Visualizing Network Intrusions

Final Report

Lam Pak Ning 3035563348

18 April 2022
Abstract

As network attacks have been growing in the recent decade, solutions like network intrusion detection systems (NIDS) are used to detect and analyse these attacks. However, as NIDS create text logs that are not easy to understand, visualization systems are used nowadays to help cyber security expert to analyse network attacks effectively. Thus, this project aims to contribute to the effort on visualizing network attacks, by studying 2 existing solutions – Splunk and Security Onion, and building a network security monitoring (NSM) solution, which improves on the existing systems. The solutions will be implemented in 2 parts – NIDS and Security Information and Event Management (SIEM) system: The NIDS will consist of Suricata and Zeek for generating network attack alerts and monitoring data; while the SIEM part will include the Elastic Stack for collecting, aggregating and visualizing the data from NIDS. The solution is now successfully implemented, which has shown its high effectiveness in helping users analyse intrusion through visualization.

Acknowledgement

The author would like to express gratitude to the project supervisor Dr Yip Beta for his guidance on exploring existing intrusions related solutions, and his advice on the scope of the project. They are essential for the project to stay feasible and process smoothly. The author would also like to thank professors from the Centre of Applied English Studies at the University of Hong Kong for providing guidance and recommendations for writing reports for this project.
# Table of Content

Abstract ............................................................................................................................................. 2  
Acknowledgement ................................................................................................................................. 2  
Table of Content ................................................................................................................................. 3  
Abbreviations ...................................................................................................................................... 4  
List of Figures ....................................................................................................................................... 5  
List of Tables ........................................................................................................................................ 6  
1 Introduction ....................................................................................................................................... 6  
2.1 Summary on Network Intrusions .................................................................................................... 7  
   2.1.1 Probes ....................................................................................................................................... 8  
   2.1.2 Privilege Escalation .................................................................................................................. 8  
   2.1.3 Denial of Service ....................................................................................................................... 8  
   2.1.4 Worm ...................................................................................................................................... 8  
2.2 Summary on Network Security Monitoring .................................................................................... 8  
   2.2.1 Network Intrusion Detection System ....................................................................................... 9  
   2.2.2 Security Information and Event Management System ............................................................ 9  
3. Methodology ..................................................................................................................................... 10  
   3.1 Studying Existing Solutions .......................................................................................................... 10  
      3.1.1 Splunk .................................................................................................................................... 10  
      3.1.2 Security Onion ....................................................................................................................... 10  
      3.1.3 Studying Setup ....................................................................................................................... 10  
      3.1.4 NIDS Tuning ......................................................................................................................... 11  
      3.1.5 Evaluation Criteria for Improvements .................................................................................... 12  
   3.2 NIDS data integration and storage ............................................................................................... 12  
      3.2.1 Development Setup ............................................................................................................... 12  
      3.2.2 Data integration and storage ................................................................................................. 13  
3.3 Data Visualization ........................................................................................................................... 14  
   3.3.1 Dashboard Design ................................................................................................................... 14  
   3.3.2 Dashboard Implementation and Evaluation ............................................................................. 14  
4 Discussion on Study ............................................................................................................................ 15  
   4.1 Study Result ............................................................................................................................... 15  
      4.1.1 Problem with Datasets ......................................................................................................... 15  
      4.1.2 Study Result on Implementation ......................................................................................... 16
4.1.3 Project Implementation ................................................................. 18
4.1.4 Study Result on Visualization ..................................................... 19
4.1.5 Project Visualization Features .................................................... 22
5. Evaluation ......................................................................................... 31
  5.1 Completed Work ............................................................................ 31
  5.2 Solution Evaluation Result .............................................................. 31
    5.2.1 Probe Attacks ............................................................................ 32
    5.2.2 Privilege Escalation Attacks ...................................................... 34
    5.2.3 DoS Attacks ................................................................................ 36
    5.2.4 Easy Deployment ....................................................................... 38
  5.3 Discussion on Result ................................................................. 39
    5.3.1 Note on encryption ................................................................. 39
  5.4 Difficulties .................................................................................... 40
    5.4.1 Features locked behind expensive licenses ................................. 40
  5.5 Future Work ................................................................................. 41
    5.5.1 Data Archival and Retention .................................................... 41
    5.5.2 Alert Feature ............................................................................ 41
    5.5.3 Testing on Network .................................................................. 41
6. Conclusion ....................................................................................... 41
References ........................................................................................... 42
Appendices ........................................................................................... 43
  1. Details on UNSW-NB15 and IDS2018 .............................................. 43
  2. Enabled/Disabled NIDS Rules ......................................................... 44
  3. Omitted Zeek Metadata Log Types ................................................... 45

Abbreviations

IoT Internet of Things

NIDS Network Intrusion Detection System

SIEM Security Information and Event Management

SPL Splunk Processing Language
List of Figures

Figure 1 Structure of studying setup.................................................................11
Figure 2 Number of generated alerts over time from importing UNSW-NB15.............15
Figure 3 Number of generated alerts over time from importing IDS2018................16
Figure 4 Implementation of Security Onion ......................................................16
Figure 5 Implementation of the project solution................................................18
Figure 6 Number of alerts over time.................................................................20
Figure 7 Information shown about a log field in Splunk and Kibana..........................21
Figure 8 SPL query for finding unusual size of ICMP sessions ..........................21
Figure 9 Common elements in dashboards .....................................................23
Figure 10 The alert dashboard ........................................................................23
Figure 11 Community ID navigation ................................................................24
Figure 12 Relation between the 10 metadata types ..........................................25
Figure 13 Navigation field for metadata ..........................................................25
Figure 14 Heatmap and Map visualization element ........................................27
Figure 15 KQL queries for probe attack anomaly detection .............................28
Figure 16 The 4 main elements for probe attack anomaly detection .................29
Figure 17 3 elements for dos attack anomaly detection ....................................30
Figure 18 Nmap scan detected .......................................................................32
Figure 19 Result from Zap and the solution’s detected probe alerts ..................33
Figure 20 Probe Alerts from real traffic ..........................................................33
Figure 21 Port Scan found by anomaly detection ................................................34
Figure 22 Sqlmap attack .................................................................................35
Figure 23 Privilege escalation attacks found ..................................................35
Figure 24 IP reputation of a top alerting IP ......................................................36
Figure 25 DoS attack result ............................................................................37
Figure 26 Visualisations of an anomaly in real traffic .......................................38
1 Introduction

In recent decades, online services become an inseparable part of our lives, which means these services often contain a massive amount of personal information and are essential to different organizations’ operations. Hence network attacks that target online services are increasingly prevalent, which often cause serious damages and bring risk to both the target organization and service users. One example of attack is data breaches, in which the attackers expose personal information stored in online services to bad actors. They can not only damage the organization’s reputation, but also put users at risk of identity theft. Couple with the fact that enormous data breaches (in order of 100 million data records) are more common now, with no sign of slowing down [1], the ability to detect and respond to network attacks has become increasingly important.

During a network attack, an attacker must interact with the target to perform tasks like gathering information or delivering malicious programs. Thus, it is possible to monitor, analyse and prevent these malicious interactions – or commonly known as network intrusions. Currently, Network Intrusion Detection System (NIDS) is used to perform this task. A NIDS functions by monitoring every network packet in a network, matching them against a set of pre-defined rules and generating alerts for possible intrusions. These alerts will be stored as text logs for security experts to analyse and take appropriate responses.

However, in the real-world, these alerts can come at a speed of 200,000 alerts per hour [2], making them unintelligible and impossible to analyse quickly. What is worse, with the growing popularity of 5G and (Internet of Things) IoT, the number of devices in organizations’ network keeps increasing, which also contribute to the growth in the number of alerts [2]. Thus, high-volume logs are now prevalent, and the effectiveness of NIDS is greatly hindered if the analysis is directly done with text logs.
To solve this problem, a system that can aggregate and visualize the alerts are used nowadays. As network intrusions often contain repeating or abnormal patterns when aggregated by attributes, like the source of intrusions or time period [3], visualization can help security experts to find these patterns effectively. To contribute to the ongoing effort on visualizing NIDS alerts, this project aims to study and reference existing visualization solutions to build a Network Security Monitoring (NSM) solution, which will include basic features for monitoring networks and visualizing intrusions, as well as enhanced features that can improve effectiveness in analysing and presenting network intrusions compared to existing solutions.

The visualization solution contains 2 major parts. The first part is the NIDS, which is used to produce alerts by monitoring inputs. These inputs include real and simulated network traffic from local network interface. Both inputs are used for evaluating the finished solution, but as real-world intrusions are uncontrollable, only the simulated ones are used for testing during development.

The second part is the Security Information and Event Management (SIEM) system, which is responsible for aggregating and visualizing logs from the NIDS. This part will include visualization elements like graphs, which are produced according to parameters like log fields and aggregation criteria. These parameters will be determined later by studying existing solutions.

In the following sections, this report first gives a summary on network intrusion types, as well as the structure and detection methods used by a typical NSM solution. Then, this report introduces the steps for studying existing solutions and establishing the design, implementation and evaluation method of the proposed solution. Next, this report discusses the findings on studying existing solutions and finalized project implementation. Lastly, this report detail the result of evaluation on the completed solution, difficulties and future work.

2.1 Summary on Network Intrusions

Network Intrusions attacks can achieve malicious goals via multiple approaches, but most intrusions can be classified as one of 4 types: [4]
2.1.1 Probes

Probe attacks work by scanning networks to find available computers or services, often by targeting a range of IPs/ports, then try to obtain information like type of service and software version. This enables attackers to better prepare future attacks, like choosing which vulnerabilities to exploit by matching the software version.

2.1.2 Privilege Escalation

Privilege escalation attacks work by exploiting a bug in software to obtain other’s access to data/services, like from normal user to root user or non-user to user. Some examples of these bugs include buffer overflow and race conditions.

2.1.3 Denial of Service

Denial of Service(DoS) attacks work by sending a large amount of invalid network traffic to the target, which overwhelms resource like router’s forwarding capacity or server’s memory and thus prevents normal traffic from accessing the target. For targets with more resources, Distributed Denial of Service(DDoS) attacks are used, where the invalid traffic will be sent by multiple hosts and hence can generate a more overwhelming amount of traffic to the target. Some common DoS attack types include TCP-SYN Flood, UDP Storm and ICMP Flood.

2.1.4 Worm

Worm attacks work by creating a worm (a program/chunk of codes) which can distributed itself from the infected computer to other computers on the network, which often cause exponential grow in the number of infected computers. The worm often has malicious function, like the WannaCry worm, which encrypts files on the infected computer.

2.2 Summary on Network Security Monitoring

Most NSM solutions include multiple parts with different functions, which are monitoring, detection, diagnosis, and response [5]. For this project, they are generalized into 2 parts: a NIDS for monitoring and detection, and a Security Information and Event Management (SIEM) system for diagnosis and response.
2.2.1 Network Intrusion Detection System

A NIDS is responsible for monitoring network activities and generating alerts for possible intrusion. It functions by inspecting every packet flowing in the network and detecting intrusion via 2 main detection methods which are misuse and anomaly detection [5].

2.2.1.1 Misuse Detection

This method uses patterns of network activity in past intrusion attacks, which are compiled into a set of rules, to match against current network activity. For example, a rule can specify a bit pattern in packet content which is known to be a privilege escalation attack, so when such content is detected in a packet, an alert is generated. Although this method is effective at finding known attacks, it highly depends on the accuracy of the rules, as any inaccuracy can lead to a high rate of false positives/negatives. Moreover, as this method depends on past data, it is of no use against novel intrusion attacks.

2.2.1.2 Anomaly Detection

This method uses a network activity profile that is generated from the normal activities, so that abnormal activity is regarded as intrusions. For example, to detect probe attacks, a threshold frequency of packets to a specific port can be set depending on past activity, so that any activities above the threshold are considered as an attack. The advantage is that it can detect novel intrusion attacks, but as this method assume all intrusions create abnormal activity, stealth intrusion attacks which disguise themselves as normal activities, are not detected.

Thus, both methods require extensive work to create the detection rules, so some existing rulesets are used for this project, which are further tuned to limit false positives/negatives.

2.2.2 Security Information and Event Management System

A SIEM system includes 2 parts which are Security Event Management (SEM), and Security Information Management (SIM). SEM’s function is to integrate data from multiple sources, which can then be visualized and analysed, while SIM’s function is to generate reports from the data and ensure regulatory compliance [5].
3. Methodology

This project will be completed in 3 phases. The first phase is studying existing solutions, which will involve analysing features provided by these solutions and determining possible areas of improvement. The second phase is NIDS data integration and storage, which will focus on combining data from NIDS for analysis and storage. The last phase is visualization, which will focus on building a dashboard that visualizes the data.

3.1 Studying Existing Solutions

To understand the features of existing solutions, 2 solutions are chosen: Splunk and Security Onion.

3.1.1 Splunk

Splunk is a commercial SIEM that can monitor networks, collect real-time data, and correlate data [5]. It is also regarded as the leading SIEM solution due to its wide compatibility with other systems [5]. Hence, it is chosen as the reference for current solutions due to its reputation, as well as its wide compatibility, which allows easy comparison between Splunk and the proposed solution by using the same NIDS logs as input. However, as Splunk is closed-source, some of its features may be difficult to replicate using only open-source tools, so it is used as a high-level reference for features of current solutions.

3.1.2 Security Onion

Security Onion is a Linux distribution that serves as an integrated package of multiple open-source tools [5], which can act as a NIDS for generating intrusion alerts, as well as combining network monitoring data from different tools to provide visualization. Thus, it is a valuable reference on how current open-source tools are used in synergy, which helps designing the implementation of the proposed solution by using similar sets of open-source tools.

3.1.3 Studying Setup

For studying Splunk and Security Onion, 2 virtual machines (VMs) will be set up, in which the 2 solutions will be installed. (See Figure 1)
For the NIDS, to facilitate comparison between solutions, they have to use the same set of NIDS tools to generate intrusion alerts, so another VM fyp needs to be set up with Suricata and Zeek, which are the default NIDS in Security Onion.

The input for both solutions is packet capture files from intrusion datasets UNSW-NB15 and IDS 2018 (For details on datasets see Appendix 1). However, there is a difference in how the input is used by the 2 solutions. For Security Onion, it already includes Suricata and Zeek, so the input can be directly processed; For Splunk, as it does not contain NIDS function, VM fyp is responsible for processing the input into logs, then the logs are sent to Splunk for visualization. The visualization result is displayed via the solutions’ web interface.

3.1.4 NIDS Tuning

As mentioned in the NSM summary, accuracy of the rules is important for NIDS. So popular rules that are well-maintained are used. For example, Suricata will be set up to use the Emerging Threats Open rules and Snort Community rules to generate alerts, with selective categories of rules enabled/disabled base on what applications/protocols/intrusion types exist in the datasets (For full list see Appendix 2). Zeek will also be set up to log network activities like HTTP/SSL/TCP to provide additional information. The rules will be tested on Security Onion first with the datasets to see if the intrusions are detected, then the rules will be copied to VM fyp for Splunk to use.
3.1.5 Evaluation Criteria for Improvements

To evaluate areas of improvement for the 2 solutions, some criteria for visualization and data aggregation are set according to past studies.

Users should be able to:

1. Detect change in number of logs over time, which is essential for analysing logs [2]
2. Detect data outliers with statistical methods, as they are often signs of intrusion [2]
3. Collectively observe different log fields for detecting patterns that involve multiple fields [2]
4. Aggregate data by large time scale (e.g. hour by hour) [2] [3] and location [3] to detect temporal and location patterns
5. Present the detected intrusion in an easy-to-understand manner for the purpose of reporting [3]

The result from the evaluation will be used to aid the design of the proposed solution.

3.2 NIDS data integration and storage

Moving to the implementation of the proposed solution, as Suricata and Zeek are the NIDS tools used in the study, the solution will continue to use them as the NIDS. The data produced by them will be integrated and stored with FileBeat and Elasticsearch.

3.2.1 Development Setup

Some VMs mentioned in the studying setup are reused. VM fyp is used for developing the NSM solution, with additional tools like Suricata, Zeek, Filebeat and Elasticsearch.

As mentioned in the introduction, the solution is tested with real-world and simulated intrusions, which is important for developing and evaluating the solution:

3.2.1.1 Real World Intrusions

For real-world intrusions, they can be monitored if the internet traffic is sent to VM fyp’s network interface directly. However, due to limitations in the author’s network setup, it is not possible. Instead, Suricata and Zeek are installed on the network router to monitor traffic and generate logs, which are then shared with VM fyp via a NFS share.
3.2.1.2 Simulated Intrusions

VM kali will be added for simulating intrusions, which run Kali Linux. Multiple tools are used to simulate 3 types of attacks:

1. Probe Attack – Nmap and ZAP
   Nmap is a network scanner that can identify and gather information about hosts, like open ports, and the services running on the ports.

   ZAP is a web application scanner, which includes functions like a web spider for indexing pages/APIs, automated/manual scan for vulnerabilities, and more.

2. Privilege Escalation - Sqlmap
   Sqlmap is an automated detection and exploiting tool for sql injection. It can scan a target for injectable parameters and execute attacks like getting the database schema and dumping table data.

3. Denial of Service - Slowhttptest
   Slowhttptest is a tool for conducting slow DoS attacks. One example is Slowloris, which works by opening many connections to the webserver, and then maintaining the connections by periodically sending only a small amount of data, causing the web server uses up resources and hence unable to accept other legitimate connections [6].

Worm Attack is not included as it requires a complex setup of multiple hosts for the worm to spread.

For the attack target, Metasploitable 2 is used. It is a vulnerable Linux distribution that contains many vulnerable services and a few web applications. It is deployed on VM fyp, so the solution will be able to monitor traffic to and from Metasploitable 2.

3.2.2 Data integration and storage

In the NSM solution, Elasticsearch and FileBeat are used for data integration and storage. They are a part of the Elastic Stack, which is the most popular free logging platform. Their features include powerful search functions on logs and built-it support for many types of logs
Thus, they are chosen to ease the difficulty in implementing similar searching features in existing solutions and speed up development.

Other storage solutions like MySQL are not considered, as logs have no relation storage requirement, relational database like MySQL are not suitable to store them. Moreover, these relational databases are developed for general data storage, so it has much fewer features compared to Elastic Stack, for example, as MySQL is just a database, it requires another solution for log processing, which takes extra time for configuration and testing; while Elasticsearch provide pre-defined ingest pipelines which are already closely integrated and can provide an “it-just-works” experience [7].

The details of the implementation of data integration and storage will be defined in later sections after the study result is detailed.

### 3.3 Data Visualization

Kibana is used to build dashboards for visualizing the integrated data, as it is a part of the Elastic Stack, so Kibana can automatically fetch data from ElasticSearch and development is greatly simplified.

#### 3.3.1 Dashboard Design

A dashboard design will be created, consisting of which attributes of the data to include (e.g. alert type/traffic amount), and which visualization features to include (e.g. type of graphs). The details of the design will be defined after the study on existing solutions is done.

#### 3.3.2 Dashboard Implementation and Evaluation

The dashboard will be implemented and tested with network traffic (simulated and real-world) as mentioned in the introduction, and then evaluated by checking whether the dashboard satisfies all criteria defined in the previous section.
4 Discussion on Study

This section will first discuss the project’s study result, then the visualization design and implementation details based on the study result.

4.1 Study Result

4.1.1 Problem with Datasets

As mentioned in the methodology section, 2 datasets, UNSW-NB15 [8] [9] and IDS 2018 [10], are used for the evaluation of the 2 solutions. For a dataset to be used, it must contain meaningful patterns that provide insight into the attacker’s activities. So they are imported to Security Onion to check whether such patterns exist.

4.1.1.1 UNSW-NB15

![Figure 2 Number of generated alerts over time from importing UNSW-NB15, showing the low count](image)

The first thing to note is the unexpected number of alerts, which is much lower than the actual number of intrusions-related packets in the dataset (~1000 vs ~100000) (see Figure 2). A likely cause is that intrusions used in the 2015 dataset are not included in the current (2021) NIDS rules. Thus, with such a low detection rate, it is not useful for observing intrusion patterns.
4.1.1.2 IDS2018

Figure 3 Number of generated alerts over time from importing IDS2018, showing the high count and spike patterns

For this dataset, the number of alerts is much higher, so patterns can be observed more easily. For example, users can detect spike patterns in alerts, which corresponds to network scans launched by the attacker (see Figure 3).

Due to this discovery, only IDS2018 is used to evaluate the 2 solutions.

4.1.2 Study Result on Implementation

As mentioned in the methodology section, Security Onion is used for studying the implementation of a NSM solution, so below are the key components of Security Onion (See Figure 4).

Figure 4 Implementation of Security Onion, with the key components and interactions between them.
The components can be divided into 3 groups:

1. Network monitoring – Suricata and Zeek
2. Data processing and storage – FileBeat and ElasticSearch
3. Visualization and analysis – Kibana and Security Onion Console

4.1.2.1 Network Monitoring

To monitor network traffic, Suricata NIDS is used for generating intrusion alerts, while Zeek is used for generating network metadata, like HTTP headers in every HTTP connection. Traffic can be fed to Suricata and Zeek either by importing packet capture files or listening on a network interface. After the traffic is processed, output logs are saved as text files in JSON format.

To aid users in finding correlated alerts and metadata logs, Community ID is generated for the logs, which is a hash of a network flow based on information like source/destination IP and port [11]. With Community ID, the task of correlated logs of the same network flow is simplified from matching multiple fields to just one field.

4.1.2.2 Data Processing and Storage

For processing and storing the logs from Suricata and Zeek, FileBeat is responsible for monitoring file directory for new logs and sending them to ElasticSearch via HTTP request to ElasticSearch’s API.

Once ElasticSearch receives the logs, they are processed and enriched according to custom “Ingest Pipelines”. For example, field names in the logs are renamed to standardized names, and locations of IP addresses are added via the GeoIP database. Finally, the processed logs are saved in the database (or index) in ElasticSearch.

4.1.2.3 Visualization and Analysis

Users can visualize and analyse the data via the web interface of Kibana, which obtain data by querying ElasticSearch via its API. Another web interface, the “Security Onion Console”, also exists, which provides functions for detailed analysis of network traffic, like inspecting and decoding each packet.
4.1.3 Project Implementation

The implementation of the project includes using Suricata, Zeek, FileBeat, ElasticSearch and Kibana, which are used in similar ways as in Security Onion, with a few minor differences:

1. Instead of custom ingest pipelines in Elasticsearch, predefined pipelines for Suricata and Zeek are used, which comes with FileBeat by default. These pipelines follow the Elastic Common Schema (ECS), which is an open-source specification of fields used for logs storage in Elasticsearch [12].

2. In Security Onion, the abovementioned Community ID is generated by Suricata and Zeek independently. However, in the project solution, it will now be generated by Elasticsearch, since Community ID is part of the ECS specification [12]. This requires no configuration, which saves the need for setting the same Community ID seed in Suricata and Zeek.

The 5 applications are deployed using docker containers (see Figure 5), which brings the benefit of easy deployment, so users do not have to install the applications manually.

![Diagram](image)

*Figure 5 Implementation of the project solution, showing the 5 applications as docker containers*

The Filebeat container is responsible for loading the pre-defined visualizations of the solution into Kibana, and for preconfiguring Kibana. It is selected due to the included tools for loading visualization to Kibana. Bash scripts are created to run the tool and add other customization via the Kibana API during first-time start-up.
To deploy the solution, users must specify the capture network interface and directory for saving the alert and metadata logs, via creating a .env file. Other optional configuration includes password for Elasticsearch and Kibana, Kibana port, trusted subnet(s), and UID/GID for permission of the log folder. If trusted subnet(s) is not set, the default is the private IP subnets, as it is needed for Suricata and Zeek to monitor traffic correctly, and also for filebeat to decide traffic direction correctly.

Once deployed, the solution automatically starts monitoring network traffic on its network interface and generated logs are visualized on the Kibana web interface. If required, users can modify the configuration of Filebeat, Kibana, Suricata and Zeek via config files, followed by a container restart.

4.1.4 Study Result on Visualization

4.1.4.1 Result Based on Predefined Criteria

To learn about visualization features and identify areas of improvement, the 2 solutions are evaluated and compared based on the 5 criteria mentioned before.

One thing to note is that both Splunk and Security Onion offer high customization ability, so users can add visualization features themselves. Hence, the study will only focus on predefined visualizations in the 2 solutions, which are visualizations in Stamus Networks App in Splunk; and Security Onion’s predefined visualization in Kibana.

The result is summarised in Table 1

<table>
<thead>
<tr>
<th></th>
<th>Splunk/Stamus Networks App</th>
<th>Security Onion/Kibana</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show change in no. of logs</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create report of intrusion</td>
<td>Yes, export as pdf</td>
<td>Yes, export as link/pdf(paid), Canvas Feature</td>
</tr>
<tr>
<td>Collectively observe fields</td>
<td>No by default</td>
<td>No by default</td>
</tr>
<tr>
<td>Aggregate by large time scale/location</td>
<td>No by default</td>
<td>No by default</td>
</tr>
<tr>
<td>Show outliers</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 1 Result of study on visualization features*
1. Show change in the number of logs

Both solutions offer this feature. For example, in Splunk, the number of Suricata alerts over time can be shown (see Figure 6).

![Figure 6 Number of alerts over time, with alert on Nmap port scan highlighted](image)

2. Creating easy to understand reports

Both solutions offer this feature. For Splunk, the visualization dashboard can be exported as PDFs, and for Kibana, the dashboard can also be exported as an URL or pdf. Kibana also offers a more advanced reporting feature called Canvas, which allows users to create stylized visualization with custom designs.

3 & 4. Collectively observe fields, Aggregate by large time scale/location

The predefined visualizations for both solutions do not provide such features, but it is possible for users to add them.

5. Show outliers

Only Splunk offer this feature, which allows users to quickly access statistic of a log field, like average and standard deviation. In Kibana, it only shows the top values of a log field, which does not help detect outliers (see Figure 7).
4.1.4.2 Other Discovered Features

1. Anomaly Detection with Splunk Processing Language

Splunk Processing Language (SPL) is included with Splunk, which provides functions like searching, filtering, inserting computed fields, enrichment, statistics and more. Hence, it is possible to detect anomalies by filtering out logs with a threshold.

For example, a predefined query can identify ICMP anomaly, by filtering for flow with a large amount of traffic, which allows users to quickly decide which logs are worth investigating (See Figure 8).

Figure 8 SPL query for finding unusual size of ICMP sessions

As anomaly detection can help users detect intrusions that are missed by the alert rules in Suricata IDS, the project solution should include some form of anomaly detection.
4.1.5 Project Visualization Features

Based on the result of the study, the project solution design contains 3 main visualization feature groups.

4.1.5.1 Visualization of Logs

To visualize intrusion alerts by Suricata and network metadata logs by Zeek, multiple dashboards are built. For the metadata logs, only some types such as connection, HTTP and SSL are visualized. Other metadata types are omitted due to the low number of logs generated with simulated and real traffic (see Appendix 3).

To explain the details of the dashboard design, they are divided into common elements, alert dashboards, and metadata dashboards, with detail as follows:

1. Common Elements

These dashboards all contain 4 basic elements: a search bar on top that allow users to input query with Kibana Query Language (KQL) and select time range, a navigation panel for linking dashboards, a panel for the total number of logs, and a histogram for showing log count change over time. Tables of source IP, destination IP and destination port are also added, with source port excluded as it is mostly randomized (see Figure 9).

Note that these tables are not present in some dashboards, as the data is often repeated in other fields (e.g. DHCP assigned address is the same as the source IP) or only involves a single value (NTP destination port is always 123).

Most of the dashboard elements have interactions available that help users refine their search, like dragging on a histogram applies a time filter, and clicking on a table/pie chart applies a filter on the clicked field. IP addresses are also clickable, which takes users to talosintelligence.com, an IP reputation search tool, which helps users to verify whether the IP is malicious.
2. Alert Dashboard

For the alert dashboard, other elements are also added. First, some visualization elements for providing extra data, including categories of the alerts on the histogram, and top alert rule names in a table. Secondly, interactive elements are also added for users to filter alerts in the dashboards, like control elements for filtering alert severity and category/group of categories (see Figure 10). Lastly, navigation fields are added for users to find related metadata logs via community ID, which takes them to the connection metadata dashboard (see Figure 11).

Figure 9 Common elements in dashboards: 1) Navigation Panel, user can click the link to other dashboards; 2) Log Count; 3) Tables for top source IP and destination IP/port; 4) Histogram on log count over time, note it also includes number of different protocols in each period

Figure 10 The alert dashboard: 1) Links to dashboards that only include categories for an attack type; 2) Control for selecting alert severity; 3) Categories of the alerts, which also has option for users to use them as filters; 4) Table of the top alert names
3. Metadata Dashboards

Moving on to the metadata dashboards, 10 types of metadata logs are visualized with elements like histograms, tables, and pie charts. Below is the summarized information of each dashboard:

<table>
<thead>
<tr>
<th>Metadata Type</th>
<th>Visualized Data (excluding source/destination IP/port)</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Protocol, Connection state (e.g. whether a connection is terminated properly), Location and Organization of source IPs</td>
<td>This is not limited to TCP connection, but also other protocols like UDP/ICMP due to how Zeek defines it. Visualization of locations of IP is explained later</td>
</tr>
<tr>
<td>DHCP</td>
<td>Message type, Sender’s MAC address, Assigned/requested IP, Lease time</td>
<td></td>
</tr>
<tr>
<td>DNS</td>
<td>Query type, Response code, Query question &amp; answer data</td>
<td></td>
</tr>
<tr>
<td>HTTP</td>
<td>Request method/Domain/Path/User agent, Response code</td>
<td></td>
</tr>
<tr>
<td>Files</td>
<td>File type, Filename, File source</td>
<td>File source means what type of metadata log the file is related to</td>
</tr>
<tr>
<td>NTP</td>
<td>None other than common ones</td>
<td></td>
</tr>
<tr>
<td>SIP</td>
<td>Method, URI, User agent, Request from/Response from</td>
<td></td>
</tr>
</tbody>
</table>
These metadata types are related to each other mainly through a shared connection log as root (see Figure 12). Thus, a navigation field is added to the connection dashboard for users to find them easily (see Figure 13).

**Table 2 Brief information of metadata dashboards**

<table>
<thead>
<tr>
<th>SSH</th>
<th>Server/Client name</th>
<th>Auth result</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL</td>
<td>Server name</td>
<td>Cert validation status</td>
</tr>
<tr>
<td>X509</td>
<td>Cert Issuer/Subject Common Name/Valid Period</td>
<td>Cert SAN DNS</td>
</tr>
</tbody>
</table>

**Figure 12** Relation between the 10 metadata types, note X509 log is only related to fluid in file logs.

**Figure 13** 1) At the bottom of the connection dashboard, there is a navigation field to another dashboard; 2) In the linked dashboard, a filter on session id is applied, so user can see what other log types are available, and see them by clicking links in the Navigation element on the left.
Note that navigation field to X509 logs is not added, as with TLSv1.3 becoming the major encryption standard, certificates are also encrypted, so most certificates are not captured, and a navigation field mostly returns a filter with no result found. If users do need to find a captured certificate, they can go to the X509 dashboard and filter the “file.x509.subject.common_name” field.

4.1.5.2 Improvements Based on Study

Based on the study result on visualization features, there are 3 main areas of improvement that need to be added to the project solution:

1. **Add show outliers feature in Kibana**

   Although the original plan is to add UI element in the log view page that shows statistics of fields, like the one in Splunk, after reviewing the documentation of Kibana, such modification cannot be easily done. The closest feature is runtime fields, which allow the creation of new fields by applying transformation on other fields, like extracting day of week from the timestamp field [13]. However, as runtime fields only have access to a single log entry, calculating statistics is not possible.

   Thus, instead of calculating statistics, outliers are shown as spikes on graphs. This visualization is included in the anomaly detection dashboards, which are explained in detail later.

2. **Add the missing visualization feature - Collectively observe different fields**

   This is added to the bar chart for some dashboards. For example, in the connection dashboard, the number of logs is grouped by protocols, so users can observe connection count, time, and protocols at the same time (see Figure 9).

3. **Add the missing visualization feature - Aggregate by large time scale and by IP location**

   For aggregating by large time scale, a custom heatmap visualization element is created using Vega which is a visualization grammar [14]. The heatmap aggregates hour by hour the number of connection logs over 7 days, so users can easily compare
the same period of each day (see Figure 14). The heatmap also allow users to click a tile, which applies a filter by the selected period.

For aggregating by IP location, the map visualization element in Kibana is used. Locations of IPs are shown as dots on the map, with their size representing the number of logs at the location. The map also allows users to draw an area over the map as a filter, so only logs in the area are shown (see Figure 14).

For the other existing features, show change in the number of logs exists as the common histogram element. While for creating reports, the canvas feature of Kibana already included several examples by default.

4.1.5.3 Anomaly Detection

As mentioned, anomaly detection dashboards are added to help discover intrusions missed by Suricata. Hence, 2 types of attack, probe (specifically port scans) and DoS, are chosen to be analysed using this method. They are chosen as their variety is lower compared to other attacks like privilege escalation and worm, so it is easier to set constraints and filter out unrelated connections. This is curial for making the anomaly stands out in visualization, so users can identify them with ease. The details of the constraints and visualization are detailed below.
1. **Probe (Port Scans)**

To narrow down the logs to visualize, 2 constraints are set. First, all outbound connections, meaning from trusted networks to the internet, are filtered out.

Second, constraints are applied to TCP, UDP and ICMP protocols. For TCP, as most probe methods do not perform a complete handshake with the target [15], connections that handshake properly can be filtered out. For UDP, inbound traffic that only contains packet from the source and no reply is included, as this could mean either the port is closed or the traffic is a possible probe. As for UDP inbound traffic with response from the destination, since UDP has no handshake protocol, it is difficult to separate probe and legitimate traffic just by observing the packet flow sequence, so they are excluded to avoid including legitimate traffic. For ICMP, only ping requests are included.

Based on the constraints, the KQL queries are as follows:

**Figure 15 KQL queries for probe attack anomaly detection**

**Explanation of query:**

For TCP/UDP:
- **ShA***: Handshake complete
- **ShD***: Handshake started then received a packet from source
- **Fa***: Handshake done before, this is termination
- **Da***: Midstream packet follow by ACK from destination
- **Dd***: UDP traffic with reply

For **ICMP**:
- **icmp.type**: 8: Ping request

For visualization, 4 main elements are used (see Figure 16):

1. An hour-by-hour heatmap of the number of possible probe connections.
2. A line chart for the number of unique destination ports over time, with the number of probe related alerts as bars. The unique count is used as probe attacks are likely to scan a large portion of ports. While the alerts count can be used to help validate the anomaly.

3. A heatmap for the number of connections to groups of ports over time. The interval for the groups is smaller for lower ports as they are more commonly used by services and hence likely the target of probes.

4. A line chart for the number of ping requests over time.

---

2. DoS attacks

As DoS are mainly “loud” attacks, only a simple constraint - no outbound connections - is needed.

As for visualization, 4 main elements are used (see Figure 17):

1. An hour-by-hour heatmap of the number of connections.

2. A line chart for the number of connections grouped by top 5 ports over time, with the number of DoS related alerts as bars. Grouping is used as DoS attacks are likely to only target a few ports. While the alerts count can be used to help validate the anomaly.
3. A line chart for the number of unique IPs over time. This is used as real DoS are likely launched from botnets, meaning many IP addresses, which can help identify an attack.

4. A line chart for the 90th and 70th percentile connection duration over time. This is used to identify slow DoS attacks. The 2 percentiles are used as long duration traffic can be legitimate, like large file download, so if there is a spike only the higher one, it could just be due to these traffic. However, if both percentiles spiked and are at a similar level, it means an increased amount of traffic has an unusually long duration and indicates a possible slow DoS attack. Note that the number 90/70 is obtained by testing on traffic to ensure the lower percentile does not spike often, so the number requires tuning on other networks.

Figure 17 3 elements for dos attack anomaly detection, 1 is omitted as it is the same one in probe attack. Note the highlight pop-up, which shows up after user drag-select a period on the line chart. User can choose to zoom in or view the period in the connection dashboard for more detail. The pop-up also exists in the probe attack dashboard.

To sum up the solution design, by implementing the 3 main visualization feature groups, the 5 criteria are satisfied, along with extra features like anomaly detection to enhance intrusion visualization. In the next section, the solution is tested against real and simulated intrusion to evaluate its ability.
5. Evaluation

In this section, the completed work is first explained, then moving on to the solution evaluation result, which includes the evaluation method, the project solution performance, and other findings from the evaluation process. Then, discussion on the summary of findings is presented. Finally, the difficulties encountered and future work are discussed.

5.1 Completed Work

The project has successfully created an NSM solution that can effectively capture and analyse network traffic, visualize intrusions found in traffic, and also provide an easy deployment method. These accomplishments are detailed in the sections below.

The summary of the project’s progress is as follows:

<table>
<thead>
<tr>
<th>Completed On</th>
<th>Description</th>
<th>Deliverable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 3</td>
<td>Project Proposal and Website</td>
<td>Project Plan and Website</td>
</tr>
<tr>
<td>Oct 23</td>
<td>Studying setup</td>
<td></td>
</tr>
<tr>
<td>Dec 31</td>
<td>Study on Existing Solutions</td>
<td>Evaluation of the solutions’ features and implementation;</td>
</tr>
<tr>
<td>Jan 5</td>
<td>Development setup</td>
<td></td>
</tr>
<tr>
<td>Jan 15</td>
<td>Data Integration and storage in development setup</td>
<td>Docker configs for deploying the development setup</td>
</tr>
<tr>
<td>Jan 23</td>
<td>Interim Report</td>
<td>Interim Report</td>
</tr>
<tr>
<td>Mar</td>
<td>Dashboard design &amp; implementation</td>
<td>Completed Dashboard</td>
</tr>
<tr>
<td>Apr</td>
<td>Dashboard Evaluation</td>
<td>Evaluation Result with simulated/real-world intrusions</td>
</tr>
<tr>
<td>Apr 18</td>
<td>Final Report</td>
<td>Final Report</td>
</tr>
</tbody>
</table>

*Table 3 Project Progress*

5.2 Solution Evaluation Result

As mentioned, both simulated and real-world intrusions are needed for evaluating the result. Thus, simulated attacks were launched to the VM fyp, and real-world traffic are captured for around 18 days, which generated ~9 million log entries. The evaluation results are divided
into 3 parts. First, result with probe attacks, then privilege escalation attacks, and finally DoS attacks.

5.2.1 Probe Attacks
For probe attacks, simulated attacks generated by Nmap and ZAP are successfully detected. Several real intrusions are also discovered. Note that anomaly detection is only tested with real traffic, as with simulated attacks, they always stand out due to the low amount of legitimate traffic.

5.2.1.1 Simulated Attack - Nmap
A TCP SYN scan on port 1-1000 is launched, with the command

\[
\text{nmap} \ -sS \ \text{fyp.lan} \ -p \ 1-1000
\]

And it is detected by Suricata

![Figure 18 Nmap scan detected](image)

5.2.1.2 Simulated Attack - Zap
Zap is used for its automated web spider function and vulnerability detection scan, where both are performed on the web application Mutillidae in Metasploitable 2. The scan successfully mapped the links in Mutillidae, and also discovers vulnerabilities, like an exposed password file in the web application. For the solution, it is also able to identify successful access to the password file, and the high rate of HTTP 404 error which is a sign of web spider scan (see Figure 19).
5.2.1.3 Real World Probe Attacks

For the real traffic, the solution shows that there are constant probe attacks throughout the 18 days, for which most of the attacks are scanning port 5060, a port for SIP/VoIP services, with most attacks originating from a single IP in Germany (see Figure 20). By going to the SIP dashboard and filtering the IP, the solution also shows that all SIP requests from the IP has “friendly-scanner” as user agent, which shows that it is a scan with the SIPVicious tool.

On the other hand, by using the probe attack anomaly detection, the solution also uncovered many big port scans, which are visualized as a spike for short scan or a bump for longer scan. By observing the port heatmap, the range of ports scanned can also be identified. Another
point to note is that the alert count is relatively constant despite the scans happening, which shows that anomaly detection can indeed help user identity probe attacks that are otherwise missed by the IDS (see Figure 21).

![Figure 21 Port Scan found by anomaly detection. Note the highlighted spike and bump, which also coincide with increased connections to port 1000-10000 and port 35000-50000, as well as increased ping requests](image)

### 5.2.2 Privilege Escalation Attacks

For privilege escalation attacks, Sqlmap is used to carry out SQL injection attacks

#### 5.2.2.1 Simulated Attack - Sqlmap

For the attack, Sqlmap is launched to conduct SQL injection and retrieve the database schema via the query parameters in the url:

```text
```

After scanning and attacking, Sqlmap successfully identified an injectable parameter – `username`, and is able to retrieve a list of databases. One database “owasp10” is selected and the schema is retrieved. For the solution, this attack is also successfully (see Figure 22).
Figure 22 Sqlmap attack: Sqlmap identified the injectable parameter (highlighted in yellow) and successfully retrieved schema of database owasp10 (highlighted in red). The solution also successfully identifies both the scan and the schema access.

5.2.2.2 Real World Privilege Escalation Attacks

For attacks in real traffic, the solution identified multiple privilege escalation attacks throughout the 18 days, for which most of the attacks are targeting port 80. These attacks mostly aim for command execution or code injection to the victim via exploits. Examples include using known exploits in D-Link router and JAWS web server. The solution also shows that most of these attacks come from USA, Russia and China (see Figure 23).

Figure 23 Privilege escalation attacks found. The blue IP address text are clickable links. The red highlighted alerts are router related. One can also identify the major source of these attacks via the map.

When reviewing data in the solution, users can also gain more information on an IP address by clicking on it, as mentioned in the design section. Using the IP with the most number of alerts as an example, it is regarded as “Untrusted” (see Figure 24), which matches the observation in the solution.
5.2.3 DoS Attacks

For DoS attacks, Slowhttptest is used to simulate slow DoS attacks. As for real attacks, no DoS attack is observed. Since anomaly detection is easy in the simulated environment, and it is better to have some baseline traffic, the test instead combines simulated attack and real traffic. It is done by exposing an Apache server to the internet via port forward and attacking it from a remote host with Slowhttptest. This method allows the solution’s ability to be tested more realistically.

5.2.3.1 DoS Attack – Slowhttptest

2 types of slow DoS attacks are used: Slowloris and Apache range header, with an Apache 2.2.8 web server as the victim. The executed commands are as follows:

```
Slowloris: slowhttptest -u http://ip:port -c 500

```

Where -c specifies the total number of connections to use in the attack.

The result is that the server is brought down by Slowloris, but not Apache range header attack.

For the solution, only the apache range header attack triggered alerts, while Slowloris is only detected via anomaly detection on connection duration. This again shows that by combining both IDS and anomaly detection, the solution is more effective in detecting intrusions (see Figure 25).
5.2.3.2 Note on DoS Anomaly in Real Traffic

Although no DoS attack is detected in the real traffic, multiple spikes in the number of inbound connections and unique source IPs are detected with the DoS anomaly detection dashboard, which are not considered as DoS attacks due to the short duration. Specifically, most of these spikes connect to port 80 and 443 from all over the world (see Figure 26). Upon inspecting HTTP metadata logs during these spikes, the solution shows that Zeek is unable to extract url and path information from them, and only knows they got an HTTP 400 error from the server on port 80. As for the connections to 443, they are encrypted, meaning not much useful information is available. Thus, one can only conclude that they are probably probe attacks on web servers, but we cannot identify the exact type.
Thus, although the solution can detect many types of intrusion with Suricata and anomaly visualization, if both Suricata and Zeek fail to provide details of the connection, it is difficult to determine the exact nature of the attack.

5.2.4 Easy Deployment

Moving on to the difficulty in deploying the solution, although the project solution is composed of 5 different applications, deploying it is easy. To start the solution, users are only required to create a file for config and execute a single docker-compose command. On the other hand, the solution also provides great flexibility. Users are free to choose how Elasticsearch data is stored, which can be either in a docker volume or a directory on the system. If a directory is chosen, the solution also provides options to set the user and group owner of the directory, which helps avoid the common permission problem with docker containers. Lastly, the config files of Suricata, Zeek, Filebeat and Kibana are directly mounted in the containers, so to modify their configuration, users only have to restart them instead of rebuilding them. Hence, the solution has achieved easy deployment, while still maintaining flexibility.
5.3 Discussion on Result

To sum up the evaluation result, the project solution has demonstrated its high effectiveness in visualizing both simulated and real-world intrusions. In particular, all 3 types of attacks are detected by the solution. Moreover, the solution allows users to utilize visualization from multiple dashboards to gain information on the attack, like in the case of probe attack on SIP port and the webserver probe attack discovered with the DoS anomaly dashboard, meaning that both the Suricata alert and Zeek metadata are utilized. However, during the evaluation, the solution has shown that its effectiveness in visualizing intrusion can be severely hindered by traffic encryption.

5.3.1 Note on encryption

To show the effect of encryption, here is a statistic from the real traffic: in the ~1 million connections that have Zeek-recognizable protocols, 95.2% are SSL/TLS, which are mainly HTTPS and DNS over TLS traffic. This means that for these connections, the only available information is from the network and transport layer, in addition to unencrypted information in TLS handshake, like Server Name Indication and the certificates. Thus, Suricata and Zeek cannot inspect the application data in the connection, meaning they are not able to perform most signature-based detection or provide details of the traffic.

5.3.1.1 Possible Solutions – Proxy

One solution is to use a kind of proxy, like SSL Visibility Appliance from Symantec, which act as a middleman between client and server. Encrypted traffic is decrypted by the proxy and sent to the solution, while at the same time re-encrypted and sent to the original destination. This allows the solution to have full access to the traffic content [16]. However, such a proxy must require a CA certificate to be installed in every client in the network, which makes it unsuitable for networks where not every client is under the network admin’s control.

5.3.1.2 Possible Solutions – Client/Server Hash

Another solution is to use hash like JA3 for SSL/TLS and Hassh for SSH, which produces a hash from information like encryption and algorithm and key exchange methods. These hash methods are included in the project solution, where JA3 hashes are displayed in the SSL dashboard and used by rules in Suricata to detect malicious servers and clients, while Hassh is displayed in the SSH dashboard, and can be used for finding unusual SSH connections by making a whitelist of client/server, and adding custom visualizations that show unknown
hashes. However, one can easily see that the intrusion detection ability provided by the hashes is dwarfed by IDS with signature-based detection.

5.3.1.3 More Challenges

Moreover, encryption protocols are improving for more privacy, like the latest TLS v1.3, which has most of the handshake encrypted, so the certificates are not extractable anymore. Furthermore, a more secure extension, Encrypted Client Hello, is in the works, which provides a further encrypted handshake, in which the Server Name Indication header and encryption details and other information are also encrypted. In this case, the JA3 hash will be unusable. Thus, encryption is going to be an increasingly difficult challenge for NSM solutions.

5.4 Difficulties
5.4.1 Features locked behind expensive licenses

Security software/resources often have features locked behind an expensive license fee, and unfortunately, it has affected the project.

For example, the free version of Splunk used has a limit of 500 MB of data imported per day, so when dealing with alerts and metadata generated from 30+GB of packet capture files in the dataset, the free version is not enough, and only the Suricata alerts are used to evaluate Splunk.

Moreover, the free Splunk used (Splunk Enterprise) is for general log aggregation, while only a paid app, Splunk Enterprise Security, exists to provide functions specialized for SIEM. Hence, a free app, Stamus Networks App, is used for the study, which does not contain many features.

Finally, the free ET Open NIDS ruleset may have also caused the low number of generated alerts in UNSW-NB15, which only contains ~25000 rules. If it is possible to use the paid ET Pro ruleset, which contains ~72000 rules, Suricata may be able to detect more intrusions in the dataset.
5.5 Future Work

With limited time, the project must omit some aspects to finish its main objectives in time. Looking forward, there are 3 goals that the project can continue to work on.

5.5.1 Data Archival and Retention

As an NSM solution is likely used for a long time, the size of captured data will cause problems sooner or later. Thus, some customizable log rotation and data retention policies can be implemented, so users can adjust them to their needs, and not worry about running out of space.

5.5.2 Alert Feature

At the project solution’s current state, users must actively observe the dashboards to notice intrusion, which means proper response to intrusions cannot be done until someone look at the dashboards. Thus, an alerting feature that notifies users when a high severity alert is generated, or an anomaly is detected, would enable users to take rapid response to intrusions.

5.5.3 Testing on Network

For all the evaluations done, the solution is only tested in the situation where a single host is the protected target. This is because internal networks are protected by a firewall, which means intrusions are much rarer and will make testing real-world intrusion difficult. Thus, the solution can be tested by monitoring a whole network’s traffic, which will likely require additional visualization to be designed.

6. Conclusion

This project aims to contribute to the effort of visualizing NIDS logs. By studying existing solutions and building an NSM solution that improves on existing solutions, the project hopes to provide additional insight on how visualization can be improved, and thus allow better efficiency in analysing and responding to network intrusions. The study, design, and implementation of the solution is successfully completed, where the solution has demonstrated its high effectiveness in visualizing real and simulated intrusion of multiple types and hence has significantly enhanced the user’s ability to analyse intrusions.
References


Appendices

1. Details on UNSW-NB15 and IDS2018

<table>
<thead>
<tr>
<th></th>
<th>UNSW-NB15</th>
<th>IDS2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attack types</td>
<td>9 (Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms)</td>
<td>7 (Brute-force, Heartbleed, Botnet, DoS, DDoS, Web-attacks, infiltration)</td>
</tr>
<tr>
<td>Attack Period</td>
<td>2 days</td>
<td>3 x 3 days</td>
</tr>
</tbody>
</table>
2. Enabled/Disabled NIDS Rules

For details of each category, refer to
https://tools.emergingthreats.net/docs/ETPro%20Rule%20Categories.pdf

<table>
<thead>
<tr>
<th>Enabled Categories</th>
<th>Disabled Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>activex</td>
<td>Malicious/compromised/ IP list:</td>
</tr>
<tr>
<td>attack_response</td>
<td>3coresec</td>
</tr>
<tr>
<td>current_events</td>
<td>botcc.portgrouped</td>
</tr>
<tr>
<td>dns</td>
<td>botcc</td>
</tr>
<tr>
<td>dos</td>
<td>ciarmy</td>
</tr>
<tr>
<td>exploit</td>
<td>compromised</td>
</tr>
<tr>
<td>exploit_kit</td>
<td>drop</td>
</tr>
<tr>
<td>ftp</td>
<td>dshield</td>
</tr>
<tr>
<td>icmp</td>
<td>threatview_CS_c2</td>
</tr>
<tr>
<td>info</td>
<td></td>
</tr>
<tr>
<td>ja3</td>
<td></td>
</tr>
<tr>
<td>malware</td>
<td></td>
</tr>
<tr>
<td>misc</td>
<td></td>
</tr>
<tr>
<td>netbios</td>
<td></td>
</tr>
<tr>
<td>rpc</td>
<td></td>
</tr>
<tr>
<td>scan</td>
<td></td>
</tr>
<tr>
<td>shellcode</td>
<td></td>
</tr>
<tr>
<td>sql</td>
<td></td>
</tr>
<tr>
<td>telnet</td>
<td></td>
</tr>
<tr>
<td>user_agents</td>
<td></td>
</tr>
<tr>
<td>web_client</td>
<td></td>
</tr>
<tr>
<td>web_server</td>
<td></td>
</tr>
<tr>
<td>worm</td>
<td></td>
</tr>
</tbody>
</table>

|                 | **Verbose/info/unused protocols:** |
|                 | activex                        |
|                 | adware_pup                     |
|                 | chat                           |
|                 | coinminer                     |
|                 | games                          |
|                 | hunting #recommended for researching potential threats |
|                 | icmp_info                      |
|                 | imap                           |
|                 | inappropriate                  |
|                 | mobile_malware                 |
|                 | p2p                            |
|                 | phishing                      |
|                 | policy                         |
|                 | pop3                           |
|                 | scada                          |
|                 | smtp                           |
|                 | smp                            |
|                 | tftp                           |
|                 | tor                            |
|                 | voip                           |
|                 | web_specific_apps              |
# 3. Omitted Zeek Metadata Log Types

<table>
<thead>
<tr>
<th>Types</th>
<th>Omit Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>dce_rpc, dhp3, dpd, ftp, irc, kerberos,</td>
<td>Low/zero logs generated in real/simulated traffic, or type is not enabled in Zeek</td>
</tr>
<tr>
<td>modbus, mysql, ntlsv, ocsp, pe, radius, rdp,</td>
<td></td>
</tr>
<tr>
<td>rfb, smb_cmd, smb_files, smb_mapping,</td>
<td></td>
</tr>
<tr>
<td>snmp, socks, traceroute, tunnel</td>
<td></td>
</tr>
<tr>
<td>intel, signature, weird</td>
<td>Zeek’s signature/anomaly detection is disabled, as Suricata is used</td>
</tr>
<tr>
<td>capture_loss, stats</td>
<td>Zeek’s performance statistic</td>
</tr>
</tbody>
</table>