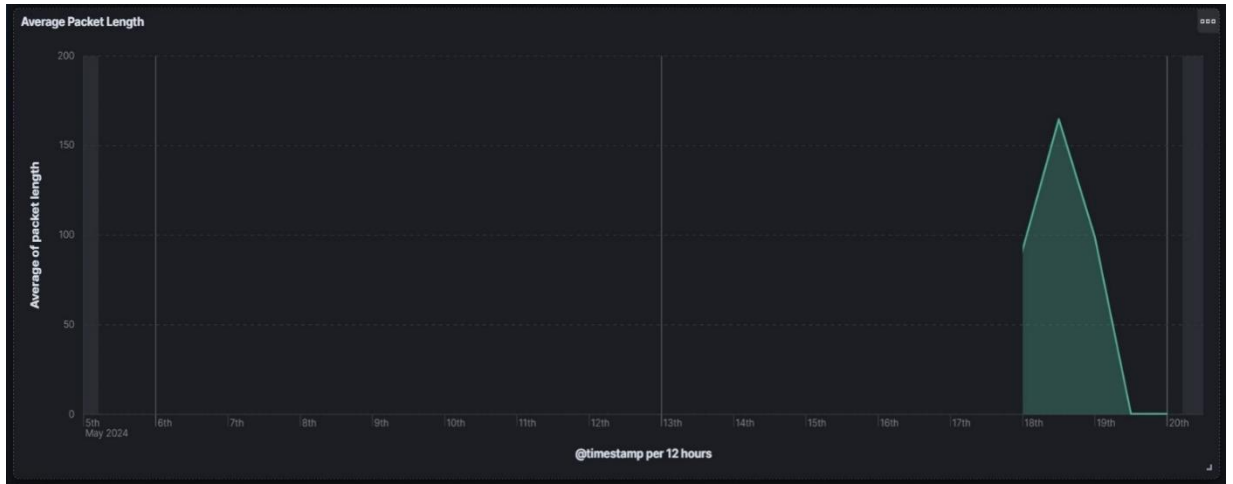


Figure 4.8: Average Packet Length

Figure 4.9 and Figure 4.10 contain two tables that provide unique features to monitor DNS activity within the network. These tables give further information on distinct forms of the queries and their responses occurring on the network.

The following tables serve to classify and illustrate the given DNS queries and answers throughout the network (e. g. A records, MX records etc.) In particular, query and answer types analysis show the basic activity of the users and devices in the network.

DNS Question Name	Query Type	Time
22972edfe2ec403a84fe2840244534c8.us-central1.gcp.cloud.es.io	A	May 19, 2024 @ 11:04:11.986
22972edfe2ec403a84fe2840244534c8.us-central1.gcp.cloud.es.io	AAAA	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1-v2.gcp.cloud.es.io	AAAA	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1-v2.gcp.cloud.es.io	A	May 19, 2024 @ 11:02:50.093
accounts.google.com	A	May 19, 2024 @ 11:03:49.106
accounts.google.com	HTTPS	May 19, 2024 @ 11:03:49.106
connectivity-check.ubuntu.com	A	May 19, 2024 @ 11:03:18.607
connectivity-check.ubuntu.com	AAAA	May 19, 2024 @ 10:59:24.260
cs22.wpc.v0cdn.net	HTTPS	May 19, 2024 @ 11:03:05.318

Figure 4.9: DNS Query Details Table

DNS Answers Name	Answers Type	Time
22972edfe2ec403a84fe2840244534c8.us-central1.gcp.cloud.es.io	A	May 19, 2024 @ 11:04:11.986
22972edfe2ec403a84fe2840244534c8.us-central1.gcp.cloud.es.io	CNAME	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1-v2.gcp.cloud.es.io	A	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1-v2.gcp.cloud.es.io	CNAME	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1.gcp.cloud.es.io	A	May 19, 2024 @ 11:04:11.986
proxy-production-us-central1.gcp.cloud.es.io	CNAME	May 19, 2024 @ 11:04:11.986
accounts.google.com	A	May 19, 2024 @ 11:03:49.106

Figure 4.10: DNS Answers Details Table

4.3 Validation of design requirements within the realistic constraints

Table 2:Validation of Design

	Parameter	Achieved
Design Requirements	The tool shall conduct performance analysis, including latency measurement, packet loss assessment, and bandwidth usage tracking	The tool tracks measurements indicating network performance.
	The tool shall integrate with external threat intelligence feeds and services to stay updated on the latest cyber threats and vulnerabilities.	The tool keeps updated with an external threat intelligence API to keep up with new threats.
	The tool shall collect and store network data, including connections, and traffic patterns.	The tool keeps all network flow statistics stored inside the Elasticsearch database.
	The tool shall have the capability to perform network discovery to identify active devices and services	A periodic or on demand scan occurs to discover all live hosts and services and find vulnerabilities on the system.
	The tool shall detect abnormal behavior or patterns in network traffic, potentially indicating security incidents or performance issues.	This requirement was achieved through the integration of the Snort IDS into the tool.
	The tool shall generate alerts and notifications for critical events, such as security breaches or performance anomalies.	The tool keeps track of alerts severity levels and sends an E-mail if a high priority alert occurs.
	The tool shall conduct vulnerability scans to identify potential security weaknesses in network devices and services.	A periodic or on demand scan occurs to discover all live hosts and services and find vulnerabilities on the system.
Economic Constraints	It will cost us the price of paid Cloud services.	For our project, Elastic's cloud service is required and be scaled to bigger networks.

5 Conclusion and Future Work

Our tool has provided the means for powerful and detailed examination of the existing network security by developing this useful Security Information and Event Management tool. The tool is effective in processing and correlating many inputs which include external intelligence data feeds, statistics of traffic flow and packet capture, results of vulnerability assessments and Snort traffic logs. Another proffered service, it gives one a simple way to watch security events, network traffic, vulnerability assessments, threat intelligence and potential threats through creation of visual Kibana boards.

The tool's efficacy has been effectively shown by this project in:

- **Proactive Vulnerability Management:** Thus, it will be possible to identify the flaws of the network infrastructure before it will be possible for them to be exploited and incorporated in the system is vulnerability scan results.
- **Improved Network Security Monitoring:** The general traffic scenario and characteristics of traffic anomalies may be explained by the network flow data. Thus, the system is capable of potential threats, for example, in the presence of malware in the network or attempts at unauthorized access, using analysis of data traffic.
- **Prioritized Threat Response:** It becomes possible to rank them according to the available information and probable outcomes in the process of their integration of data from multiple sources. This in a way helps security staff to attend to these incidents as and when they happened without having to misappropriate their resources on issues that are not very crucial.
- **A comprehensive perspective of network security:** the tool consolidates the presentation of security events, network, or potential risks and threats. Security staff can make better decisions for improving the position of a network security due to such deep insight.

Future improvements for the project include the following:

Possibly, extending the use of the approaches based on machine learning to diagnose the patterns of the network traffic and the vulnerabilities in the networks' topology may lead to the subsequent adoption of the methods based on the prediction and prevention of possible threats.

Augmenting the power one has for analytics in network flow data can assist in providing more and perhaps, improved results pertaining to traffic movement and discovery of intrusive attempts and perhaps threats.

It possible to expand the threat intelligence feeds to enhance the system's proactive defense strength by presenting more options on current threats and exposures to the network.

The system can turn into an even more powerful weapon, ensuring long-term stability and robustness of a network; the latter will continue developing and adopting such possible evolutions.

6 References

- [1] Caswell, B., Beale, J., & Baker, A. (2007). Snort Intrusion Detection and Prevention Toolkit. Syngress.
- [2] Talukder, S. and Talukder, Z., 2020. A survey on malware detection and analysis tools. International Journal of Network Security & Its Applications (IJNSA) Vol, 12.
- [3] Wang, Y., Bai, Y., Li, L., Chen, X. and Chen, A., 2020, June. Design of network vulnerability scanning system based on NVTs. In 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC) (pp. 1774-1777). IEEE.
- [4] Wang, Y. and Yang, J., 2017, "Ethical Hacking and Network Defense: Choose Your Best Network Vulnerability Scanning Tool," 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA), Taipei, Taiwan, pp. 110-113. DOI: 10.1109/WAINA.2017.39.
- [5] Laštovička, M., Husák, M., & Sadlek, L. (2020, April). Network monitoring and enumerating vulnerabilities in large heterogeneous networks. In NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium (pp. 1-6). IEEE.
- [6] Masumi, K., Han, C., Ban, T., & Takahashi, T. (2021, April). Towards efficient labeling of network incident datasets using tcpreplay and snort. In Proceedings of the eleventh ACM conference on data and application security and privacy (pp. 329-331).
- [7] Bajer, M. (2017, August). Building an IoT data hub with Elasticsearch, Logstash and Kibana. In 2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW) (pp. 63-68). IEEE.
- [8] Kleindienst, P. (2016). Building a real-world logging infrastructure with Logstash, Elasticsearch and Kibana.
- [9] Wuryani, R., Fenriana, I., Putra, D. S. D., Lasut, D., & Hariyanto, S. (2023). Network Security Analysis with SnortIDS Using ACID (Analysis Console for Intrusion Databases). *bit-Tech: Binary Digital-Technology*, 5(3), 145-154.
- [10] Chhillar, K., & Shrivastava, S. (2021). University computer network vulnerability management using Nmap and Nexpose. *International Journal*, 10(6).
- [11] Sani, J. (2023). Improved Log Monitoring using Host-based Intrusion Detection System. *AIJMR-Advanced International Journal of Multidisciplinary Research*, 1(1).
- [12] Brázda, J. Collection and Evaluation of Monitoring Data from the Kubernetes Environment.
- [13] Mahmood, M., & Shafi, Q. (2023). A Smart IDS in IoT System to Detect Zero-Day Intrusions Using Automated Signature Update.
- [14] Shah, N., Willick, D., & Mago, V. (2022). A framework for social media data analytics using Elasticsearch and Kibana. *Wireless networks*, 28(3), 1179-1187.

- [15] <https://github.com/jaegeral/security-apis>
- [16] <https://en.wikipedia.org/wiki/Elasticsearch>
- [17] <https://idstools.readthedocs.io/en/latest/tools/u2json.html>
- [18] <https://www.youtube.com/@Elastic>
- [19] <https://www.youtube.com/@AliYounesGo4IT>
- [20] [https://owl.purdue.edu/owl/research and citation/apa6 style/apa formatting and style guide/in text citations the basics.html#:~:text=APA%20citation%20basics,the%20end%20of%20the%20paper.](https://owl.purdue.edu/owl/research_and_citation/apa6_style/apa_formatting_and_style_guide/in_text_citations_the_basics.html#:~:text=APA%20citation%20basics,the%20end%20of%20the%20paper.)
- [21] <https://pulsedive.com>