Doctoral Dissertation   Academic Year 2022

Malicious Traffic Detection In Encrypted Network

Andrey Ferriyan

Graduate School of Media and Governance

Keio University

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

ABSTRACT

Traditional Network Intrusion Detection Systems (NIDS) practically cannot deal with encrypted data, hence malicious applications, which increasingly use TLS, can freely flow undetected. Therefore, a network intrusion detection system breakthrough is required to prevent and mitigate the risk of malicious applications.

This dissertation aims to detect malicious traffic in an encrypted network that utilizes SSL/TLS traffic. The challenges in detecting malicious applications are a lack of publicly open datasets, NIDS's incapability to detect attacks, and drawbacks of compliance. The dissertation focuses on methods for the detection of the malicious application, the procedures on how to build a new dataset, and determining the compliance of cybersecurity. Among the factors preventing evaluation and comparison are a lack of proper documentation, a lack of comparison technology, a lack of crucial features such as ground-truth labels, and a publicly available, and real-world environment. Two requirements are needed to overcome this. The content requirements focus on the produced dataset, such as complete capture of the traffic, payload, anonymity, ground-truth, up-to-date, labeled dataset, and encryption information. The process requirements focus on how the dataset is built. These requirements produced the HIKARI-2021 dataset and enables future dataset development, which assists security researchers in evaluating network intrusion detection systems.

Existing approaches such as Deep Packet Inspection require traffic decryption and hence potentially breach privacy. Furthermore, the key finding was that the malicious application tends to use weak encryption, offers fewer extensions, and has a specific pattern that differs from the others. The method called TLS2Vec analyzed TLS handshake and the payloads which can be used to take immediate action before the conversation is finished. The evaluation revealed that the detection performance using only TLS handshake information to distinguish between Malicious and Benign with two public datasets reached 99%. The average detection rate was 82% of the total multiclass target. In order to have a comprehensive view in terms of preventing malicious applications, measuring cybersecurity compliance is needed.

The evaluation was carried out to identify the potential risk of the data center using COBIT and vulnerability assessment to measure the current condition of the data center. The evaluation focuses on the monitor the infrastructure for security-related events.

Keywords: Privacy Preserving IDS; TLS; Encrypted Malicious Traffic, network intrusion detection systems; network intrusion datasets; encrypted network traffic; https;

Thesis Committee:

Advisor: Prof. Keiji Takeda, Keio University

Co-Advisor:

(1) Prof. Osamu Nakamura, Keio University

(2) Prof. Rodney D. Van Meter, Keio University

(3) Prof. Jun Murai, Keio University

(4) Project Assoc. Prof. Achmad Husni Thamrin, Keio University

ACKNOWLEDGEMENTS

In the name of Allah, the Most Gracious and the Most Merciful.

I would like to thank all the people who support and encourage my research. I would like to express my deepest appreciation to my Thesis Advisor Professor Keiji Takeda for his guidance, understanding, patience, and encouragement to finish this thesis. I am deeply indebted to Project Associate Professor Achmad Husni Thamrin for his professional advice, guidance, and encouragement. I would also like to extend my deepest gratitude to Professor Rodney D. Van Meter and Professor Jin Mitsugi for their supervision. My work would not be completed without their professional comments. I am extremely grateful to Professor Osamu Nakamura for his guidance and encouragement and also Professor Jun Murai for his guidance, whose words gave me inspiration for my research.

I would also like to thank Professor Hiroyuki Kusumoto, Professor Keio Okawa, Professor Keisuke Uehara, Project Professor Shigeya Suzuki, and Project Associate Professor Masaaki Sato for their kind comments. I also thank every member of the KUMO and RGROOT group in the laboratory. I thank Dr. Shota Nagayama for his help, and resources, Dr. Nitish Rajoria for his friendly and great discussion, Shota Kawamoto, for friendship and advice related to the Japanese language, Korry Luke, for the warm and funny discussion, Ryosuke Abe for his seriousness and a great example for others, Yasunobu Toyota for his help related to network and server, Yuka Shori-Kataoka and her daughters for a nice discussion about Mie Prefecture and Sport Climbing. I also greatly acknowledge the secretary of the Murai Laboratory, Yasue Watanabe, and Yumiko Usami, who have helped me with the scheduling and the paper. I also thank Visiting Assistant Professor Ahmad Almansour and his family for their friendship and for letting me introduce and explain how Islam spread in Indonesia. I thank Sofyan Wijaya and Mujiantok from ATSOFT Teknologi for great friendship and for everything I have been through since 2006.

Finally, my deepest gratitude goes to all of my family members. It would not be possible to finish this thesis without their support. I would like to thank my dearest Mother and Father, my brothers Henry and Bobby, my sister Frida, my beloved wife Shofi, and my children Ilyaqofa.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Network traffic nowadays is mainly encrypted for communication security and privacy. The existing security applications, such as traditional Network Intrusion Detection Systems (NIDS), cannot handle cyber-attacks associated with encrypted traffic. There is a lack of methods to deal with encryption traffic, which malicious applications use to avoid detection from NIDS systems. Therefore, NIDS that can handle encrypted traffic is crucial for the advanced protection of networks.

1.1 PROBLEM STATEMENT

A NIDS is a significant part of the toolkit that use organizations to monitor their network perimeter. Although it has been widely studied for more than 30 years, this research area remains a hot topic. The development of NIDS becomes more complex as researchers realize that existing systems and architectures are not good enough. The systems and architecture will always change along with the new threats and new attack vectors.

1.1.1 *Irrelevant and Obsolete Dataset*

A NIDS always deals with massive data that has features that may be irrelevant or redundant. A NIDS has high computational demands that may affect the quality of training and accuracy. To address the irrelevant or redundant data, selecting subsets of data by finding the optimal features is needed [116]. In several cases, using the optimal set of features might reduce training time at the cost of accuracy. An intrusion dataset is multidimensional [112] and includes fields such as timestamp, protocol type, packet size, communication duration between the server and the client, and packet loss ratio, etc., which are very complex.

1.1.2 *Lack of Publicly Available Datasets*

Estimating how much malicious detection methods have improved in the NIDS field has been challenging. The training of

NIDSs that employ machine learning depends on the available datasets, but obtaining a reliable dataset for comparison is difficult. Among the factors that make it difficult to compare datasets are a lack of proper documentation of the methods [137], a lack of comparison methodology [11], and a lack of important features, such as ground-truth labels, and publicly available and real-world environment traffic. The report [126] showed that from 2010 to 2015, 125 published papers performed NIDS evaluation using the KDD99 [136] dataset which is obsolete and irrelevant with the current situation.

The main problem is that the dataset is not up-to-date with the current situation and attack vectors. Furthermore, the recent network traffic is mainly being encrypted for communication security and privacy, and only very few datasets reflect this situation. Most of the existing research that employs encrypted traffic explains more on different scopes, such as traffic classification and analysis [143]. Although there exist a lot of such research [32], the dataset is not publicly available, due to its sensitivity to the data. Therefore, requirements for making a dataset which can be produced practically to serve as the basis to build the comprehensive model for the detection of new attacks are needed.

1.1.3 *Low Reliability and Privacy*

The major shortcoming of NIDS is that its detection and performance cannot easily adapt to changing attacks and environments. Most of the Internet traffic is encrypted, as can be seen from the number of active certificates issued by Let's Encrypt [76], Firefox Telemetry [44], and Google transparency report [50]. In 2020, VMWare [138] had estimated that 70% of 15 attacks would leverage encryption and were predicted to increase in the following year. Along with this change, malware authors have started to leverage TLS to hide their malicious activities. They share their information by receiving, sending instructions, and retrieving sensitive data from the infected machines.

The encrypted traffic analysis has several approaches such as, port-based, payload-based, and statistical-based. Port-based approach deduces application types by assuming that most of the applications use standard TCP or UDP port numbers, however, this approach has the drawback which loses its effectiveness in the presence of proprietary applications or protocols that use the non-standard port number, port tunneling, etc. Subsequently, the payload-based approach cannot deal 100% with encrypted traffic due to the content being opaque and requires additional methods such as Deep Packet Inspection (DPI) to inspect. However, DPI has a high computational overhead and is not privacy-friendly. Statistical-based approaches might handle the issues, whereas port-based and payload-based cannot. However, they still have some problems, such as time constraints, the obsolete dataset, etc.

1.2 DISSERTATION CONTRIBUTIONS

This dissertation contributes to several key areas of Network Intrusion Detection Systems. This section briefly summarizes the various contributions of this dissertation as pinpointed in the followings:

(1) To find the optimal set of features, I present a Genetic Algorithm based on optimized feature selections for network intrusion detection systems. I used a one-point crossover for the Genetic Algorithm (GA) parameters instead of the two-point crossover used in the previous research.

(2) I proposed new requirements for creating new datasets. Furthermore, I created a new NIDS dataset that covers the network traffic with encrypted traces. The new NIDS dataset I call HIKARI-2021 was a mix of real-traffic and synthetic attack traffic. This dataset contains labeled traffic data, such as brute force login and probing, with more than 80 features.

(3) I proposed a privacy-preserving method that used Deep Learning techniques which I call TLS2Vec. The goal was to detect encrypted malicious traffic that utilized SSL/TLS communication from the TLS handshake and the payloads without waiting for the conversation to finish.

(4) I proposed a method to assess the compliance of the data center at Sleman Regency using the COBIT framework incorporated with a vulnerability assessment. The objective was to assess the data center in terms of the process of securing the environment and the monitoring applications such as NIDS. Therefore, the organization might recognize the potential risk and how to handle incidents in the future.

1.3 SOFTWARE AND DATASET RELEASED

I have released the tools on how to produce a synthetic dataset. The tools are available to download on https://github.com/andreysfc/generating-encrypted-network. Meanwhile, the HIKARI-2021 datasets were available at https://zenodo.org/record/5199540. Both the tools and the HIKARI-2021 datasets were contributions from section (1).

1.4 DISSERTATION STRUCTURE

This dissertation consists of seven chapters. The first chapter is the overview of this dissertation. Chapter 2 is a background and brief summary of Non-Encrypted and Encrypted NIDS and the evolution of NIDS to the current state. Chapter 3 describes a method to improve classification of NIDS. Chapter 4 demonstrates the process on how to make a new dataset based on the encrypted network traffic. Chapter 5 shows how to detect encrypted malicious application running in the network. Chapter 6 describes a measurement on how to evaluate of the NIDS using COBIT, and chapter 7 is the conclusion.

2 BACKGROUND

This chapter introduces the description of IDS classification, the detection type, and then the current state of the art of the NIDS in the encrypted network. This chapter summarizes the background knowledge of various topics that will be discussed further in the rest of the dissertation.

2.1 ATTACK LIFECYCLE

There are several attack lifecycle frameworks introduced by researchers such as Threat Genomic [39] with ten components of a sequence defined as Base Type of Action: reconnaissance, commencement, entry, foothold, lateral movement, acquiring control, acquiring the target, implementation/execution, concealing and maintaining, and withdrawal. Mandiant Targeted Attack Lifecycle [88] has eight sequences: initial reconnaissance, initial compromise, establishing a foothold, escalating privileges, internal reconnaissance, moving laterally, maintaining a presence, and completing a mission. Lockheed Martin: Kill Chain Phases [58] consists of seven stages: reconnaissance, weaponization, delivery, exploitation, installation, command, and control (C2), and actions on objectives. National Cyber Security Centre [95] adopted and simplified the Lockheed Martin Kill Chain phase with four main stages: Survey, Delivery, Breach, and Affect. Referring to [97], the intrusion activity must include several processes such as reconnaissance, penetration, attack, and exploitation.

However, the framework I offered is slightly different. In my definition, a network attack is a part of a multi-stage attack that starts from Reconnaissance, Attack, Exploit, Breach, and Control. The significant point of my framework is how to deliver the attack and maintain the application vulnerability. My framework summarizes all the above frameworks and simplifies the process. Figure 1 illustrates the stages of attack in a subsequent order.

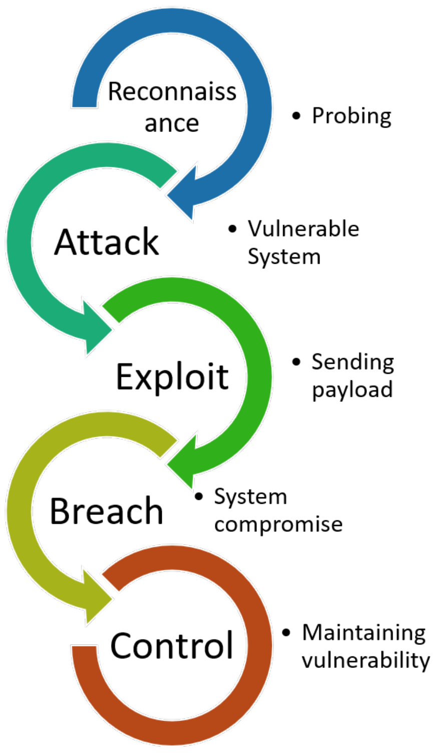

Figure 1: Network Attack Lifecycle.

In reconnaissance, the intruder probes and finds which port is open and the possibility of the vulnerable application. Probing will contain stealth probes like stealth scan by transmitting SYN packet over the target server or transmitting FIN packet [16], without encrypted traffic. Closed ports response from target server means that probing fail to discover open ports or vulnerable software. If the target server responds, the subsequent step is checking the vulnerable application. This process is enclosed with encryption if running on an encryption protocol (e.g. HTTPS). In the third stage, the intruder transmits the exploit containing a payload to the target server. If the traffic flows on the encrypted network, no traditional NIDS or Network Intrusion Prevention System (NIPS) can detect this attack. The firewall cannot block it because these look like legitimate traffic (protocol mimicry). In the fourth stage, the target is breached if the intruder successfully executes malicious code and escalates their privilege through credential harvesting. In the last part, the intruder implants a persistent backdoor or downloading additional utilities to maintain the access.

2.2 NETWORK INTRUSION DETECTION SYSTEMS

The intrusion activity is increasing along with the growth of the network. Internet Security Threat Report [1] shows that in the last 8 years, more than 7.1 billion identities have been exposed in data breaches. It is unavoidable that besides implementing security in software and designing hardware, IT systems must continuously monitor the environment to ensure well function, noticing any sign of intrusion, attack, or anomalies. NIDS becomes an additional line of defense to protect the critical network infrastructure and its systems. Thus, the NIDS and Intrusion Prevention System (IPS) is crucial in network security. A NIDS monitors a host or a network and analyzes any signs of intrusion, malicious behavior, or anomalies. Its goal to detect any potential attempt to bypass the security mechanism, which will compromise the confidentiality, integrity, and availability of information. The NIDS will generate alerts from detected malicious host or network. Figure 2 depicts a generic deployment of NIDS.

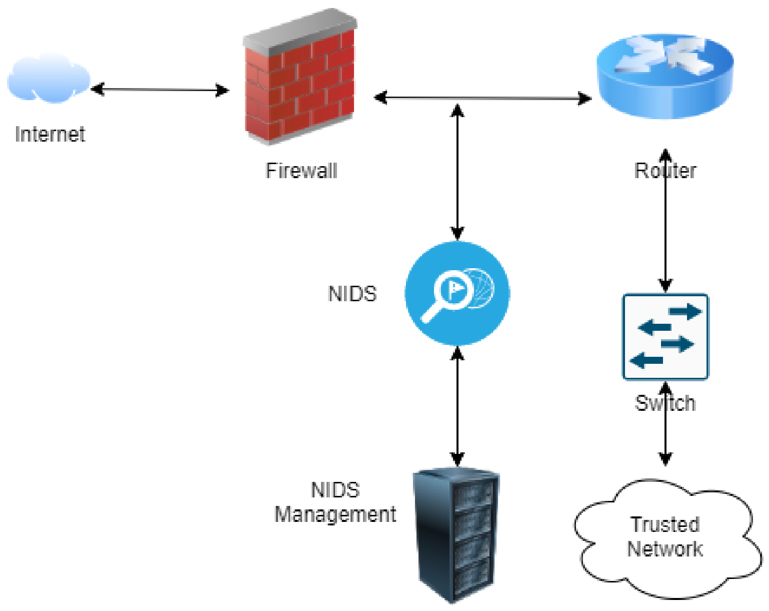

Figure 2: The generic deployment of NIDS.

The following section explains the most common classification of NIDS and its definitions.

2.2.1 *Deployment Position*

In regard to the deployment, NIDS can be divided into host-based and network-based. Host-based detects intrusion on a host machine by analyzing events and behavior of users in a single computer (single host). The deployment requires a sensor to audit the information and responds to occurring events. The sensor allows for collecting detailed information. However, it requires the deployment of sensors on every host. In contrast with the network-based, it might protect partial or the entire network. In the recent works of NIDSs, researchers introduce several approaches, such as wireless-based [17], hybrid-based [67, 79], IoT-based [151], SDN-based [134], and even cloud-based approaches [24].

2.2.2 *Detection Based Approach*

NIDSs can be categorized further based on their detection such as signature-based (misuse-based) and anomaly-based. Signature-based detection monitors known attacks and detect their occurrence in the network. The main advantage of this approach is the minimum false alarm, whereas the main drawback is its unsuitability for identifying novel attack behavior and unknown intrusions. Such detection is due to their predetermined rules in the system. However, this NIDSs detection are the most common for monitoring the network due to its high precision in detection. Examples of signature-based NIDSs include Suricata [4] and Snort [111].

Anomaly-based detection monitors and analyzes any deviations from normal activity. The system initially learns the normal state from the activity and afterward defines the deviations as an intrusion [23]. The anomaly-based detection does not require any signature or rule for detection. The main advantage of this approach is the system can detect unknown attacks. However, this mechanism is costly and comes up with high false positives. The effect is due to the difficulty in finding the boundary between normal and abnormal activity for intrusion. Normal and abnormal activity can be trained using machine learning approaches. NIDSs generate responses based on the intrusion symptoms. There are two types of response: active and passive. Traditional NIDSs are passive monitoring. The passive response goal is to inform any intrusion and take further action by notifying the system administrator about the intrusion. The passive response needs human involvement to respond to the intrusion. Human involvement might damage the system due to the time gap between notifying and action. The active response produces an automated action to reduce the intrusion effects without human involvement [90].

2.3 THE EVOLUTION OF INTRUSION DETECTION SYSTEMS

The two published reports by Anderson [99] and Denning [33] were the start of researches on Intrusion Detection Systems (IDSs). Anderson discussed ways to improve security auditing using audit trails. His work described classified internal hackers into three different groups: legitimate, masquerader, and clandestine users. Denning's research became an inspiration for future researchers. The prototype called Intrusion Detection Expert System (IDES) uses a rule-based expert system to detect known malicious activities. Table 1 describes the history from the beginning of the IDS.

Table 1: The evolution of IDS since 1980s

Year	Type of Attacks	Method
1980-1989	User behavior, masquerading, penetration, leakage.	Anomaly, single host
1990-1999	User behavior, worm.	Anomaly, network-based, distributed, AI
2000-2009	User activity, Probing, Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R).	Host, Anomaly, Machine Learning (ML)
2010-2022	SQL Injection, XSS, Fuzzers, Analysis, Backdoors, DoS, Exploits, Shell-code, Sinkhole, Hello Flood, Wormhole.	Host, Anomaly, ML, Deep Learning (DL)

2.3.1 *Taxonomy and State of the Art*

The detection of encrypted malicious traffic has gained attention in the network security fields. While many research topics are related to traffic classification with different techniques, only a few studies focus on encrypted malicious traffic. To present the current state of research on encrypted traffic, I introduce a taxonomy that provides details of use cases, traffic category, and techniques. Figure 3 depicts a comprehensive manner in an encrypted traffic research to distinguish my research from the others.

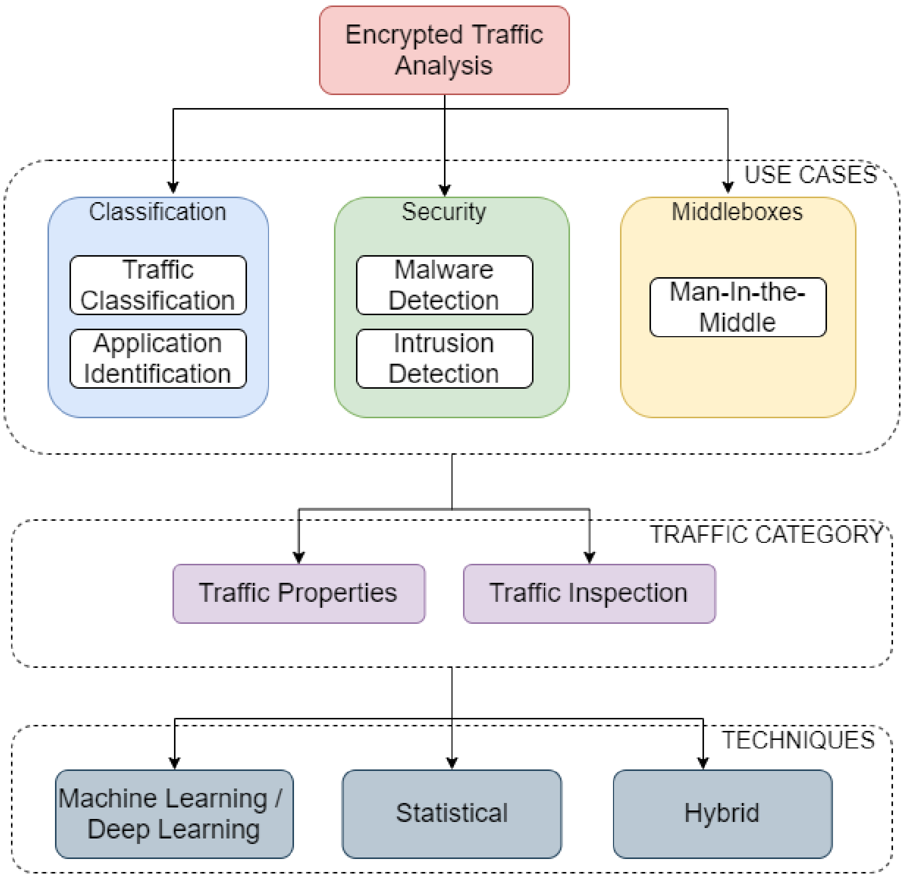

Figure 3: A taxonomy of encrypted network traffic is categorized by use cases, traffic category, and techniques.

In 2015, Velan et al. [143] proposed multi-level taxonomy for the encrypted traffic methods. Their survey differentiates traffic classification based on (i) the input (e.g., traffic properties, traffic payload, and hybrid), (ii) the techniques (e.g., payload inspection, graphical, statistical, machine learning, and hybrid), (iii) the output, and (iv) the dataset used. The signature-based approach proposed by [40] relies on Snort [128] to detect encrypted malicious traffic. While Snort is capable of a Deep Packet Inspection (DPI), the pattern matching rules are inadequate to handle malicious traffic, especially the encrypted. Papadogiannaki et al. [100] proposed a fast signature-based NIDS, detecting malicious traffic even if it is encrypted. While the detection in signature-based is swift for known attacks, its rules must be updated regularly to keep up with all available attack signatures.

Callegati et al. and Sen et al. proposed man-in-the-middle (MITM) and DPI [20, 119]. MITM and DPI must decrypt traffic using a key for inspections, hence they are expensive and not applicable without the key and potentially breach privacy. That is why traffic analysis is preferred [9], but analysis without decryption has its challenges due to conversations over encrypted channels.

Wala and Cotton [144] studied traffic analysis with fingerprinting by aggregating several pieces of information from Zeek-IDS [152], such as operating system and the version, open ports on a host, and types of browsers. Prasse et al. [103] proposed how to detect malware traffic on client computers based on HTTPS traffic analysis based on statistical patterns in the timing and the sequence of the network flows from and to the client.

Anderson and McGrew [10] analyzed the network flow with a feature set involving domain experts in supervised learning. The analysis was based on millions of TLS encrypted sessions from a commercial malware sandbox for more than one year. Shekhawat et al. [123] proposed a method to detect malicious traffic by performing feature analysis on several logs generated from Zeek-IDS. This analysis determined the relative importance of these features from three of the logs. Zheng et al. [157] proposed two-layer detection to identify malicious TLS flow. Their method used two different layers. In the first layer, a set of TLS handshake features was employed and trained to distinguish between benign and malicious TLS. The second layer was then used to identify malware families using TLS handshake features and statistical features. The authors claimed that their method could reach 99.45% of accuracy. Dai et al. [29] proposed malicious traffic of detection based approach on multi-view features. They extracted the features from multiple views such as flow statistics, SSL handshake field, and certificate key. While their methods are comprehensive, using flow statistics requires communication to finish between client and server. At the same time, it will affect the delay of the action because it requires some-times to wait for the session to finish communicating between parties.

Several works have been carried out in unsupervised learning to detect anomalies within network flow. The work of [8] proposed an unsupervised NIDS to detect unknown network attacks in encrypted networks. Su et al. [132] implemented a hierarchical clustering algorithm to extract huge training data. At the same time, Li et al. [78] proposed a clustering algorithm with a combination of K-Nearest Neighbors (KNN) for detecting anomalies. The advantage of unsupervised learning is to detect unknown malicious traffic within an encrypted network. However, without adequate information to determine the ground truth, there is a possibility that a high false alarm may occur. Furthermore, the algorithm might take time to analyze any possibilities.

2.3.2 *Machine Learning Methods*

Many survey papers provide implementation details on the NIDS. In NIDS incorporated with machine learning, feature extraction and feature selection play an essential part to increase the detection of the attack. Feature extraction transforms the data into a new feature space while feature selection tries to find the subset of the original features. Applying proper techniques in both might significantly improve classification and processing times. In some cases of feature selection, the accuracy might increase by removing noise in the dataset. However, researchers need to carefully implement it, especially between studies on datasets versus real-world implementation. Detailed implementation of feature selection is provided in chapter 3.

Zaman et al. [150] proposed feature selection in NIDS using three different types of evolution algorithms: Genetic Algorithm (GA), PSO (Particle Swarm Optimization), and Differential Evolution (DE). They compared the performance of these algorithms and validated them with Neural Network and Support Vector Machine using KDD Cup 99 dataset. From 41 features in the dataset, the optimal features selected by GA, PSO, and DE were 16, 15, and 13, respectively. They found that DE was the best with 99.75% classification accuracy with 1.62 seconds in training time.

Ahmad et al. [2] proposed an NIDS using a feature selection with Genetic Principal Components (GPC) by selecting a subset of features with optimal sensitivity and the highest discriminatory power. They implemented SVM with Radial Basis Function Kernel (RBF) to evaluate the features selection method. The implementation of this method with only 10 features from the KDD Cup 99 could achieve 99.96% accuracy. Unfortunately, the training time was averagely high compared to the others.

Pervez and Farid [102] proposed a feature selection with a filter method and used SVM to classify. The goal was to reduce input features from the training data. In the training phase, they used training data and 10-fold cross-validation. Their experiment showed that with three features, the classification accuracy managed up to 91% and with 36 up to 41, the classification accuracy was 99%. However, their experiment did not mention anything about training time.

It is important to understand that many papers experiment with obsolete datasets (even recent studies). Newer attacks can be totally different from the old attacks in terms of technical implementation and their characteristics. A new attack class cannot be detected using obsolete research datasets and might not relevant for the future study. However, there is a technical challenge in terms of computation process. Applying feature selection to a dataset, which changes rapidly, might hinder the detection. Finding the optimized features takes a considerable amount of time in computing process.

2.3.3 *Natural Language Processing Based NIDS*

Natural Language Processing (NLP) is a subset of Artificial Intelligence (AI) to understand human language using the input from the texts or voice. Specifically, NLP learns and extracts features from a sample consisting of sentences or words. NLP is used nowadays in cybersecurity areas to process the unstructured data generated from security events.

Huang et al. [57] used NLP to automatically detect malicious domains based on the analysis of discussions on a technical mailing list. They designed a system named Gossip which extracted the features from email threads, participation of the users in the discussions, and content keywords. The system detects the malicious domains without directly crawling the suspect websites. The results show that Gossip produces high accuracy.

Zhuo et al. [158] used word embeddings to learn syntactically and the semantic relationships from a large amount of network log data. Their works apply to non-textual data which is treated as a string. Their proposed method demonstrates high accuracy with a small training sample.

Word embeddings are techniques that convert words or sentences into vector representations. It does this by tokenizing each word or sentence and converting it into vector space. Word embedding aims to capture the semantic meaning of words in a sequence of text. Similar words or sentences will have high score representations.

Word2Vec is one way to implement word embeddings. Developed by Tomas Mikolov, Word2Vec used the cosine similarity metric to find similarities among words or sentences. Word2Vec has two different architectures, Continuous Bag-of-Words (CBOW) and Skip-gram. Detail explanation and implementation of Word2Vec are presented in the proposed method in chapter 5.

Based on Baroni et al. [14], the implementation of Word2Vec has proven to be successful in different areas. In non-network areas, Baek and Chung [12] used Word2Vec to build a multimedia recommendation system based on trust relationships by encoding the sentiment words in related comments into word vectors. At the same time, Chuan et al. [26] adopted Word2Vec to capture the relationship between harmonic and meaningful tonal in music. Ring et al. [110] proposed a similarity measure between IP Addresses using Word2Vec, where the idea is to extract available context information from the network connection. Goodman et al. [49] proposed translated packets into vectorized representations then formed a sequence of words. In contrast, Li et al. [77] used Word2Vec to deal with the semantic gap in the anomalous HTTP traffic.

3 FEATURE SELECTION USING GENETIC ALGORITHM

This chapter presents the construction of optimal features using NSL-KDD dataset. In this chapter, I address the irrelevant or redundant feature problem by selecting the optimal features and processing offline [116] data packets to find the most optimal features. With the optimal features, I may reduce training time with the cost of accuracy. This chapter used the NSL-KDD dataset as well as its modified versions to reflect the recent attack types in order to determine and measure performance in my NIDS. NSL-KDD dataset is multi features data which has been summarized into 41 features. From several metaheuristic approaches, GA was chosen because it can approximate solution from high dimensional data and solve complex optimization problem [38]. During feature selection process with GA, I chose Correlation-based Feature Selection (CFS) for my fitness value because CFS does not induce into high computational resource and does not crave for any specific threshold.

3.1 DATASET MODIFICATION

I used NSL-KDD dataset for the experiments, which is the most frequently used by security researchers for testing the network intrusion detection system performances and identify between intrusion and normal connections. NSL-KDD dataset contains 41 features with 125,973 instances. Each instance represents a connection between server and client and is labeled as either normal or attack. There are 22 attacks features in total, categorized into four groups: Denial of Service (DoS), Probing (Probe), Remote to Local (R2L), and User to Root (U2R).

The advantage of using NSL-KDD dataset is that it does not contain any redundant instances in the training set. However, based on my survey findings, NSL-KDD dataset suffers from irrelevant attacks features. I looked into each attack and based it on the recent incidents of such attacks in 2017 [85] I determined 12 attacks features that are irrelevant nowadays, such as back, ftp_write, imap, land, multihop, neptune, phf, pod, satan, smurf, teardrop, and loadmodule. There were 58,630 instances in these 22 attacks, and by eliminating the 12 irrelevant attacks, I obtained 9,011 instances in the remaining 10 attacks.

With the attack categorization according to their relevance, I modified the NSL-KDD dataset and came up with three datasets. (1) Training data where all instances with irrelevant attack features were labeled as normal instances. (2) Training data containing only instances with relevant attack features. In addition, (3) training data where instances with irrelevant attack features were labeled as "other_attack". Table 2 shows the attacks and the instance counts of the original NSL-KDD dataset along with the three modified datasets.

3.2 EXPERIMENTS

After I created three different datasets, I performed feature selections to the three training datasets using GA to find the optimal features. The efficiency and the result from GA are dependent on its tuning parameters. There are several best practices of configuration parameters for GA, such as [150], [31] and [51]. I compared configuration parameters from [150] and [31] with two-point for crossover operator with the same configuration, I using one-point crossover only. Table 3 shows the difference between parameters from GA.

3.2.1 *Parameters for GA*

3.2.1.1 *Individual Representation*

Each instance in the dataset is an individual representation for GA, and modeled as a 41-bit vector. Each bit in the vector stands for a feature, where 1 means the feature exists, and vice versa. Instances were randomly selected in the initialization of the feature selection.

3.2.1.2 *Fitness Value*

I used Correlation-based Feature Selection (CFS) for the fitness value. The GA evaluates the fitness from individuals where each population is called generation. CFS [54] uses a heuristic approach to score subsets of feature by evaluating the importance of features. It evaluates the worth of a subset of feature by considering the individual predictive ability of each feature and the degree of redundancy between them. High correlation feature

Table 2: Attack features and their instance counts for the NSL-KDD dataset (ORI), normal-labeled irrelevantattack-feature dataset (ON), removed irrelevant-attack-feature dataset (RM), and the dataset where the irrelevant attacks are labeled as other attack (OA).

Attack Features	ORI	ON	RM	OA	Category Attack
normal	67343	116942	67343	67343	-
other attack	-	-	-	49599	OA
neptune	41214	MTN	Removed	OA	DoS
warezclient	890	890	890	890	R2L
ipsweep	3599	3599	3599	3599	Probe
portsweep	2931	2931	2931	2931	Probe
teardrop	892	MTN	Removed	OA	DoS
nmap	1493	1493	1493	1493	Probe
satan	3633	MTN	Removed	OA	Probe
smurf	2646	MTN	Removed	OA	DoS
pod	201	MTN	Removed	OA	DoS
back	956	MTN	Removed	OA	DoS
guess passwd	53	53	53	53	R2L
ftp write	8	MTN	Removed	OA	R2L
multihop	7	MTN	Removed	OA	R2L
rootkit	10	10	10	10	U2R
buffer overflow	30	30	30	30	U2R
imap	11	MTN	Removed	OA	R2L
warezmaster	20	20	20	20	R2L
phf	4	MTN	Removed	OA	R2L
land	18	MTN	Removed	OA	DoS
load module	9	MTN	Removed	OA	U2R
spy	2	2	2	2	R2L
perl	3	2	3	2	U2R
Total Instances	125,973	125,973	76,374	125,973	

Table 3: Difference between parameters of GA.

Parameters	Zaman	Mod-Zaman	Dejong	Mod-Dejong
Crossover type	two-point	one-point	two-point	one-point
Crossover rate	0.7	0.7	0.6	0.6
Generation	50	50	1000	1000
Population	20	20	50	50
Mutation	0.024	0.024	0.001	0.001

with the attack feature and yet uncorrelated with each other are preferred. After completing each generation, the GA considers, if an upcoming generation like the most fit individuals (based on fitness value) from the previous generation can be included. This selection method of invididuals uses tournament selection to make sure that the best fitness individuals are selected for crossover. Subsequently, the GA determines how bits is swapped among the pairs.

Using CFS for the fitness value, I performed feature selections with Mod-Zaman and Mod-Dejong on the ON, RM, and OA datasets. The Mod-Zaman parameters resulted in 17 features for ON, 21 features for RM, and 11 features for OA. Meanwhile, the Mod-Dejong parameters obtained in 16, 16, and 21 for the respective datasets. I used this configuration to apply feature selection in two different parameters of GA. Tables 4 and Tables 5 show the feature selection results for both parameters.

3.2.2 *Evaluation*

Five classification algorithms (e.g., Random Forest, Bayesian Net, Naive Bayes, K-Nearest Neighbor (KNN), and C4.5) were applied to the three different datasets. The experiments environment was conducted by means of GNU/Linux Lubuntu with 3.6 GHz Intel Core i7 with 8-cores and 32GB of RAM. I performed the evaluations in two steps. First, I applied the classifiers without GA to find the best performance in training time (TT) and classification rate (C) among the classifiers without any feature selection. Second, I applied the classifiers to the three different datasets after feature selection with Mod-Zaman parameters and Mod-Dejong parameters. To avoid overfitting, I used 10-fold cross validation and repeated the process 3 times and took a mean for classification accuracy. The results without GA are presented in Table 6 for evaluation with ORI, ON, RM, and OA using five different classifiers.

Based on Table 6, in general I see improvements in classification rate on the modified datasets, and in particular, it removes irrelevant attacks from the dataset reduces the training time by more than half without any significant decrease in accuracy. The incorrect classification using Random Forest on the ORI dataset was 0.1326%, and thus, 167 instances were falsely detected, while the ON, RM and OA datasets were 0.0937% (118), 0.1139% (87), and 0.01199% (151), respectively. The Random Forest classifier outperformed the other classifiers in terms of classification rate

(C), but K-Nearest Neighbor showed the best performance for training time (TT). Hence, I only used Random Forest classifier for the evaluations after feature selections.

Several noteworthy points can be drawn from Table 7. The first is the fact that my Mod-Zaman and Mod-Dejong configurations performed better in the classification rate compared to the original ones, while the training time gave mixed results. Meanwhile, I did not see any patterns in the number of features across the configurations. This result is in line with [25], where feature selection can improve classification accuracy but it depends on the right feature not instead of the whole or subset of features. My configurations performed better in the training time except for the Mod-Dejong on OA dataset when compared with the results without feature selections. Furthermore, while reduced classification rate was expected after feature selection, I saw significant drops of more than 0.2 point percent compared to that without feature selection, in the configurations that resulted in fewer than 17 features.

3.3 SUMMARY

I performed feature selections using GA to find the optimal features from NSL-KDD dataset. NSL-KDD dataset suffers from irrelevant attack features, where out of the 22 attacks features only 10 are still relevant now. Hence, I created three more datasets based on the NSL-KDD by removing the irrelevant attack features, converting them into the normal and into a single attack. I believe these three datasets can be employed by other researches in the future. In evaluating the feature selection, I used five classifiers againts the datasets without feature selection and I found that Random Forest gave the best results both in the training time and classification rate and it performed better on the modified datasets compared to the original NSL-KDD dataset. I then evaluated my feature selections, where I used one crossover instead of two in the previous research, using Random Forest and I found that my parameters, in general, gave better results even though they experienced classification rate drops in several cases. I believe that these warrants require further investigations in the future.

Table 4: ON, RM and OA results using Mod-Zaman parameters.

Feature	Information	ON	RM	OA
1	duration	-	✓	-
2	protocol_type	-	-	✓
3	service	-	✓	✓
4	flag	-	-	✓
5	src_bytes	✓	✓	
6	dst_bytes	✓	✓	✓
9	urgent	-	-	✓
10	hot	-	-	✓
11	num_failed_logins	✓	✓	-
12	logged_in	✓	-	✓
13	num_compromised	-	✓	✓
14	root_shell	✓	-	✓
15	su_attempted	✓	-	✓
16	num_root	-	✓	-
17	num_file_creations	✓	✓	-
20	num_outbound_cmds	✓	-	-
21	is_host_login	✓	-	-
22	is_guest_login	✓	-	-
25	serror_rate	-	✓	-
26	srv_serror_rate	-	✓	-
28	srv_rerror_rate	✓	-	-
30	diff_srv_rate	-	✓	-
31	srv_diff_host_rate	-	✓	✓
33	dst_host_srv_count	✓	✓	-
34	dst_host_same_srv_rate	-	✓	-
36	dst_host_same_src_port_rate	-	-	✓
37	dst_host_srv_diff_host_rate	✓	✓	-
38	dst_host_serror_rate	-	✓	-
40	dst_host_rerror_rate	-	✓	-

Table 5: ON, RM, and OA results using Mod-Dejong parameters.

Feature	Information	ON	RM	OA
1	duration	-	-	-
2	protocol_type	-	✓	-
3	service	-	✓	✓
4	flag	✓	-	✓
5	src_bytes	✓	✓	✓
6	dst_bytes	✓	-	✓
7	land	-	-	✓
8	wrong_fragment	-	-	✓
9	urgent	-	-	✓
10	hot	-	✓	-
11	num_failed_logins	-	✓	✓
12	logged_in	-	✓	-
13	num_compromised	-	✓	-
14	root_shell	-	-	✓
15	su_attempted	-	-	✓
16	num_root	✓	-	-
17	num_file_creations	✓	-	✓
18	num_shells	-	✓	✓
19	num_access_files	✓	✓	✓
20	num_outbound_cmds	-	✓	✓
21	is_host_login	-	-	✓
22	is_guest_login	✓	-	-
23	count	-	✓	-
25	serror_rate	✓	-	✓
26	srv_serror_rate	-	-	✓
27	rerror_rate	-	✓	✓
28	srv_rerror_rate	✓	✓	-
30	diff_srv_rate	-	-	✓
31	srv_diff_host_rate	✓	-	-
32	dst_host_count	✓	✓	-
33	dst_host_srv_count	✓	-	-
34	dst_host_same_srv_rate	✓	✓	✓
35	dst_host_diff_srv_rate	-	✓	✓
37	dst_host_srv_diff_host_rate	-	-	✓
39	dst_host_srv_serror_rate	-	-	✓
40	dst_host_rerror_rate	✓	-	-
41	dst_host_srv_rerror_rate	✓	-	-

Table 6: Evaluation from ORI, ON, RM, and OA before feature selection.

	ORI		ON		RM		OA	
Classifiers	TT (sec)	C (%)	TT (sec)	C (%)	TT (sec)	C (%)	TT (sec)	C (%)
Random Forest	26.41	99.86	25.86	99.9	18.7	99.85	26.17	99.88
Bayesian Net	2.53	97.37	2.04	95.69	0.83	97.89	2.6	95.19
Naive Bayes	0.66	48.38	0.6	72.46	0.21	58.27	0.47	80.43
KNN	0.03	99.63	0.03	99.77	0.03	99.67	0.01	99.77
C4.5	10.8	99.75	12.84	99.85	4.98	99.74	11.82	99.77

Table 7: Summary from Random Forest classifier with GA using parameters from Zaman, Dejong, and using modified parameters Mod-Zaman and Mod-Dejong with selected features (F).

	ON			RM			OA		
	TT (sec)	C (%)	F	TT (sec)	C (%)	F	TT (sec)	C (%)	F
Zaman	28.11	96.61	13	7.89	99.72	14	29.06	99.78	22
Dejong	20.75	99.07	15	15.49	97.8	22	21.3	99.19	19
Mod-Zaman	25.76	99.72	17	9.94	99.82	21	13.62	97.86	11
Mod-Dejong	24.69	99.32	16	8.58	99.17	16	29.54	99.72	21

4 GENERATING ENCRYPTED SYNTHETIC ATTACK TRAFFIC

The dataset is an important part to build machine learning-based IDS models. The process starts with capturing traffic either as a packet or flow from the internet. Afterward, the captured traffic is compiled into a specific type of data containing network-related features, including labeling. Labeling is a crucial process for the dataset. Handling ground-truth is a real challenge, especially when experts cannot determine whether the traffic is an attack or benign. This is the reason why researchers use synthetic traffic. However, this implies that the generated traffic is not representative of the real-world environment. In a nutshell, the process of making a dataset starts with capturing traffic, and ends with the final preprocessing phase. The final result from the preprocessing phase is a labeled dataset. Each data point is classified into malicious or benign. The file contains tabular data in a human-readable format, such as a CSV file, or binary form, such as an IDX file. The number of detected malicious or false alarms can be used to benchmark the dataset.

The existing datasets lack reliably encrypted traces and practicality to produce as the basis to build the comprehensive model for the detection of new attacks. Most of the existing research that employs encrypted traffic are focused on different scopes, such as traffic classification and analysis [143]. Although such research exists [32], the dataset is not publicly available, due to the sensitivity of the data.

Benchmark datasets are an important basis to evaluate and compare the quality among different IDS. Based on the detection methods, there are three types of IDS: signaturebased, anomaly-based, and a combination of signature-based and anomaly-based. These three types of IDS benchmark their systems with the KDD99 dataset, which is obsolete. The signature-based one focuses on building automatic signature generation [64], while the anomaly-based focuses on observing an outlier from the legitimate profile [3]. The signature-based type relies on a pattern-matching method to identify and attempt to match with the signatures database. When an attack attempt matches with the signature, an alert is raised. The signature-based type has the highest accuracy and lowest false alarm rate but this type cannot detect unknown attacks. While the anomaly-based type might detect unknown attacks by comparing abnormal traffic with the normal traffic, the ratio of false alarm rates remains high.

I presented a tool and requirements for making a new dataset created by generating encrypted network traffic in a real-world environment. I reviewed the existing datasets and provided the most important features from their dataset, such as the capture duration of the network traffic, the kind of attack they implemented, and the data format they used. I then described the dataset generation methodology along with the attack traffic generation and explained the characteristics of the attack traffic. Subsequently, I described the network configuration for generating network traffic, the scenarios, the tools and code I used to generate, and the capture duration of the network features. I analyzed the dataset and provided information on how the labeling worked.

4.1 REVIEW OF EXISTING DATASETS

Many researchers have published papers based on generated IDS datasets, such as ISCX [124], UNSW-NB15 [93], and UGR'16 [86]. In this section, I introduced several IDS datasets with their characteristics and highlighted several important requirements from their perspective.

4.1.1 *KDD99*

The KDD99 dataset was created in 1999, using tcpdump, and was built based on the data captured by the DARPA 98 IDS evaluation program [80]. The training data are around four gigabytes of compressed TCP data from seven weeks of network traffic. The network traffic contains attack traffic and normal traffic. The capture of the network traffic was done in a simulated environment. The dataset contains a total of 24 attack types, which fall into four main categories: Denial of Service (DOS), Remote to Local (R2L), User to Root (U2R), and probing. KDD99 has been used extensively in IDS research. The report [126] showed that during 2010–2015, 125 published works performed IDS evaluation using KDD99. While this dataset is considered inadequate for evaluation, since there are a lot of redundant instances recorded, the main problem is that the dataset is not up to date with the recent situation and attack vectors. Although many researchers were already convinced with this information, studies from another group of researchers argued that this dataset is the most widely used for benchmarking [98] or for limiting their study only for KDD99 [83].

4.1.2 *MAWILab*

MAWI was built in 2001 and consists of a set of labels locating traffic anomalies in the MAWI archive [46]. This dataset contains tcpdump packet traces captured from an operational testbed network in a link between Japan and the United States. The archive contains 15 minutes of daily traces. This dataset is huge with a very long period. The labeled MAWI archive is known as MAWILab, obtained from a graph-based methodology that combines different and independent anomaly detectors [45]. MAWI archives labeling is based on inferences that results in no ground-truth traffic that can be used for evaluation. The label has three classes: anomalous for a true anomaly, suspicious that indicates that the traffic is likely to be anomalous, and notice that is assigned as an anomaly but it does not reach a consensus from all anomaly detectors. Several researchers used MAWILab for anomaly detection [53] and generated labeled flow [68]. The limitation of this dataset is that the packet traces are captured for 15 minutes each day. The header information is available in the packet traces but the payload is removed.

4.1.3 *CAIDA (Center of Applied Internet Data Analysis)*

CAIDA has several different types of datasets, categorized as ongoing, one-time snapshot, and complete [18]. CAIDA collects the data from different locations, and each of the datasets has different characteristics, such as Distributed Denial of Services (DDoS) attack, UDP probing, BGP monitoring, IPv4 census with passive traffic traces captured from a darknet, an academic ISP, and a residential BGP with active measurements of ICMP ping, HTTP GET and traceroutes. Most of the datasets are anonymized with IP addresses and the payload, which severely reduces their usefulness. Based on their catalog, during 2017–2020, most of the papers related to IDS and security focused on Denial of Service (DoS) [61, 84], Distributed Denial of Service (DDoS) [56], DNS security [55], Network Telescope Daily Randomly, and Uniformly Spoofed Denial-of-Service (RSDoS) Attack Metadata. Each record contains the IP address of the attack victim, the number of distinct attacker IPs in the attack, the number of distinct attacker ports and target ports, the cumulative number of packets observed in the attack, the cumulative number of bytes seen for the attack, the maximum packet rate seen in the attack as the average per minute, the timestamp of the first and the last observed packet of the attack, the autonomous system number of target_IP at the time of the attack, and the country and continent geolocation of target_IP at the time of the attack. This dataset is updated every day.

4.1.4 *SimpleWeb*

SimpleWeb is a dataset collected from the network of the University of Twente [13]. This dataset contains packet headers of all packets that are transmitted over the uplink of access to the internet. The packets are captured with tcpdump and Netflow version 5. The payload from the packets is removed because it contains sensitive information, such as HTTP requests or conversations of instant messaging applications. The labeled dataset for suspicious traffic is collected by using a honeypot server. Despite the unavailability of no ground-truth data, researchers still use it to compare it with the different real-world environment (e.g., campus network, backbone link) [52], while others employ it for background traffic for botnet detection [145], and to evaluate the publicly available dataset for similarity searches to detect network threats [22].

4.1.5 *NSL-KDD*

NSL-KDD is an updated dataset that tries to solve some of the inherent problems in the KDD99 dataset [136]. The NSL-KDD dataset contains features and labels indicating either normal or an attack, with specific types of attacks. Every instance in the training set contains a single connection session, which is divided into four groups, such as basic features from the network connection, content-related features, time-related features, and host-based traffic features. Each instance is labeled either as normal or attack. These attacks are categorized into four groups: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L), and Probing. Many researchers use it as a benchmark to help them to compare the performance of their network intrusion detection systems. Several groups of researchers used different scopes, such as IoT-based networks [81] and Vehicular

Ad Hoc Network (VANET) [47]. The former is for SYN flood, UDP flood, and Ping of Death (PoD) detection, while the latter is mostly for DDoS detection. Other researchers used different methods and switched from conventional machine learning to deep learning based methods [35, 133, 155].

4.1.6 *IMPACT*

IMPACT is a marketplace of cyber-risk data. The data distribution and tool repository are provided by multiple providers and stored and accessed from multiple hosting sites [59]. The datasets related to cyber-attacks, such as the daily feed of network flow data produced by Georgia Tech Information Security Center's malware analysis system, are updated once a year. These datasets are only open for specific countries based on the approval by the Department of Homeland Security (DHS).

4.1.7 *UMass*

UMass is a trace repository provided by the University of Massachusetts Amherst [141]. The network-attack-relevant data are provided with various types of data, such as traffic flow from the TOR network [94], a trace of attack simulation on peer-to-peer data sharing network [15], passive localization attack simulation with reality mining dataset [37] containing sensor data (proximity, location, location labels, etc.), and survey data (personal attributes, research group, position, neighborhood of apartment, and lifestyle).

4.1.8 *Kyoto*

This dataset was created between 2006 and 2015 by Kyoto University through honeypot servers. This dataset was created using Bro 2.4 (the former name of Zeek) with 24 statistical features consisting of 14 features extracted based on the KDD99 dataset and an additional 10 features, such as IDS_detection, Malware_detection, Ashula_detection, Label, Source_IP_Address, Source_Port_Number, Destination_IP_Address, Destination_Port_Number, Start_Time, and Protocol [73]. The information is limited to the attack information targeting the honeypot server. There are no packet traces or information about the payload. Furthermore, the information on how to label the dataset is not found [129]. Several published papers using the Kyoto dataset focused on anomaly detection, especially on the feature analysis [127], feature dimensionality reduction and ensemble classifier [115].

4.1.9 *IRSC*

This dataset was created by Indian River State College and consists of network flows and full packet capture [159]. The dataset represents a real-world environment in which the attack traffic has two different types: the controlled version, which is synthetically created by the team, and the uncontrolled version, which are the real attacks. The flow based traffic was created with the Silk [69] and the full packet capture was created with the Snort IDS [128]. The additional source of flow data was produced from the Cisco firewall using NetFlow version 9. While the authors stated that the dataset is a complete capture with payload and flow data, unfortunately, it is not publicly available.

4.1.10 *UNSW-NB15*

UNSW-NB15 was created using a commercial penetration tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). This tool can generate hybrid synthetically modern normal activities and contemporary attack behaviors from network traffic [93]. They collected tcpdump traces for a total duration of 31 h. From these network traces, they extracted 49 features categorized into five groups: flow features, basic features, content features, time features, and additional generated features. Feature and statistical analyses are the most common methods used in several published papers employing UNSW-NB15 [63, 72, 105]. Whereas [105] could obtain 97% accuracy by using 23 features, [63] incorporated the XGBoost algorithm for feature reduction, using several traditional machine learning algorithms for evaluation, such as Artificial Neural Network (ANN), Logistic Regression (LR), k-Nearest Neighbor (kNN), Support Vector Machine (SVM) and Decision Tree (DT).

4.1.11 *UGR'16*

This dataset was created from several NetFlow v9 collectors located in the network of a Spanish ISP [86]. It is composed of two different types of datasets that are split in weeks. First, the calibration set contains real background traffic data, and second, the test data contain real background traffic and synthetically generated traffic data with well-known types of attacks. Due to the nature of the NetFlow data, payloads from the network traffic were not included. The types of attacks implemented in this dataset are Low-rate DoS, Port scanning, and Botnet traffic. Between 2017 and 2021, I found mixed methods from several published papers, such as [104, 106], and Rajagopal et al. [106], who argued that conventional machine learning methods were ineffective and instead used stacking ensembles to improve performance and reliable predictions, while [104] proposed hybridized multi-model system to improve the accuracy of intrusion detection. Ref. [148] addressed imbalanced data problems by producing synthetic data with the Generative Adversarial Network (GAN).

4.1.12 *CICIDS-2017*

This dataset was created by the Canadian Institute for Cybersecurity at University of Brunswick in 2017. The purpose of CICIDS-2017 was intrusion detection, and it consisted of several attack scenarios. In this dataset, the attack profiles were produced using publicly available tools and codes. Six attack profiles were implemented, such as brute force, heartbleed, botnet, DoS, DDoS, web attack, and infiltration attack. The realistic background traffic was generated, using a B-Profile system [122]. The B-Profile system extracted 25 behaviors of users based on several protocols, such as HTTP, HTTPS, FTP, SSH, and SMTP. The network traffic features were captured with CICFlowMeter [74], which extracted 80 features from the pcap file. The flow label included SourceIP, SourcePort, DestinationIP, DestinationPort, and Protocol. Mixed methods were used, incorporating CICIDS-2017 to detect specific attacks such as DoS attack [71] by using feature reduction, web-attack detection [70], and anomaly web traffic [135] with ensemble architecture and feature reduction. Others were improving the AdaBoost-based method [149] to counter the imbalance of the training data [130], and combining feature selection and information gain to find relevant and significant features and to improve accuracy and execution time.

4.2 DATASET REQUIREMENTS

While the authors of ISCX [124], UGR'16 [86], and CICIDS-2017 [74] introduced a new dataset and provided extensive requirements about the dataset, their works have different research objectives and scope. In contrast to their earlier dataset, this work aims to complement the gap in the previous requirements.

4.2.1 *Requirements for IDS Evaluation Datasets*

Generally, different datasets have different assets and requirements. Shiravi et al. [124] focused on accurate labeling in the dataset by building a systematic profile to generate the dataset. They argued that the network traffic should be as realistic as possible, so a complete capture in a realistic network must be satisfied. It will impact anonymity and lead to potential privacy issues. Fernandez et al. [86] provided only information flow and focused on the duration of the capturing. Furthermore, a flow format with only 5-tuple is not enough and needs additional features if the malicious traffic is delivered via an encrypted protocol, such as HTTPS. I found that the requirements to build an IDS dataset from Sharafaldin et al. [122] is extensive. Unfortunately, their generated traffic comes from an emulated network, which is missing a realistic environment. In addition, the information about ground-truth and how the labeling works was not found in their works and, thus, is likely to be inaccurate and unreliable for analysis. Cordero et al. [27] created a tool called ID2T and I found that their requirements are better in practical terms. They categorized the requirements into functional and non-functional ones. Functional requirements focus on what is needed to construct datasets, while the non-functional requirements specify several criteria that need to be satisfied to be of practical use.

All of the requirements share high similarity. However, none of these works highlighted the importance of encrypted traffic in the dataset, and this is one of the emphases in my requirements. I derived my requirements for datasets based on the above works as well as by reviewing the existing datasets which described that the quality of the dataset mostly affects the outcome of the NIDS system. I classified the requirements into content requirements and process requirement. The content requirements are similar to [27], such as functional requirement, which focuses on what is needed to construct a dataset, and [124] on complete network traces and realistic network traffic capture. The process requirement is similar to that of [86] in the documentation point. While this is not enough, the information on how to produce a new dataset and practical implementation is does not exist.

The proposed requirements try to fill the gap of information from previous datasets. Based on my content requirements, I found at least four missing points:

(1) Most of the datasets were not anonymized, such as KDD99, SimpleWeb, NSL-KDD, Kyoto, IRSC, and UNSW-NB15. I chose to preserve privacy by anonymizing only a specific part of the background traffic based on the Crypto-Pan algorithm.

(2) The majority of the datasets were impractical to generate, such as KDD99, CAIDA, NSL-KDD, IMPACT, UMass, IRSC, UNSW-NB15, and CICIDS-2017.

(3) They did not have ground-truth data, such as MAWILab, CAIDA, SimpleWeb, IMPACT, UMass, Kyoto, and CICIDS-2017.

(4) As for encryption information, most of the datasets contained non-encrypted traffic, except for MAWILab, UGR'16, and CICIDS-2017. These datasets neither focused on nor classified encrypted traffic. However, HIKARI-2021 particularly addressed encrypted traffic.

The content requirements focus on the assets of the dataset to achieve a practical way to produce a dataset, while the process requirement specifies the information on how the dataset is built, so a new dataset can be built in the future using the same process. These requirements are listed below along with some descriptions of each item.

4.2.1.1 *Content Requirements*

(1) Complete capture: complete capture of the network traffic, such as communication between host, broadcast message, domain lookup query, and the protocol being used. The most important thing from complete capture is that both flow data and pcap should be available.

(2) Payload: payload is not needed for a flow-based approach. However, having comprehensive information and extracting the most out of the data is important. HIKARI-2021 is

the dataset that provides labeled encrypted traffic, while the well-known datasets do not focus on encrypted traffic. There is a possibility that a full payload captured might be useful in the future.

(3) Anonymity: synthetic traffic should provide full packet capture, while real traffic must anonymize certain packets to preserve privacy.

(4) Ground-truth: the datasets should provide realistic traffic from a real production network, compared with the synthetic traffic, and ensure no unlabeled attack in the ground-truth.

(5) Up to date: both packet traces from flow data and pcap should always be accessible by repeating the capturing process of the network traffic. Because the data are subject to change over time, repeating the procedures guarantees that the dataset always obtains the latest information.

(6) Labeled dataset: correctly labeling data as malicious or benign is important for accurate and reliable analysis. The labeling process is a manual task and determined by the flow with a combination of the source IP address, source port, destination IP address, destination port, and protocol.

(7) Encryption Information: information on how to establish benign or malicious traffic must be stated. I focused on application layer attacks, such as brute force and probing that employ HTTPS with TLS version 1.2 to deliver the attacks.

4.2.1.2 *Process Requirement*

Methods: producing a new dataset with specific requirements and practical implementation is important. Therefore, the methods should cover information on how to generate the dataset, how to generate the benign and attack traffic, how the background traffic is being captured, how the labeling process works, and how to implement it in the network. Furthermore, I need to determine the scenarios and how to deliver the synthetic attack in the network. In addition, the information of what features and how many can be extracted from the packet traces should be declared. Information on how to make a new dataset should be available in details and practical to generate.

4.2.2 *Comparison of the Existing Datasets against the above Requirements*

Comparisons between IDS datasets are shown in Table 8, where I assessed the datasets in Section 4.1, based on the predetermined requirements in Section 4.2.1.

Table 8: Comparison of IDS datasets based on the requirements in Section 4.2.1 with Complete Capture (CC), Payload (P), Anonymity (A), Ground-Truth (GT), Up to Date Traffic (UT), Labeled (L), Encryption (E), and Practical to Generate (PG).

Dataset	CC	P	A	GT	UT	L	E	PG
KDD99 [80]	Yes	Yes	No	Yes	No	Yes	No	No
MAWILab [46]	Yes	No	Yes	No	Yes	Yes	Yes	Yes
CAIDA [18]	Yes	No	Yes	No	Yes	No	No	No
SimpleWeb [13]	Yes	No	No	No	No	No	No	Yes
NSL-KDD [136]	Yes	Yes	No	Yes	No	Yes	No	No
IMPACT [59]	Yes	No	Yes [1]	No	Yes	No	No	No
UMass [141]	Yes	Yes	-	No	No	No	No	No
Kyoto [73]	Yes	Yes	No	No	No	Yes	No	Yes
IRSC [159]	Yes	Yes	No	Yes	No	Yes	No	No
UNSW-NB15 [93]	Yes	Yes	No	Yes	No	Yes	No	No
UGR'16 [86]	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes
CICIDS-2017 [74]	Yes	Yes	Yes	No	No	Yes	Yes [2]	No

[1] Mix datasets with partial anonymization
[2] Mix data between un-encrypted data, such as HTTP, and encrypted data, such as SSH.

I was unable to find the information regarding the anonymity of the UMass dataset; therefore, no indicator was given. As for the IMPACT dataset, this platform has many datasets, some parts of which are anonymized, while others are not. In the CICIDS-2017 dataset, one part of the traffic has samples for encrypted traffic with benign and attack profiles.

I highlighted four points from the above comparisons. First, there is a need to have encrypted samples of benign and attack traffic. I found that [46] in their dataset have information on whether the traffic is anomalous or suspicious but it depends on the anomaly detectors. The payload from the packet traces was not included. This has limited the capability of IDS because many attacks cannot be detected only by network flow with only

5-tuple attributes. In addition, [74] in their datasets included the traffic from benign and attack profiles from SSH. While this is beneficial, the diversity of the attack needs to be expanded to applications, such as browser attacks, or with different protocols, such as HTTPS, and I did not find that this protocol existed in their dataset. Second, I discovered that most of the datasets were not anonymized, which was probably due to the fact that their testing beds were in a controlled environment or they had consent with their activity. The former is the best option with the consequences that the traffic will have more synthetic traffic while reducing the real traffic. The latter is preferred if they can preserve privacy. Furthermore, privacy can be maintained by anonymizing the traffic, but being highly anonymized may decrease the results of the analysis [65, 124, 142]. Third, I also highlighted that most of the datasets did not have ground-truth data and background traffic, which limited the analysis only to their model. Fourth, there is a need for a methodology on how to create a new dataset. This is due to the nature of the network environment that is subject to change over time. How to create new datasets with the practical implementation is important, so researchers may make their dataset and evaluate it with their environment. This methodology can serve as a guideline for IDS researchers to follow the making of a practical dataset.

4.3 HIKARI-2021 GENERATION METHODOLOGY

In this section, I explained my methodology to produce my dataset, which I called HIKARI-2021. The process started with creating a victim network, where background traffic was captured, and attackers generated synthetic benign traffic, using a benign profile (details in Section 4.3.3), and malicious traffic, using an attacker profile (details in Section 4.3.4). The attacker traffic was captured in the attacker network. I did this to differentiate between synthetic benign and malicious traffic. Distinguishing between benign and malicious traffic was based on several criteria (details in Section 4.3.4). I then processed the packet traces to anonymize the background traffic and extract the features. The packet traces and extracted features, as well as the documentation, constituted the produced dataset.

I focused on application layer attacks that employ HTTPS. Based on the report from the 2021 Data Breach Investigation, 80% of the attack vectors came from application-layer attacks. There are many attacks on the internet but this work does not examine the number of attacks I can generate. The survey from netcraft.com and websitesetup.org revealed that WordPress, Joomla, and Drupal are among the ten most popular open-source CMSs, with the combined market share of almost 50%. Based on the information from CVE, more than 300 vulnerabilities existed for WordPress from 2006 to 2021, 92 vulnerabilities for Joomla from 2004 to 2021, and 202 vulnerabilities for Drupal from 2002 to 2021. More than half of the vulnerabilities from these three CMSs are part of Brute Force and Probing. Therefore, this research does not aim to examine the attack diversity but is more emphasized on the kind of attack to be delivered in the encrypted network. I decided to focus on the common application-layer attacks, such as brute force and probing on this ground. This is done to allow the IDS researcher to build their script based on my tool to enrich the attack, such as SQL Injection, Denial of Service, etc.

4.3.1 *Network Configuration for Generating Dataset*

Figure 4 shows my network configuration, where attackers are on a separate network from the victims.

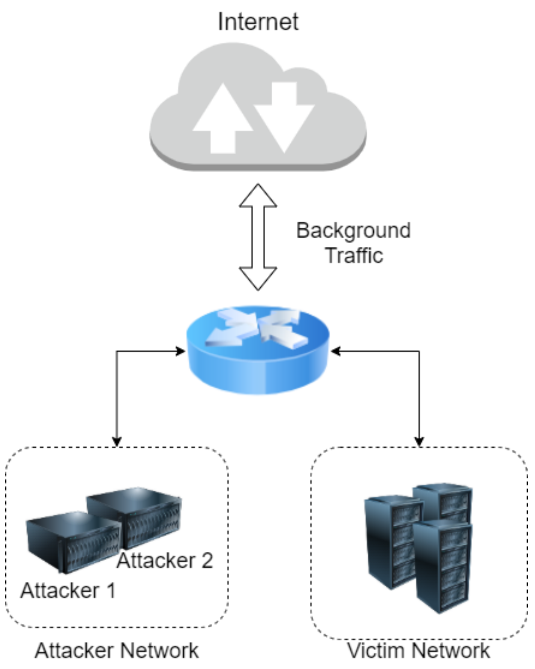

Figure 4: Network configuration to generate dataset.

The format of the data I captured is pcap, which provides the configuration with the following important point:

(1) The attacker network with two machines was deployed with CentOS 7 and CentOS 8. There were no specific criteria of the attackers' machines as long as they can run Bash and Python scripts. The Python version was 3.8.8 from Miniconda 3.

(2) In the victim network, three machines were deployed with one Debian 8 machine running Joomla 3.4.3, and two Debian 9 machines running Drupal 8.0 and WordPress 5.0. There were no specific criteria for the OS version for the victim network, and the three different Content Management Systems (CMS) such as Drupal, WordPress, and Joomla used default themes and plugins. These three open-source CMSs were chosen based on their popularity. These machines were used for collecting the background traffic.

4.3.2 *Background Profile*

Generating realistic data is important. For the background traffic, I captured all the data without any filter or firewall in the victim network. Therefore, there is a possibility that the background traffic may contain malicious traffic or attacks. To preserve privacy without degrading the result of the analysis, I anonymized several pieces of information, such as IP address and the payload.

4.3.3 *Benign Profile*

To generate the benign profile, I considered using a profile similar to human behavior. To achieve this goal, I used Selenium [118], which runs two headless browsers: Google Chrome and Mozilla Firefox. These two browsers act like humans by clicking random links from multiple websites, sign up as a user, sign in, post an article to the target victim's server, and sign out. To behave like a human and to avoid being detected as a bot or web spider, I used several configurations, such as user-agent and random delay, for every sequence of action. The addresses of the websites are from Alexa's top 1 million visitors [5]. The benign profile was developed with Python script; this activity simulates benign traffic. All benign traffic is captured without anonymizing the payload and this type of traffic generates HTTPS only.

4.3.4 *Attacker Profile*

The attack traffic is generated synthetically, by firstly targeting a specific page for user login of the CMSs, and then by scanning their vulnerability. Both of the attacks are delivered via the HTTPS protocol. The attacks are delivered on different days with different scenarios (details in Section 4.3.5). The types of attacks are listed as follows:

(1) Brute force attack: this attack is the most famous for cracking passwords. The attacker usually repeatedly tries to gain the target over and over using all possible combinations using a dictionary of possible common passwords [30]. I developed a script that mimics a brute force attack, using a browser to deliver the attack. I intentionally added a user to the three different CMSs to play a role as an admin and password, which I took randomly from [30]. The purpose was to make sure that the brute force attack was delivered successfully.

(2) Brute force attack with different attack vectors: while the first type of attack uses the browser as the attack vector, the second attack uses a different attack vector, XMLRPC. I developed a script that accesses XMLRPC for gaining valid credential access.

(3) Probing: this is also called vulnerability probing. This script scans the web applications, such as Joomla, WordPress, and Drupal to find their vulnerability. The tools for vulnerability scanning are publicly available. For this dataset, the scripts used these probing scripts: droopescan [36] for WordPress and Drupal, and joomscan [62] for Joomla.

I provided the template script to customize the attack profile so researchers may use it for making custom attacks using different vectors. Distinguishing between an attack profile and benign profile was done based on the source IP address, source port, destination IP address, destination port, protocol, and the day both of the profiles being generated. In addition, to determine benign traffic, any destination addresses in the Alexa list are considered benign.

4.3.5 *Scenarios*

I captured the traffic non-consecutively between 28 March 2021 and 4 May 2021, with each capture session lasting for 3 to 5 hours. In the first scenario, no attack traffic was generated, and only background traffic was being captured. In the second scenario, brute force attack traffic was generated for 2 days. Furthermore, a brute force with different attack vectors was generated in the third scenario, while the last scenario, generated the vulerability scanning of WordPress, Joomla, and Drupal.

4.3.6 *Dataset Preprocessing*

The traces were captured using tcpdump with full packet capture. As for the background traffic, I fully captured the traffic but then I anonymized it to maintain privacy. To preserve privacy, I used a Crypto-PAn based algorithm [41]. The complete dataset contains several files: pcap files from background traffic, and synthetic attacks. The flowmeter files with pkl and CSV are available for downloads [43]. The preprocessing flow from pcap files into CSV files is presented in Figure 5.

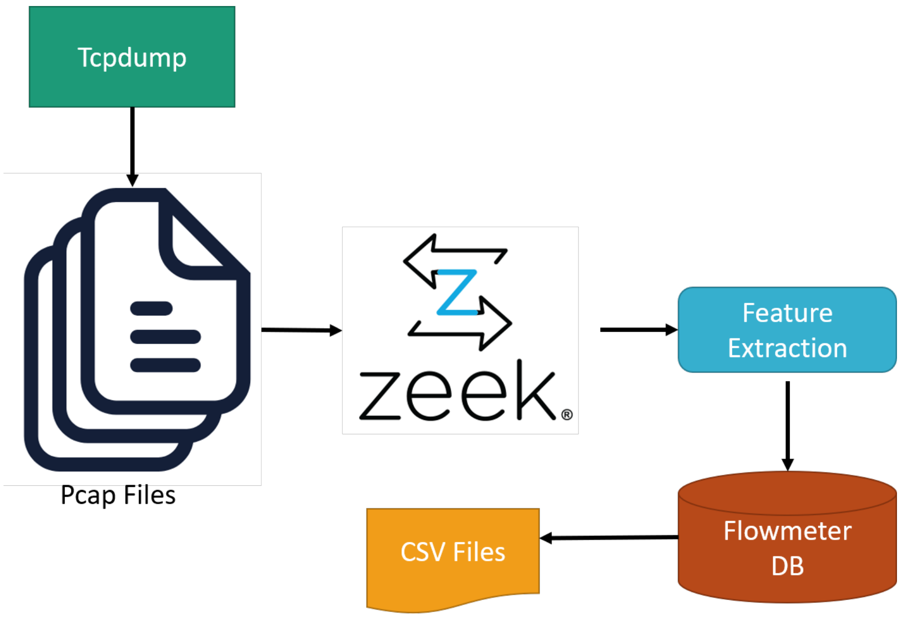

Figure 5: The preprocessing flow of HIKARI-2021 dataset.

4.3.7 *Labeling Process*

During background traffic validation, I found malicious traffic resulted from the Zeek rules, which shows that some traffic is that of malicious cryptomining, such as XMRIGCC. I then separated and added it as a new attack, which I categorized as XMRIGCC CryptoMiner. Labels were applied on a per-flow basis. In the background traffic, I did not find any attack besides the cryptomining. Other than background, my labeling was based on the generated synthetic rules, such as source IP address, source port, destination IP address, destination port, and protocol. The dataset consists of two labels: traffic_category and label. The former represents the name of the traffic category, while the latter is only a single value with 0 representing Benign, and 1 representing Attack as shown in Table 9.

Table 9: Labeled features information.

Traffic Category	Label	Total Flows (Flowmeter)	No. Encrypted Session
Background	Benign	170,151	36,782
Benign	Benign	347,431	116,309
Bruteforce	Attack	5884	5884
Bruteforce-XML	Attack	5145	5145
Probing	Attack	23,388	23,388
XMRIGCC CryptoMiner	Attack	3279	0

4.3.8 *Feature Description*

HIKARI-2021 features were extracted using Zeek. Table 10 shows the features while Figure 6 displays a statistical description of the features. Most of the features were adopted from CICIDS-2017, while uid, originh, originp, responh, responp, traffic_category, and Label were derived from Zeek.

Table 10: List of features in HIKARI-2021.

No	Feature	No	Feature	No	Feature
1	uid	36	bwd_pkts_payload.min	71	bwd_bulk_rate
2	originh	37	bwd_pkts_payload.max	72	active.min
3	originp	38	bwd_pkts_payload.tot	73	active.max
4	responh	39	bwd_pkts_payload.avg	74	active.tot
5	responp	40	bwd_pkts_payload.std	75	active.avg
6	flow_duration	41	flow_pkts_payload.min	76	active.std
7	fwd_pkts_tot	42	flow_pkts_payload.max	77	idle.min
8	bwd_pkts_tot	43	flow_pkts_payload.tot	78	idle.max
9	fwd_data_pkts_tot	44	flow_pkts_payload.avg	79	idle.tot
10	bwd_data_pkts_tot	45	flow_pkts_payload.std	80	idle.avg
11	fwd_pkts_per_sec	46	fwd_iat.min	81	idle.std
12	bwd_pkts_per_sec	47	fwd_iat.max	82	fwd_init_window_size
13	flow_pkts_per_sec	48	fwd_iat.tot	83	bwd_init_window_size
14	down_up_ratio	49	fwd_iat.avg	84	fwd_last_window_size
15	fwd_header_size_tot	50	fwd_iat.std	85	traffic_category
16	fwd_header_size_min	51	bwd_iat.min	86	Label
17	fwd_header_size_max	52	bwd_iat.max		
18	bwd_header_size_tot	53	bwd_iat.tot		
19	bwd_header_size_min	54	bwd_iat.avg		
20	bwd_header_size_max	55	bwd_iat.std		
21	flow_FIN_flag_count	56	flow_iat.min		
22	flow_SYN_flag_count	57	flow_iat.max		
23	flow_RST_flag_count	58	flow_iat.tot		
24	fwd_PSH_flag_count	59	flow_iat.avg		
25	bwd_PSH_flag_count	60	flow_iat.std		
26	flow_ACK_flag_count	61	payload_bytes_per_second		
27	fwd_URG_flag_count	62	fwd_subflow_pkts		
28	bwd_URG_flag_count	63	bwd_subflow_pkts		
29	flow_CWR_flag_count	64	fwd_subflow_bytes		
30	flow_ECE_flag_count	65	bwd_subflow_bytes		
31	fwd_pkts_payload.min	66	fwd_bulk_bytes		
32	fwd_pkts_payload.max	67	bwd_bulk_bytes		
33	fwd_pkts_payload.tot	68	fwd_bulk_packets		
34	fwd_pkts_payload.avg	69	bwd_bulk_packets		
35	fwd_pkts_payload.std	70	fwd_bulk_rate		

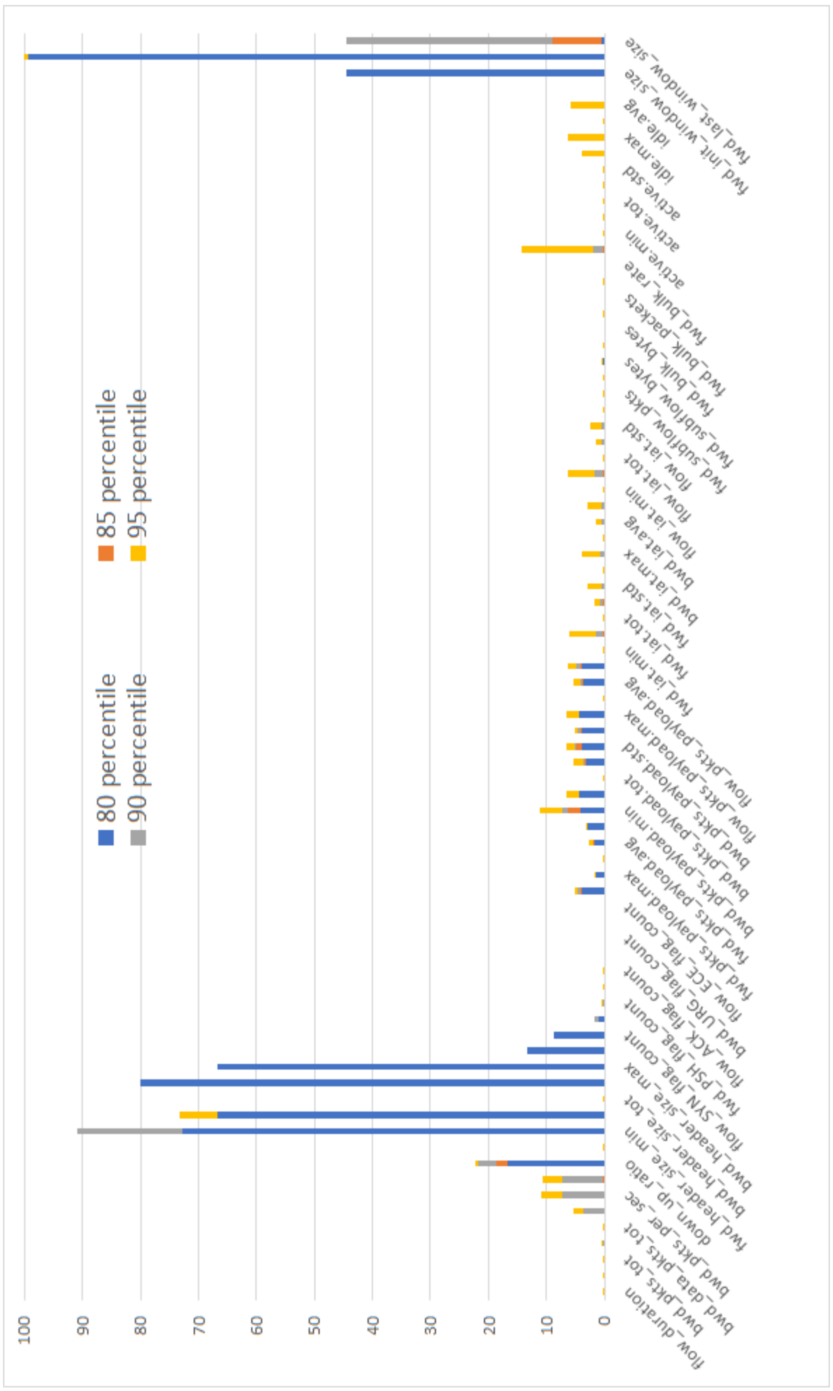

Figure 6: Most of the features are skewed, where the value of the 95th percentile is less than ten percent of the maximum value.

4.3.9 *Performance Analysis*

I conducted an examination using a basic performance analysis by means of four machine learning algorithms. Table 11 displays the performance of the examination results in Accuracy, Balanced Accuracy, Precision, Recall, and F1.

Table 11: Basic Performance Analysis.

Algorithm	Accuracy	Balanced Accuracy	Precision	Recall	F1
KNN	0.98	0.94	0.86	0.90	0.88
MLP	0.99	0.99	0.99	0.99	0.99
SVM	0.99	0.99	0.99	0.98	0.99
RF	0.99	0.99	0.99	0.99	0.99

4.4 COMPARISON OF KDD99, UNSW-NB15, CICIDS-2017, AND HIKARI-2021

Table 12 shows an analysis comparison among KDD99, UNSW-NB15, CICIDS-2017, and HIKARI-2021. The table consists of seven parameters: the number of unique IP addresses, simulation, duration of the data capture, format data, attack category, feature extraction tools, and the number of features extracted from each dataset. The number of unique IP addresses of CICIDS-2017 and HIKARI-2021 were from the unique destination IP addresses from the dataset. Partial means that the dataset is mixed between a simulation or synthetic and real-network environment.

Table 12: The dataset comparison of KDD99, UNSW-NB15, CICIDS-2017, and HIKARI-2021.

Parameters	KDD99	UNSW-NB15	CICIDS-2017	HIKARI-2021
Number of unique IP address	11	45	16,960	7991
Simulation	Yes	Yes	Partial	Partial
Duration of the data being captured	5 weeks	16 h	65 h	39 h
Format of the data collected	tcpdump, BSM, dumpfile	pcap files	pcap files	pcap files
Number of Attack categories	4	9	7	4
Feature extraction tools	Bro-IDS	Argus, Bro-IDS	CICFlowmeter	Zeek-IDS, python
Number of features	42	49	80	86

4.5 SUMMARY

Publicly available up-to-date datasets to benchmark and compare data among IDS are important, especially as the network traffic is changing over time. I made a new requirement for building new datasets, which are lacking in the existing datasets, such as anonymization, payload, ground-truth, encryption, and a practical method to implement it. Anonymizing certain data will prevent privacy issues, while capturing data with the payload will enrich the information that I can collect for detecting malicious traffic within encrypted traffic. Providing the ground-truth data is crucial, so no unlabeled attack is recorded in the dataset. The lack of existing datasets with encrypted traffic, even though most present-day traffic uses it for delivering attacks, has become my concern. I then generated a new IDS dataset called HIKARI-2021, which covers the network traffic with encrypted traces. The datasets were produced with a mix of ground-truth data, which are missing in the existing IDS datasets. The datasets are available publicly. I adopted more than 80 features from CICIDS-2017 and added more features as a reference, such as a source IP address (originh), source port (originp), destination IP address, and destination port. I labeled each flow as benign or attack, where benign has two categories (Benign or Background), while attack has four (Bruteforce, Bruteforce-XML, Probing, and XMRIGCC CryptoMiner).

I want to highlight what makes my dataset different from the existing IDS datasets based on my proposed ideal requirements. The first difference is seen from the content requirements, such as complete capture, for which I provided all traces with pcap files (e.g., background traffic, benign, and attack); the payload was provided with the exception that I anonymized the background traffic, while anonymity is part of a requirement to preserve privacy. The ground-truth and labeled were manually evaluated based on the source IP address, source port, destination IP address, destination port, and protocol. This process aims to make sure that no unlabeled attack is in the ground-truth. The last requirement is encryption, which serve as one of the most important requirements, since unknown malicious traffic uses these attack vectors to deliver attacks.

The second difference to highlight is process requirement. It is to ensure that researchers can follow the guidelines to create their dataset. The information on how to generate the synthetic attacks and the network configuration should be available. I provided the scripts on how to capture and generate the synthetic attacks from the attack profile. The tools for mimicking human interaction, such as browsing and clicking random links, are available. These two profiles, the attack profile and benign profile, are important for producing new data if researchers want to add more attack vectors and update the traffic with their own needs. The labeling process script to produce ground-truth data is provided. The process requirement can be implemented in the controlled environment so that researchers can make new datasets based on their network configuration. For a basic evaluation, I examined the performance of the HIKARI-2021 dataset in terms of Accuracy, Balanced Accuracy, Precision, Recall, and F1, using four machine learning algorithms.

5 ENCRYPTED MALICIOUS TRAFFIC DETECTION

The traffic encryption prevents traditional NIDS from inspecting the payload, which is crucial to determine whether the traffic is benign or malicious. On the other hand, the most well-known approaches, such as port-based [120] and signature-based approaches [21], do not work well. A port-based inspection can classify traffic according to the service name and port number registry assigned by Internet Assigned Number Authority (IANA) [121]. However, this approach does not work correctly if the application changes to a custom port. Signature-based network intrusion detection system may solve the problem, but this condition needs a predetermined set of rules and features, which is unreliable to detect unknown malicious traffic. Many researches have used payload-independent statistical approaches from 1999 up to today [7, 28, 89, 156]. The use of flow attributes, such as packet length, inter-arrival time, and flow duration makes it unnecessary to dissect the payload and avoid a user's privacy breach [125].

5.1 DATASET DESCRIPTION

In this research, I employed two public datasets that contain malicious encrypted traffic. I chose datasets with higher encrypted malicious traffic than the other publicly available datasets. Nonetheless, I did not include datasets, such as [82], as they are not publicly available. TU-Malware-Capture [131] is a dataset produced from Malware Capture Facility Project [87] responsible for long-term captures. Following this, I utilized Jason Stroschein public Github malware samples [60] and Zeus, benign, and Cobalt from TU-Malware-Capture and Trickbot from Jason Stroschein. Zeus, Cobalt, and Trickbot were selected based on their communication, the variety of the activities [154], and how their activity opens the backdoor for further malicious application [140].

All the datasets were raw PCAP files consisting of TLS packets and any other packets. I considered only the TLS packets. Table 13 shows the basic information of label, total packets, total TLS sessions, and the average number of application data packets per session (Avg. App Data / Session) from the total of the datasets. The Zeus, Benign, and Cobalt are derived from CTU-Malware-Capture, while Trickbot is from public Github of Jason Stroschein.

Table 13: Basic information of the datasets.

Label	Total Packets	Total TLS Sessions	Avg. App Data / Session
Zeus	2,201,308	10,581	1.932
Benign	1,601,294	1,461	1.249
Cobalt	1,471,709	250	2.000
Trickbot	895,172	33	1.909

5.2 METHODOLOGY

The key point of my research is to analyze the TLS session by identifying the features from the TLS handshake and the payload. Most of the encrypted network traffic has a specific format that differs from the others. Thus, using the knowledge of this format, it is possible to differentiate and identify the malicious traffic. TLS2Vec has three steps:

1) Feature Extraction: extract features from raw PCAP to build a corpus.

2) Building Vocabulary and Token Parser: uses tokenization technique to extract words from the training dataset and then applies word embedding technique to represent words.

3) Training Model: TLS2Vec trains the dataset using LSTM and BiLSTM.

Our study analyzes whether the TLS handshake and payload can be used to determine the malicious traffic from benign before the conversation finishes between client and server. I want to emphasize that the results from the methods can be used to take immediate action as soon as I can classify the traffic as malicious. Since opening and analyzing the encrypted content without a private key is almost impossible, this leaves us with the unencrypted content that resides in the header, the TLS handshake, and the statistical information of payload. I decided to analyze the TLS handshake and payload and remove the rest.

I did not include the analysis of the host's behavior from this research, since the analysis of the host's behavior was used to determine the encrypted communication from remote access trojan (RAT). The method includes the study of the transmission multiple session-based using 5-tuple. Thus, the method cannot run independently but serves as an ensemble with the NIDS based on host behavior.

5.2.1 *Feature Extraction*

TLS improves the SSL version 3 protocol that provides transport-level security over TCP protocol [6]. The TLS consists of a Record Protocol that acts as an envelope for Application Data [34]. There has been a limited possibility of extracting information from encrypted content. However, it is possible to obtain information from two ways: the unencrypted and its properties and the packet length of the payload. I can get information from the unencrypted phase: the initial handshake and its properties, and the authentication information exchange identifier from the client and the server. During the initial TLS handshake, the client and server are negotiated with both exchanging cipher suites information and which protocol version is used. Many cipher suites exist and are not all implemented by the client's application. Usually, the client's application specifies the supported cipher suites and which cipher suites are recommended for the initial handshake. The important thing from this behavior is to distinguish malicious applications from the client. The next is the authentication exchange information, such as the extensions [96]. My method mainly considered extensions such as *elliptic_curves* and *ec_point_formats* for the features, while the malicious applications tend to use weak encryption and offer fewer extensions. Such weak encryption is perhaps because of the lack of concern with a strong encryption mechanism. Some of the authors disregarded the encryption mechanism. On the other hand, the other is only concerned with strong encryption [153].

In the payload-based inspection, most of the techniques use the information from the payload of the application layer [66]. My method is different from [119] and [20], which requires decrypting the traffic using the key. The packet length from the TCP Application Data is the additional information feature extracted from encrypted traffic. Although some TLS server communication parameters can be changed over time, the application-specific profile usually reflects the Application Data's content

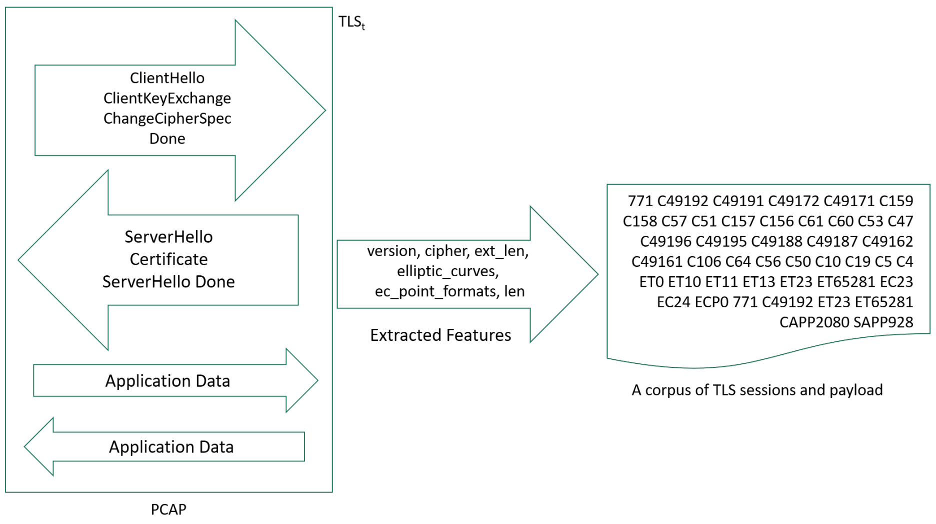
TLS$_t$
ClientHello
ClientKeyExchange
ChangeCipherSpec
Done
ServerHello
Certificate
ServerHello Done
Application Data
Application Data
PCAP
version, cipher, ext_len, elliptic_curves, ec_point_formats, len
Extracted Features
771 C49192 C49191 C49172 C49171 C159 C158 C57 C51 C157 C156 C61 C60 C53 C47 C49196 C49195 C49188 C49187 C49162 C49161 C106 C64 C56 C50 C10 C19 C5 C4 ET0 ET10 ET11 ET13 ET23 ET65281 EC23 EC24 ECP0 771 C49192 ET23 ET65281 CAPP2080 SAPP928
A corpus of TLS sessions and payload

Figure 7: Overview of TLS2Vec.

size. I considered using the TCP packet length from Application Data. While I cannot see into the unencrypted content, the packet length is directly linked to the payload of the traffic and usually follows an application-specific profile [75]. Thus, I can use packet length to determine which traffic is benign or malicious based on the specific application profile.

The unencrypted communication phase between client and server complies with TLS protocol, which can be treated as one sentence in a TLS session. Each sentence has a specific pattern that can distinguish between malicious and benign. Similar words with correlation are represented by similar vectors closely placed in a vector space. In the network, the sequence from the TLS sessions is textual data that should be encoded into a numeric representation. In this case, I argue that a numeric vector can represent the sequence of traffic with similar vectors in the embedding space. Similar traffic represented by words will have the same vector space, benign or malicious.

Figure 7 shows the steps from feature extraction to building a corpus. The single box is a TLS session with two payloads from a PCAP file with a right arrow representing a client and a left arrow representing a server with features. The first two arrows in the left box are handshake while the rest represent TCP Application Data which I call payload. I collected zero to a total of two or more payloads from each client and server. Multiple TLS sessions exist in the PCAP file with the symbol of TLS$_t$. The arrow in the middle represents features that are being extracted to a corpus. The document symbol on the right represents a corpus containing TLS handshake and two payloads from the features.

The TLS sessions consist of hundreds of features. Most of the contents are encrypted except for the handshake contains handshake protocols like Client Hello, session ID, the version of the TLS, length, the cipher suites, compression method, and extensions. The PCAP contains multiple session conversations between the client and the server, from the three-way handshake to the TLS sessions. I extracted the features from the raw PCAP from two datasets and preprocessed the data based on the session by considering a raw PCAP with multiple TLS sessions (TLS_t, TLS_{t+1}...TLS_{t+n}). For each session, I extracted the TLS handshake TLS/SSL version (*version*), Cipher Suites List from Client Hello or Server Hello (*cipher*), Extension Length from Client Hello or Server Hello (*ext_len*), Elliptic Curves from Client Hello (*elliptic_curves*), Elliptic Curve Point Formats from Client Hello (*ec_point_formats*), and the payload (*len*). The extracted features were then constructed into words. Each record in my dataset describes one TLS session, which can be viewed as a sentence.

Our study treated a sentence as a TLS language that follows TLS protocol. Because I extracted TLS session raw PCAP files, the decimal representation from the extracted data would create a row of numbers representing a communication session between a client and a server. I then added a symbol on each feature based on their representation, as shown in Figure 7 in the document symbol. In the document symbol from Figure 7, C, ET, EC, ECP, CAPP, or SAPP represents *cipher*, *ext_len*, *elliptic_curves*, *ec_point_formats*, and *len*. The information in the document symbol is a sample of a single TLS session from the client and server taken from Trickbot malicious traffic. The collection of all these words is known as a corpus.

5.2.2 *Building Vocabulary and Token Parser*

In this phase, converting words from the corpus into vector space models uses Gensim [108] version 4.1.2. Word2vec technique trains each word in the corpus and converts each into a 300-dimensional vector. I used 300-dimensional based on several references [91, 92, 107]. In this phase, vocabulary was built using the tokenization technique to extract words from the training dataset and to apply word embedding. A token parser then used the vocabulary to convert each TLS session in the dataset into a token sequence.

In general, the Word2Vec technique has two architectures, the CBOW and Skip-gram. The CBOW tries to predict the target word based on the surrounding words' context. On the other hand, Skip-gram tries to predict the context word from a word. In CBOW, the word representation is combined to predict the word in the middle. Both architectures might be used for different purposes in the NIDS incorporated with machine learning. In terms of the training time between CBOW and Skip-gram, the former is slightly faster and has less computational cost, while Skip-gram might better predict a large dataset. Skip-gram is more suitable for anomaly detection, which does not make any assumption about the malicious traffic instead of on benign traffic. While in CBOW, the prediction of malicious traffic is based on the surrounding information and its similarity. To determine which one has a better prediction, I used CBOW and Skip-gram for evaluation.

5.2.3 *Training Model*

Two types of Recurrent Neural networks (RNN), LSTM and Bidirectional LSTM (BiLSTM), were used to evaluate the performance. LSTM and BiLSTM is specialized with their effectiveness in memorizing particular patterns. At the same time, both algorithms are suitable for detecting malicious traffic with payloads [109, 147].

5.3 EVALUATION

This section summarizes the performance of the TLS2Vec by performing with binary and multiclass targets. I grouped three malicious traffic such as Zeus, Cobalt, Trickbot into a single class Malicious for the binary target. The total number of TLS sessions of Malicious traffic was 10,163 after grouping. The considered evaluation used 5-fold cross-validation (CV-5). It is noteworthy that an imbalance of data distribution exists in this dataset. I used both the categorical_crossentropy and binary_crossentropy for the loss function. The Adam optimizer was applied with a learning rate of 0.001. The rest of the hyperparameters used their default values. The softmax was used for the activation function for categorical_crossentropy, while sigmoid was used for binary_crossentropy. Figure 8 illustrates

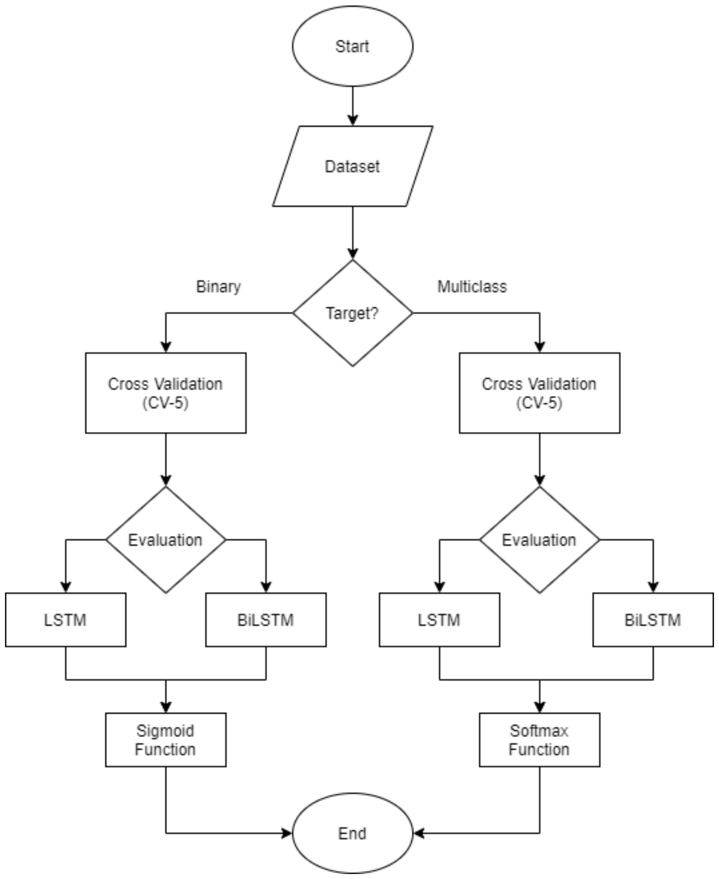

Figure 8: The evaluation procedure.

the evalution procedure. Table 14 provides the hyper-parameters regarding detection based approach on LSTM and BiLSTM.

In general, the overall accuracy metric might be used to measure the results in the binary target. However, to rely on the performance only for accuracy is unfavorable, especially when involving a class imbalance dataset. I adopted F1-score to give more insight into the dataset in this research. Based on my analysis of the datasets, I found that most of the payloads from the malicious traffic from each session mainly were two payloads, with several sessions having only three and very few sessions having more than that. From this result, I proceeded with my analysis using zero payload, one payload, and two or more payloads. I compared my analysis with a non-TLS2Vec method. In non-TLS2Vec I used the same features as in TLS2Vec except when I did not use Word2Vec and removed the symbols that represents features such as cipher, ext_len, elliptic_curves, ec_point_formats, and len. I did this because as far as I know, I did not found any similar methods.

Table 14: LSTM and BiLSTM hyper-parameter for binary dan multiclass target

Hyper-parameter	Binary Target	Multiclass Target
Activation Function	sigmoid	softmax
Epoch	30	30
Optimizer	adam	adam
Learning Rate	0.001	0.001
Batch Size	32	32
Loss Function	binary crossentropy	categorical crossentropy

In the first evaluation, I started with a TLS handshake with zero, one, and two or more payloads with TLS2Vec with CBOW, TLS2Vec with Skip-gram, and a Non-TLS2Vec. As illustrated in Figure 9, both CBOW and Skip-gram with LSTM and BiLSTM performed similarly well in detection, with the former slightly better.

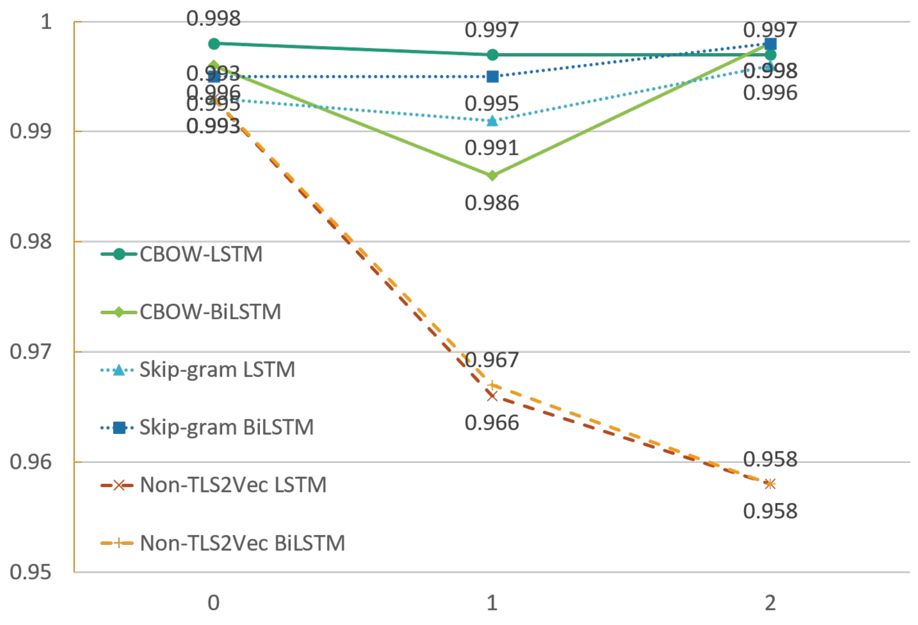

Figure 9: The evaluation between TLS2Vec with CBOW, TSL2Vec with Skip-gram, and Non-TLS2Vec. The x-axis is the payload from zero, one, and two or more payloads, while the y-axis is the F1 score.

The better performance was understandable since both can predict the next words in the input sentence from the TLS. In a non-TLS2Vec method, the performance significantly decreased along with the addition of one to two or more payloads.

In CBOW-LSTM, the high F1 score started with zero then it was relatively stable in one to two or more, indicating that the LSTM could detect almost all the Malicious and Benign with the default hyper-parameter. Compared with the counterparts, CBOW-BiLSTM slightly decreased after zero payload and then increased again in two or more. The primary reason was that CBOW-BiLSTM needed to fetch more training data to reach equilibrium with one payload. On the other hand, Skip-gram BiLSTM was slightly better than Skip-gram LSTM. The figure showed us that the start from zero which gradually increased after one payload indicated that having more payload was better for improving the performance. The result was coherent as BiLSTM run inputs in two ways, and this implies that having more payloads might offer better detection in Benign and Malicious using default hyper-parameters. Based on this result, I continued to use TLS2Vec with CBOW and Non-TLS2Vec, and exclude TLS2Vec with Skip-gram. In the subsequent evaluation, I used a different strategy for detection by using a multiclass target. The results showed in Figure 10.

The figure illustrated the trend in the increase of detection for both CBOW-LSTM, CBOW-BiLSTM, and non-TLS2Vec. In CBOW-LSTM, both Zeus and Cobalt were stable, starting from zero, one, and two or more. The result indicated that LSTM could detect all the Zeus and Trickbot starting from the TLS handshake. However, in the Benign class, the performance slightly decreased with one payload and then slightly increased with two or more payloads. The results showed that LSTM needed to fetch more training data to reach the same performance with zero payload for one to two or more. As for the counterpart, Trickbot increased along with payloads. Although the performance went up, the trend was stable. In a non-TLS2Vec, the performance decreased along with the addition of the payloads for all classes.

In the following evaluation, I added more parameters for evaluation using discretization transformation by binning the payload. Discretization transformation reduced the traffic patterns and discriminated among the traffic by grouping those with similar payloads [113, 114]. Furthermore, binning reduced the vocabulary. The payload from the datasets had 872 unique values.

The problem in the discretization was selecting the intervals or bins. I used two different methods for selecting the number of bins: bit-length, which I called Bit-Width, and the Sturges

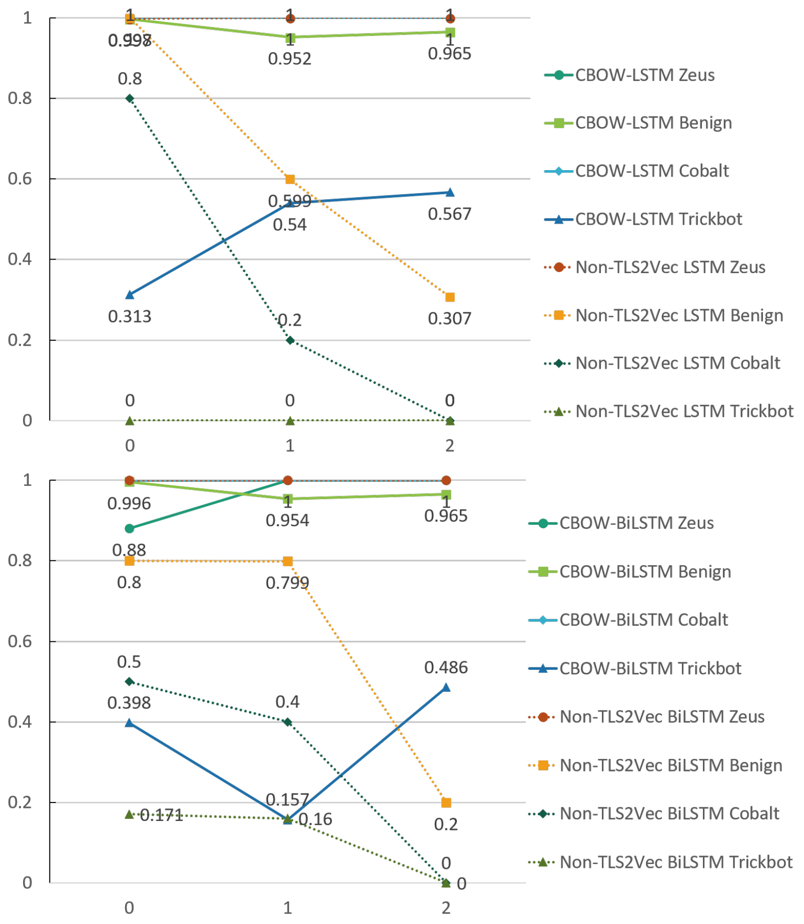

Figure 10: The multiclass target with LSTM (top) and BiLSTM (bottom). The x-axis is the payload from zero to two or more, while the y-axis is the F1 score.

rule, as this rule was used for constructing histogram and non-stationary data [139]. For the Bit-Width (BW), I used intervals from 0 to 63, 64 to 127, 128 to 255, 256 to 511, 512 to 1023, and equal to more than 1024. The payload in the intervals of six categories would be converted according to the bins. For the Sturges rule, I used two different combinations called Sturges-EqualWidth (SEW) and Sturges-EqualFreq (SEF). SEW ensured that each bin had an equal range of the original value, while with SEF, each bin had the same number of samples. For SEW and SEF, selecting a number of bins followed the Sturges rule for only one payload. The bins range was resulted from one payload then was applied to payload two or more.

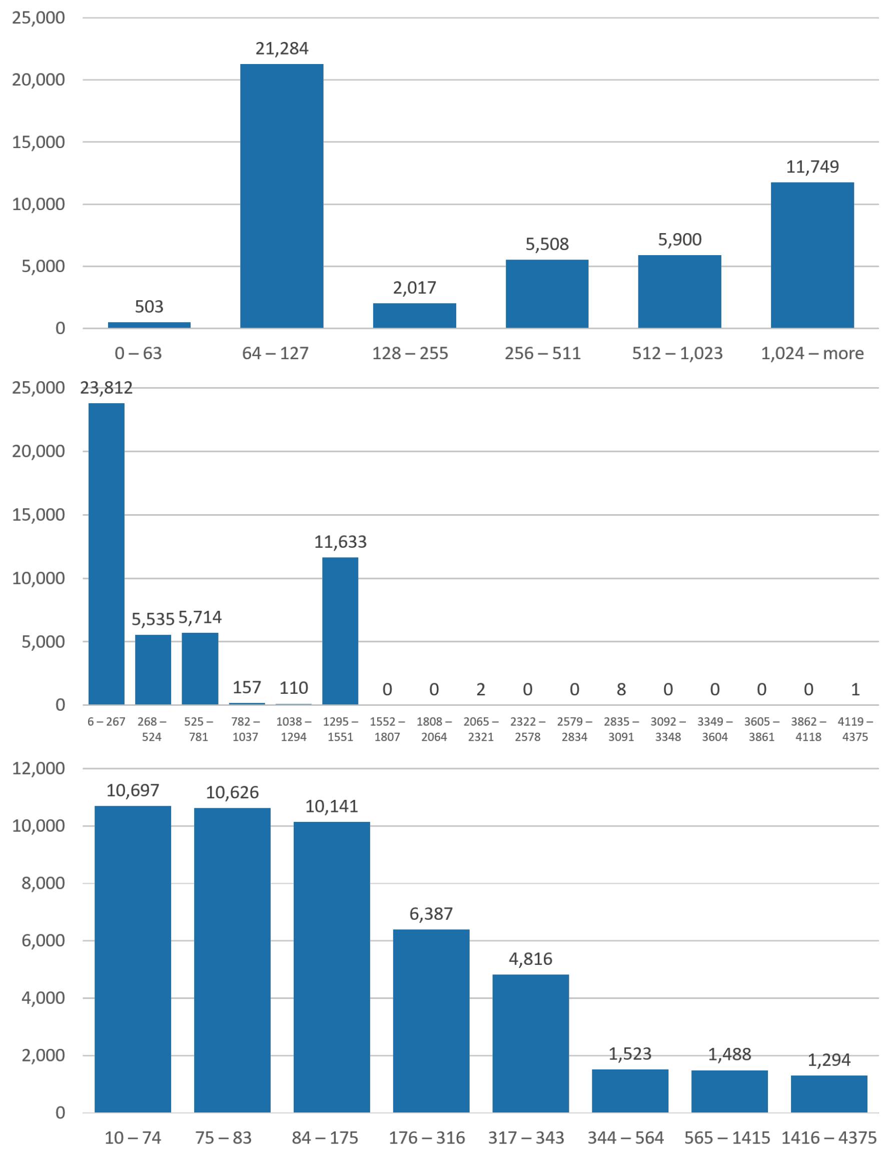

Figure 11: The histogram of one-payload length with BW (top), SEW (middle), and SEF (bottom). The x-axis is the bin range, while y-axis is the member count.

It is noteworthy that during the binning process with SEW, I found that several bins had zero members (see Figure 11). The results were due to SEW's nature, which grouped members by their equal range. From this point, if the bin's range was zero, no payload existed within the range. In the subsequent evaluation, I started with the implementation of BW, SEW, and SEF and then evaluated all four classes.

Figures 12, 13, and 14 showed the results of the multiclass target using LSTM and BiLSTM with BW, SEW, and SEF, respectively.

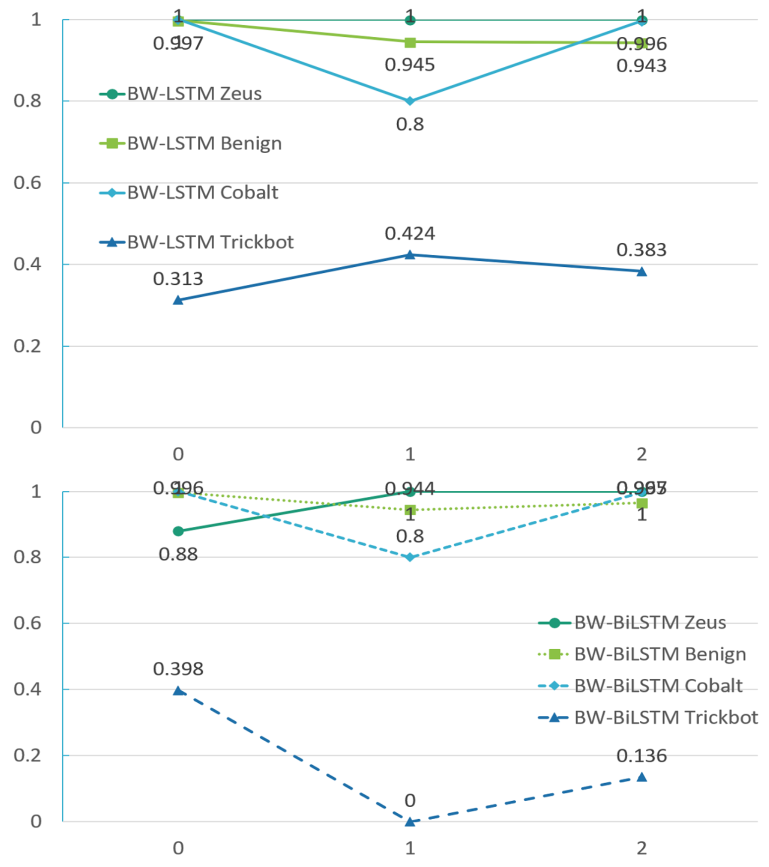

Figure 12: Multiclass target evaluation results using BW-LSTM (top) and BW-BiLSTM (bottom). The x-axis is the payload from zero (with no binning), one, and two or more, while the y-axis is the F1 score.

BW, SEW, and SEF were relatively stable for Zeus and Benign, whereas for Cobalt, I found a decrease in BW and SEF. Similarly, Trickbot also decreased in SEW and SEF. The results showed that binning had no significant effect on detection.

On this basis, I concluded that my methods can be used to detect encrypted malicious traffic based on my results. I could tell it from the evaluation using CBOW-LSTM, CBOW-BiLSTM, then using Skip-gram LSTM, and Skip-gram BiLSTM. I then continued the evaluation using three different kinds of binnings. First, I compared CBOW and Skip-gram starting with zero to two or more payloads, which increased performance. My method could detect Benign and Malicious with an average results of 0.999 using only TLS handshake information. Both models could learn to distinguish between Malicious and Benign. The TLS handshake consists of a cipher suite produced from the Malicious, which was different from the Benign, and the

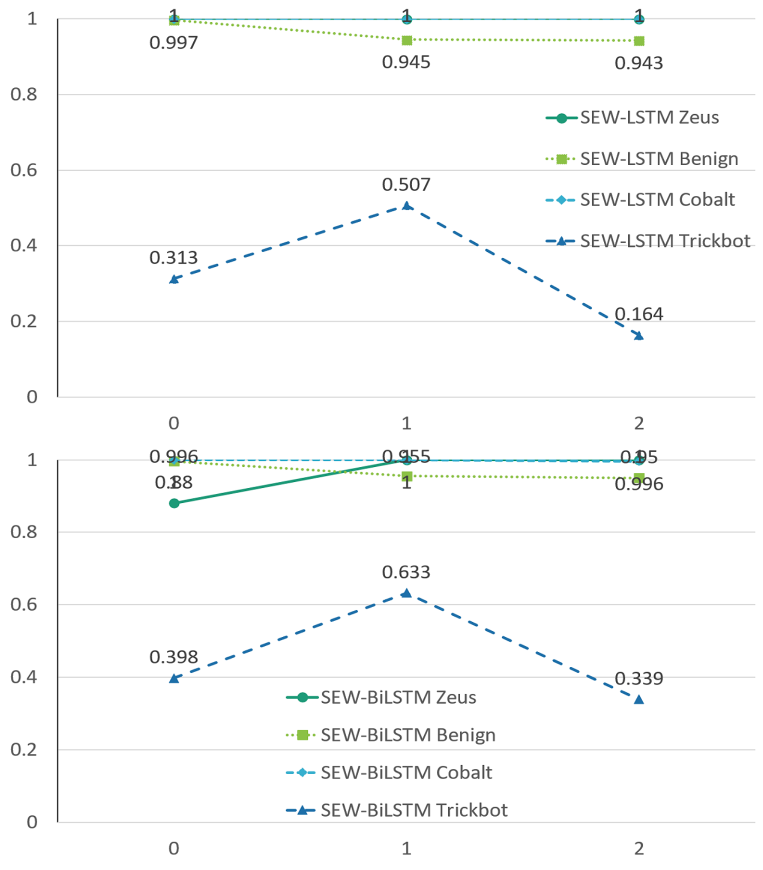

Figure 13: Multiclass target evaluation results using SEW-LSTM (top) and SEW-BiLSTM (bottom). The x-axis is the payload from zero (with no binning), one, and two or more, while the y-axis is the F1 score.

model could learn and detect data to distinguish information. The reason is that the TLS handshake from non-malicious traffic might be produced from the library of the OS. In contrast, the TLS handshake from the malicious traffic was produced by the malicious application [48]. From this point, I can generalize my model using a TLS handshake.

The average detection rate is 0.82 for the score total of Zeus, Benign, Cobalt, and Trickbot in combination. I then evaluated the method using three different binnings: BW, SEW, and SEF. Binning is a way to represent data that has many varieties in terms of the payloads. The goal of using these three kinds of binnings is to characterize the traffic better. However, the implementation of binning will reduce the vocabulary. In general, the implementation of binning has no significant effect on detecting malicious traffic.

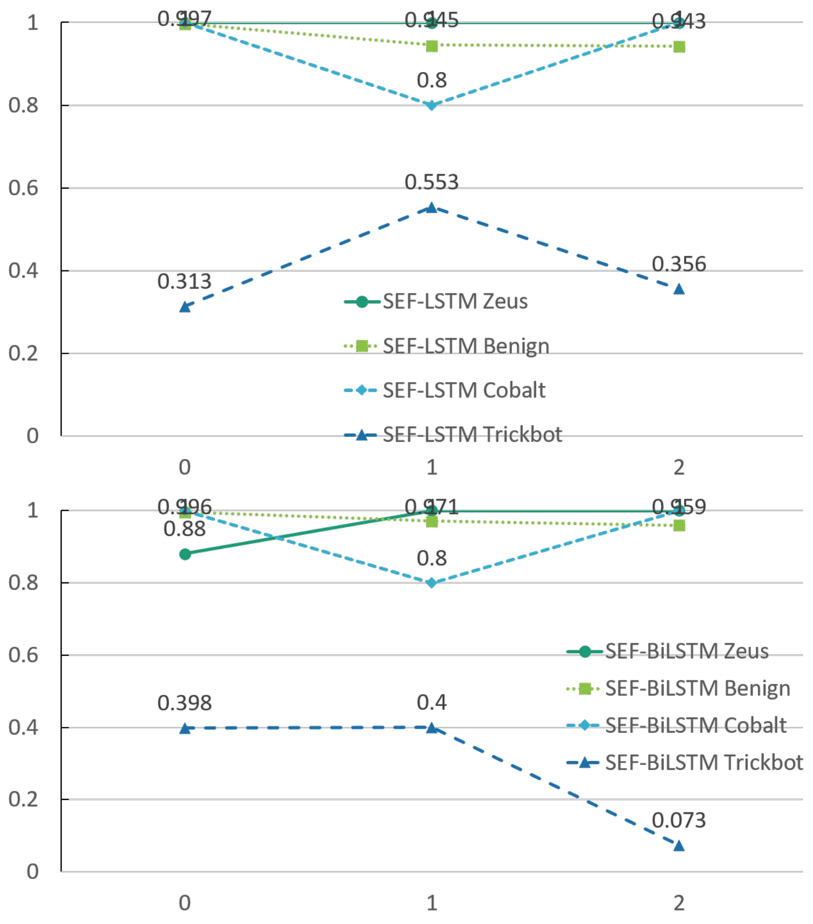

Figure 14: Multiclass target evaluation results using SEF-LSTM (top) and SEF-BiLSTM (bottom). The x-axis is the payload from zero (with no binning), one, and two or more, while the y-axis is the F1 score.

5.4 SUMMARY

This work contributed to the literature by providing a method for detecting malicious traffic using a TLS session before the conversation between client and server finishes. To achieve this, I extracted meaningful information from the TLS handshake and some portion from the TCP length of Application Data, which I call payload, and then created a corpus of traffic. I then predicted malicious traffic based on the TLS session and payload.

The advantage of TLS2Vec is that it can detect the binary target and the multiclass target using the payload. However, I also found that using my method has disadvantages over the minority class when combined with the binnings. This research revealed that it is sufficient to use only a TLS handshake to detect encrypted malicious traffic. The additional payloads from one and two or more can increase the performance. However, the effect depends on the TLS handshake.

6 CYBERSECURITY COMPLIANCE

Encrypted malicious applications pose a risk to organizations. In the previous chapter, my method focused on technical detection to handle encrypted malicious traffic. Besides the technical method regarding detecting encrypted malicious applications, additional research with broader view is necessary. In this regard, cybersecurity compliance can be measured to determine the organizational risk. According to [117], to obtain comprehensive system security, it is necessary to assess and evaluate all aspects of security from a computer network, application, operating system, database, physical, and environment. In this research, cybersecurity compliance measurement was carried out in the data center of the Indonesian local government at Sleman Regency.

6.1 ICT COMPLIANCE AT SLEMAN REGENCY

Sleman Regency is an Indonesian regency located in the north of the Yogyakarta Special Administrative Region, Indonesia. Figure 15 shows the position of Sleman Regency.

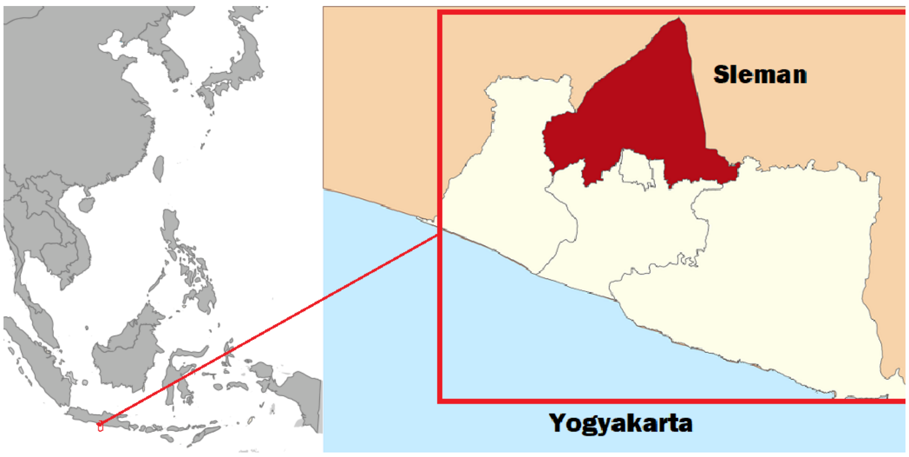

Figure 15: Sleman Regency in the Special Region of Yogyakarta [146].

Sleman is an advanced regency in terms of the use of information technology. The data center has more than 30 servers and several network equipment, which are managed by the ICT (Information and Communications Technology) unit. Each server has a different purpose, such as providing services for the public and internal departments, an email service for every local government staff, a hosting server for other departments and units, and a web server for public information regarding Sleman Regency. Several incidents occurred during the development of the infrastructure and the network systems, including the defacement of several subdomains, Distributed Denial of Service (DDoS) attacks on the VoIP server, and remote security holes in the server, where management authority is not from the ICT unit. However, the server is located in the data center of Sleman Regency.

The evaluation is typically only performed when an incident happens since this government organization does not have a standard evaluation planning and a standard concept in safety evaluation. One of the measurements to assess the condition of the governance data center is the COBIT (Control Objectives for Information and Related Technology) framework. The COBIT framework is widely adopted and used as one of the standards in researching the assets associated with information technology.

6.2 ALIGNING COBIT FRAMEWORK WITH CYBERSECURITY

An integrated framework is necessary to ensure the business goals are aligned with IT goals. This research was using COBIT 4.1 [19] and the sub-domain deliver and support (DS) 5, which ensure system security. IT managers or any part of the organization responsible for security should follow this guideline. The DS5 consists of eleven: management of IT security (DS5.1), IT security plan (DS5.2), identify management (DS5.3), user account management (DS5.4), security testing, surveillance, and monitoring (DS5.5), security incident definition (DS5.6), protection of security technology (DS5.7), cryptographic key management (DS5.8), malicious software prevention, detection, and correction (DS5.9), network security (DS5.10), exchange of sensitive data (DS5.11).

Each sub-domain can be implemented and evaluated within the organization. COBIT is not a technical framework. The general guideline shows that it can be integrated with other frameworks. Monitoring the infrastructure for security-related events can be applied by network monitoring tools such as NIDS. In this regard, the research focus on DS5.

6.3 METHODOLOGY

Two methods were applied in this research. The first method is checking the compliance using COBIT. The second is performing vulnerability assessment. COBIT has a particular method to calculate maturity model to see the extent of the security implementation in the data center by looking at the security management process. It is thus aimed at measuing the level of efficiency of security management. The standard score of the COBIT range between levels of 0 to 5.

6.3.1 *Questionnaires and Data Collection*

The survey was conducted in 2013. The technical documentation and information about applications installed on the servers were collected, such as IP addresses, type of operating system, UNIX-like or Windows, the version of the operating system, and what applications were installed on the servers. Questionnaires were compiled from the DS5 sub-domain and distributed to the staff. Table 15 shows a questionnaire sample.

Table 15: Ensure System Security (DS5)

No	Activity	Answer		
		0	1	2
1	The organization do not recognize the need of IT security.			
2	There are no clear accountability and responsibility for ensuring system security.			
3	IT security management measures are not implemented.			
4	There are no security reports and response related to IT security.			
5	There is a lack of recognizing the process of security administration system .			

Each staff should answer the questionnaire based on their knowledge. Based on [42, 101], the answer from column 0 means "No" with the value 0, column 1 means "Not Sure" with the value 0.5, and column 2 means "Yes" with the value 1.

6.3.2 *Data Reliability*

Cronbach Alpha was applied to the results of the questionnaire to ensure the data reliability using the scale from 0 to 1. If the result of the formula is equal to or more than 0.6, the data is considered reliable. The result from the scale is used to calculate the maturity model. Furthermore, a tool to calculate the maturity model and vulnerability assessment is needed.

6.3.3 *Implementation*

Implementation of the maturity model tool consisted of a reliability test and analysis of maturity. The reliability test determined the level of reliability of measurement before continuing to maturity model calculations. Meanwhile, the maturity score was generated after obtaining the questionnaire results from the respondents. The tool used CVE for the vulnerability knowledge base. Implementation of a vulnerability tool consisted of collecting binary versions from all servers. The tool parsed every binary application system from the servers. If the binary version matched the CVE list, then it was considered a vulnerability.

6.4 MATURITY MODEL ANALYSIS

Maturity model analysis was performed to assess the scale of capability and maturity. The maturity analysis has 6 levels. Level 0 is Non-existent: the department has not even recognized there is an issue to be addressed. Level 1 is Initial: there is evidence that the department has recognized the issues and need to be addressed. However, no standardized processes. The overall approach to management is disorganized. Level 2 is Repeatable: processes have developed to the stage where similar procedures are followed by different people undertaking the same task. There are no formal communications or standard procedures, and responsibility is left to the individual. Level 3 is the Defined process: procedures have been standardized and documented and communicated through training. The procedures are not sophisticated. Level 4 is Managed and measurable: the department monitors and measures compliance with procedures. Processes are under constant improvement and provide good practice. Level 5 is Optimised: all the processes have been refined to a level of good practice and continuous improvement. IT is used as an integrated way to automate the workflow.

The All Questions column is the total questions from each level. The Total Questions column is the total answer collected from all the staff of the department. The Maturity Value column is the results from Total Questions divided by All Questions. The Maturity Value Normalization column is each Maturity Value divided by a total number of Maturity Value from all levels. The Maturity Model column is level times Maturity Value Normalization. Table 16 shows that the value of the maturity model is 2.852. Based on the ISACA and IT Governance Insitute

(ITGI), the value of 2.852 is between the value 2.51 and 3.50 from the maturity level criteria indicating that at level 2. The results show that there was no security plan, no security testing, surveillance, and monitoring, and malicious applications prevention and detection such as NIDS was never found.

Table 16: The results of maturity model

Maturity Level	All Questions	Total Question	Maturity Value	Maturity Value Normalization	Maturity Model
Non-Existent	5	7.5	1.5	0.044	0
Initial	6	35.5	5.917	0.175	0.175
Repeatable	8	60	7.5	0.222	0.444
Defined Process	7	46.5	6.643	0.197	0.591
Managed and Measureable	12	66	5.5	0.163	0.652
Optimised	10	67	6.7	0.198	0.99
Total			33.76	1	2.852

6.5 SUMMARY

The result of the experiment shows that the maturity model from DS5 was 2.852 for the Department of Transportation, Communication, and Information. The maturity level reached by the ICT unit was at level 2 or Repeatable for current conditions meaning there are no standard procedures, and responsibility is left to the individual. Furthermore, the potential risk from the results was no clear information, planning, or implementation of security tools such as NIDS.

7 CONCLUSION

In this dissertation, my research theme is NIDS focusing on detecting malicious traffic in encrypted network. Until nowadays, many researchers still used the obsolete dataset which is irrelevant and cannot be used for detecting new attacks. I have showed this in the previous chapter that I have to remove irrelevant data which is unusable or is no longer categorized as an attack, such as neptune, teardrop, satan, smurf, pod, back, ftp_write, multihop, imap, phf, land, and load_module. The other issue was several attack data such as load_module, ftp_write, multihop, phf, perl, and spy were underrepresented. This will have an impact on accuracy. To handle the irrelevant data, I proposed modifications of the dataset using three methods and used GA to find the optimal features.

Because network traffic changes over time, the publicly available datasets represented by the current network situation are essential. This week's traffic might be different from the previous month. That is why the concept of up-to-date datasets is necessary. The missing requirement from the current dataset is encryption information and a practical method to implement it. I proposed two requirements for building a dataset: content and process requirements. These two are a foundation for building high-quality data. Through a better method of constructing a dataset, a security researcher might produce their dataset and improve it. In the future, I would like to extend HIKARI-2021 dataset observation with the background traffic and add an evaluation. Because background traffic is uncertain and not labeled in the data, the possible approach for evaluation is using machine learning with unsupervised learning. Furthermore, I would like to compare performance of the existing datasets and proceed with the analysis of application identification, which is important because malicious traffic may be disguised using reserved ports to bypass firewalls or IDS and blend with normal network activity.

Nowadays, many malicious applications use TLS to hide their malicious activities. I found that some applications, such as Zeus, and Trickbot still become a threat, especially when the author of the application shares the source code with the public.

The method I proposed, TLS2Vec, provided a promising result in detecting encrypted malicious traffic with higher accuracy, especially those malicious applications that use their own TLS library instead of a library available on the host. However, the accuracy of detection is dependent on the TLS handshake. I found that the malicious applications used short sessions.

TLS2Vec cannot run independently and can be used as an ensemble with other methods such as host behavior analysis. During the analysis, I encountered a class imbalance within the data for the multiclass targets with one class being a small minority compared to others such as Trickbot. Having a minority class will affect in the performance of detection. In a non-TLS2Vec, I did not use Word2Vec and removed the symbols that represent some features such as cipher, ext_len, elliptic_curves, ec_point_formats, and len. I also found that TLS2Vec performed better than Non-TLS2Vec, which use neither symbols nor Word2Vec embedding. Binning the payload length in order to reduce the vocabulary does not improve the performance, but rather gave worse performance for the minority class. Furthermore, I would like to highlight the points that differentiate my solution from others. Using my method, TLS2Vec could detect encrypted malicious traffic by using the TLS handshake and a portion of the payload. As a result, I did not need to dissect the traffic to maintain the user's privacy. There are three areas in this work that might be worth exploring. In terms of traffic analysis and NIDS, malicious applications used shorter sessions. However, does it apply to the other than Zeus, Cobalt, and Trickbot? Next is related to the possibility of operating NIDS in real time given the encrypted traffic. In terms of Deep Learning, future research is suggested to address the possibility to use the corpus to detect the variants of malicious traffic, such as Zeus.

Further research may be extended from this research by adding an analysis of the compliance of cybersecurity. In chapter 6 the assessment was performed at data center in Sleman Regency. The study aimed to assess the current condition of the data center and assess the maturity level of security as well as provide solutions in particular on IT security. Identifying weaknesses can help evaluate and provide solutions for a better future. I found that on a scale of 0 to 5, the data center is at level 2 or Repeatable. Since there was no comprehensive planning, I found that there was no implementation of NIDS to monitor malicious traffic. Hence, I concluded that further maturity model testing should be carried out periodically. In the management scope, information regarding responsibility and accountability must also be clear. Then, the NIDS should be installed in the gateway to monitor malicious traffic.

NIDS will remain as a hot topic in the future given the increasingly encrypted traffic and the progressively sophisticated attacks. In the current situation of the increasingly complex cyber-attacks, a general-purpose NIDS is incapable of solving specific problems. As an example, IoT-based networks are subject to different kinds of attacks. Thus, in the future, researchers certainly need new techniques, detection algorithms, and a new way to solve several problems in privacy, the massive data coming from multiple sensors, and the collaboration of multiple threat intelligence information.

8 LIST OF PAPERS AND PRESENTATION

8.1 PEER-REVIEWED JOURNALS

(1) Andrey Ferriyan, Achmad Husni Thamrin, Keiji Takeda, Jun Murai. ***Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic***. MDPI Applied Sciences Special Issue "Machine Learning for Cybersecurity Threats, Challenges, and Opportunities II". 2021; 11(17):7868.
DOI: https://doi.org/10.3390/app11177868.

(2) Andrey Ferriyan, Achmad Husni Thamrin, Keiji Takeda, Jun Murai. ***Encrypted Malicious Traffic Detection Based on Word2Vec***. MDPI Electronics Special Issue "Design of Intelligent Intrusion Detection Systems". 2022, 11(5), 679. https://doi.org/10.3390/electronics11050679.

(3) Andrey Ferriyan, Jazi Eko Istiyanto. ***Data Center Governance Information Security Compliance Assessment Based on the Cobit Framewok***. International Journal of Advanced Computer Science and Applications (IJACSA), Vol 6 Issue 2, February 2015.
DOI: https://dx.doi.org/10.14569/IJACSA.2015.060205.

8.2 INTERNATIONAL CONFERENCES

8.2.1 *Oral Presentation*

(1) Andrey Ferriyan, Achmad Husni Thamrin, Keiji Takeda, Jun Murai. ***Feature Selection Using Genetic Algorithm to Improve Classification in Network Intrusion Detection System***. International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC) 2017. September 26-27, 2017, 4 pages, English.
DOI: https://dx.doi.org/10.1109/KCIC.2017.8228458.

(2) Andrey Ferriyan, Achmad Husni Thamrin, Keiji Takeda, Osamu Nakamura, Jun Murai. ***Applying Convolutional Neural Network to Improve Malware Traffic Detection***.

11th Japan Muslim Researchers Interdisciplinary Symposium 2019 (11th JMRIS). July 20, 2019.

BIBLIOGRAPHY

[1] *2019 Internet Security Threat Report*. Online; retrieved on December 12, 2019. 2019.

[2] Iftikhar Ahmad, Muhammad Hussain, Abdullah Alghamdi, and Abdulhameed Alelaiwi. "Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components." In: *Neural Computing and Applications* 24.7 (2014), pp. 1671–1682. ISSN: 1433-3058. DOI: 10.1007/s00521-013-1370-6. URL: https://doi.org/10.1007/s00521-013-1370-6.

[3] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. "A survey of network anomaly detection techniques." In: *Journal of Network and Computer Applications* 60 (2016), pp. 19–31. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2015.11.016. URL: https://www.sciencedirect.com/science/article/pii/S1084804515002891.

[4] Eugene Albin and Neil C. Rowe. "A Realistic Experimental Comparison of the Suricata and Snort Intrusion-Detection Systems." In: *2012 26th International Conference on Advanced Information Networking and Applications Workshops*. 2012, pp. 122–127. DOI: 10.1109/WAINA.2012.29.

[5] *Alexa*. https://www.alexa.com. Online; retrieved on May 14, 2021. 2021.

[6] Christopher Allen and Tim Dierks. *The TLS Protocol Version 1.0*. RFC 2246. 1999. DOI: 10.17487/RFC2246. URL: https://rfc-editor.org/rfc/rfc2246.txt.

[7] NG Bhuvaneswari Amma and S Selvakumar. "A statistical class center based triangle area vector method for detection of denial of service attacks." In: *Cluster Computing* 24.1 (2021), pp. 393–415.

[8] Payam Vahdani Amoli and Timo Hämäläinen. "A real time unsupervised NIDS for detecting unknown and encrypted network attacks in high speed network." In: *2013 IEEE International Workshop on Measurements & Networking (M&N)*. IEEE. 2013, pp. 149–154.

[9] Blake Anderson and David McGrew. "Identifying Encrypted Malware Traffic with Contextual Flow Data." In: *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*. AISec '16. Vienna, Austria: Association for Computing Machinery, 2016, 35–46. ISBN: 9781450345736. DOI: 10.1145/2996758.2996768. URL: https://doi.org/10.1145/2996758.2996768.

[10] Blake Anderson and David McGrew. "Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity." In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '17. Halifax, NS, Canada: Association for Computing Machinery, 2017, 1723–1732. ISBN: 9781450348874. DOI: 10.1145/3097983.3098163. URL: https://doi.org/10.1145/3097983.3098163.

[11] Adam J. Aviv and Andreas Haeberlen. "Challenges in Experimenting with Botnet Detection Systems." In: *4th Workshop on Cyber Security Experimentation and Test (CSET 11)*. San Francisco, CA: USENIX Association, Aug. 2011. URL: https://www.usenix.org/conference/cset11/challenges-experimenting-botnet-detection-systems.

[12] Ji-Won Baek and Kyung-Yong Chung. "Multimedia recommendation using Word2Vec-based social relationship mining." In: *Multimedia Tools and Applications* (2020), pp. 1–17. DOI: https://doi.org/10.1007/s11042-019-08607-9.

[13] Rafael Ramos Regis Barbosa, Ramin Sadre, Aiko Pras, and Remco van de Meent. "Simpleweb/university of twente traffic traces data repository." In: *Centre for Telematics and Information Technology, University of Twente* (2010). https://research.utwente.nl/en/publications/simplewebuniversity-of-twente-traffic-traces-data-repository.

[14] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors." In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, 2014, pp. 238–247. DOI: 10.3115/v1/P14-1023. URL: https://aclanthology.org/P14-1023.

[15] George Bissias, Brian Neil Levine, Marc Liberatore, and Swagatika Prusty. "Forensic identification of anonymous sources in oneswarm." In: *IEEE Transactions on Dependable and Secure Computing* 14.6 (2015), pp. 620–632. DOI: https://doi.org/10.1109/TDSC.2015.2497706.

[16] Elias Bou-Harb, Mourad Debbabi, and Chadi Assi. "Cyber Scanning: A Comprehensive Survey." In: *IEEE Communications Surveys Tutorials* 16.3 (2014), pp. 1496–1519. DOI: 10.1109/SURV.2013.102913.00020.

[17] Ismail Butun, Salvatore D. Morgera, and Ravi Sankar. "A Survey of Intrusion Detection Systems in Wireless Sensor Networks." In: *IEEE Communications Surveys Tutorials* 16.1 (2014), pp. 266–282. DOI: 10.1109/SURV.2013.050113.00191.

[18] *CAIDA datasets*. https://www.caida.org/catalog/datasets/completed-datasets/. Online; retrieved on May 10, 2021. 2021.

[19] *COBIT 4.1*. https://www.isaca.org/resources/cobit. Online; retrieved on July 18, 2021. 2021.

[20] Franco Callegati, Walter Cerroni, and Marco Ramilli. "Man-in-the-Middle Attack to the HTTPS Protocol." In: *IEEE Security & Privacy* 7.1 (2009), pp. 78–81.

[21] Zigang Cao, Gang Xiong, Yong Zhao, Zhenzhen Li, and Li Guo. "A survey on encrypted traffic classification." In: *International Conference on Applications and Techniques in Information Security*. Springer. 2014, pp. 73–81. DOI: https://doi.org/10.1007/978-3-662-45670-5_8.

[22] Milan Čermák and Pavel Čeleda. "Detecting Advanced Network Threats Using a Similarity Search." In: *IFIP International Conference on Autonomous Infrastructure, Management and Security*. Springer. 2016, pp. 137–141. DOI: https://doi.org/10.1007/978-3-319-39814-3_14.

[23] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly Detection: A Survey." In: *ACM Comput. Surv.* 41.3 (2009). ISSN: 0360-0300. DOI: 10.1145/1541880.1541882. URL: https://doi.org/10.1145/1541880.1541882.

[24] Zouhair Chiba, Noureddine Abghour, Khalid Moussaid, Amina El Omri, and Mohamed Rida. "A survey of intrusion detection systems for cloud computing environment." In: *2016 International Conference on Engineering MIS (ICEMIS)*. 2016, pp. 1–13. DOI: 10.1109/ICEMIS.2016.7745295.

[25] Carlton Chu, Ai-Ling Hsu, Kun-Hsien Chou, Peter Bandettini, and ChingPo Lin. "Does feature selection improve classification accuracy? Impact of sample size and feature selection on classification using anatomical magnetic resonance images." In: *NeuroImage* 60.1 (2012), pp. 59–70. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2011.11.066. URL: https://www.sciencedirect.com/science/article/pii/S1053811911013486.

[26] Ching-Hua Chuan, Kat Agres, and Dorien Herremans. "From context to concept: exploring semantic relationships in music with word2vec." In: *Neural Computing and Applications* 32.4 (2020), pp. 1023–1036. DOI: https://doi.org/10.1007/s00521-018-3923-1.

[27] Carlos Garcia Cordero, Emmanouil Vasilomanolakis, Aidmar Wainakh, Max Mühlhäuser, and Simin Nadjm-Tehrani. "On generating network traffic datasets with synthetic attacks for intrusion detection." In: *ACM Transactions on Privacy and Security (TOPS)* 24.2 (2021), pp. 1–39. DOI: https://doi.org/10.1145/3424155.

[28] Manuel Crotti, Francesco Gringoli, Paolo Pelosato, and Luca Salgarelli. "A statistical approach to IP-level classification of network traffic." In: *2006 IEEE International Conference on Communications*. Vol. 1. 2006, pp. 170–176. DOI: 10.1109/ICC.2006.254723.

[29] Rui Dai, Chuan Gao, Bo Lang, Lixia Yang, Hongyu Liu, and Shaojie Chen. "SSL Malicious Traffic Detection Based On Multi-View Features." In: *Proceedings of the 2019 the 9th International Conference on Communication and Network Security*. ICCNS 2019. Chongqing, China: Association for Computing Machinery, 2019, 40–46. ISBN: 9781450376624. DOI: 10.1145/3371676.3371697. URL: https://doi.org/10.1145/3371676.3371697.

[30] *Daniel Miessler 10k most common credentials*. https://github.com/danielmiessler. Online; retrieved on May 14, 2021. 2021.

[31] Kenneth A. De Jong and William M. Spears. "An analysis of the interacting roles of population size and crossover in genetic algorithms." In: *Parallel Problem Solving from Nature*. Ed. by Hans-Paul Schwefel and Reinhard Männer. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 38–47. ISBN: 978-3-540-70652-6.

[32] Michael J De Lucia and Chase Cotton. "Identifying and detecting applications within TLS traffic." In: *Cyber Sensing 2018*. Vol. 10630. International Society for Optics and Photonics. 2018, 106300U. DOI: https://doi.org/10.1117/12.2305256.

[33] D.E. Denning. "An Intrusion-Detection Model." In: *IEEE Transactions on Software Engineering* SE-13.2 (1987), pp. 222–232. DOI: 10.1109/TSE.1987.232894.

[34] Tim Dierks and Christopher Allen. *Rfc5246: The Transport Layer Security (TLS) Protocol Version 1.2*. RFC 5246. RFC Editor, Oct. 2008. URL: https://www.rfc-editor.org/rfc/rfc4180.txt.

[35] Yalei Ding and Yuqing Zhai. "Intrusion detection system for NSL-KDD dataset using convolutional neural networks." In: *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence*. 2018, pp. 81–85. DOI: https://doi.org/10.1145/3297156.3297230.

[36] *Droopescan*. https://github.com/droope/droopescan. Online; retrieved on April 30, 2021. 2021.

[37] Nathan Eagle and Alex Sandy Pentland. "Reality mining: sensing complex social systems." In: *Personal and ubiquitous computing* 10.4 (2006), pp. 255–268. DOI: https://doi.org/10.1007/s00779-005-0046-3.

[38] Saber M. Elsayed, Ruhul A. Sarker, and Daryl L. Essam. "A new genetic algorithm for solving optimization problems." In: *Engineering Applications of Artificial Intelligence* 27 (2014), pp. 57–69. ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2013.09.013. URL: https://www.sciencedirect.com/science/article/pii/S0952197613001875.

[39] J. Espenschied and A. Gunn. *Threat Genomics*. Tech. rep. SecurityMetrics.org, Oct. 2012. URL: http://www.securitymetrics.org/attachments/Metricon-7-paper-Threat-Genomics-EspenschiedGunn-2012.pdf.

[40] Loic Etienne. *Malicious traffic detection in local networks with snort*. 2009.

[41] Jinliang Fan, Jun Xu, Mostafa H. Ammar, and Sue B. Moon. "Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme." In: *Computer Networks* 46.2 (2004), pp. 253–272. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2004.03.033. URL: https://www.sciencedirect.com/science/article/pii/S1389128604001197.

[42] Andrey Ferriyan. "Data Center Governance Information Security Compliance Assessment Based on The Cobit Framework (Case Study: The Sleman Regency Data Center)." MA thesis. Universitas Gadjah Mada, 2014.

[43] Andrey Ferriyan, Achmad Husni Thamrin, Keiji Takeda, and Jun Murai. *HIKARI-2021: Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic*. Version 1.2.0. June 2021. DOI: https://doi.org/10.5281/zenodo.5111946. URL: https://doi.org/10.5281/zenodo.5111946.

[44] *Firefox Telemetry*. https://docs.telemetry.mozilla.org/datasets/other/ssl/reference.html. Online; retrieved on December 17, 2021. 2021.

[45] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. "Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking." In: *Proceedings of the 6th International COnference*. 2010, pp. 1–12. DOI: https://doi.org/10.1145/1921168.1921179.

[46] *Fukuda Lab Mawi Archive*. https://fukuda-lab.org/mawilab. Online; retrieved on May 10, 2021. 2021.

[47] Ying Gao, Hongrui Wu, Binjie Song, Yaqia Jin, Xiongwen Luo, and Xing Zeng. "A distributed network intrusion detection system for distributed denial of service attacks in vehicular ad hoc network." In: *IEEE Access* 7 (2019), pp. 154560–154571. DOI: https://doi.org/10.1109/ACCESS.2019.2948382.

[48] Gibran Gómez, Platon Kotzias, Matteo Dell'Amico, Leyla Bilge, and Juan Caballero. "Unsupervised Detection and Clustering of Malicious TLS Flows." In: *arXiv preprint arXiv:2109.03878* (2021).

[49] Eric L Goodman, Chase Zimmerman, and Corey Hudson. "Packet2Vec: Utilizing Word2Vec for feature extraction in packet data." In: *arXiv preprint arXiv:2004.14477* (2020).

[50] *Google Transparency Report*. https://transparencyreport.google.com/https/overview?hl=en. Online; retrieved on December 2, 2021. 2021.

[51] John J. Grefenstette. "Optimization of Control Parameters for Genetic Algorithms." In: *IEEE Transactions on Systems, Man, and Cybernetics* 16.1 (1986), pp. 122–128. DOI: 10.1109/TSMC.1986.289288.

[52] Steffen Haas. "Security Monitoring and Alert Correlation for Network Intrusion Detection." https://ediss.sub.uni-hamburg.de/handle/ediss/8930. PhD thesis. Staats- und Universitätsbibliothek Hamburg Carl von Ossietzky, 2020.

[53] Mounir Hafsa and Farah Jemili. "Comparative study between big data analysis techniques in intrusion detection." In: *Big Data and Cognitive Computing* 3.1 (2019), p. 1. DOI: https://doi.org/10.3390/bdcc3010001.

[54] Mark A. Hall. *Correlation-based feature selection for machine learning*. Tech. rep. 1998.

[55] Cristian Hesselman, Merike Kaeo, Lyman Chapin, Kimberly Claffy, Mark Seiden, Danny McPherson, Dave Piscitello, Andrew McConachie, Tim April, Jacques Latour, et al. "The DNS in IoT: Opportunities, Risks, and Challenges." In: *IEEE Internet Computing* 24.4 (2020), pp. 23–32. DOI: https://doi.org/10.1109/MIC.2020.3005388.

[56] Nico Hinze, Marcin Nawrocki, Mattijs Jonker, Alberto Dainotti, Thomas C Schmidt, and Matthias Wählisch. "On the potential of BGP flowspec for DDoS mitigation at two sources: ISP and IXP." In: *Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos*. 2018, pp. 57–59. DOI: https://doi.org/10.1145/3234200.3234209.

[57] Cheng Huang, Shuang Hao, Luca Invernizzi, Jiayong Liu, Yong Fang, Christopher Kruegel, and Giovanni Vigna. "Gossip: Automatically Identifying Malicious Domains from Mailing List Discussions." In: *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ASIA CCS '17. Abu Dhabi, United Arab Emirates: Association for Computing Machinery, 2017,

494–505. ISBN: 9781450349444. DOI: 10.1145/3052973.3053017. URL: https://doi.org/10.1145/3052973.3053017.

[58] Eric M Hutchins, Michael J Cloppert, Rohan M Amin, et al. "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains." In: *Leading Issues in Information Warfare & Security Research* 1.1 (2011), p. 80.

[59] *IMPACT Cyber Trust*. https://www.impactcybertrust.org/. Online; retrieved on May 10, 2021. 2021.

[60] *Jason Stroschein public github malware samples*. https://github.com/jstrosch/malware-samples. Online; retrieved on May 10, 2021. 2021.

[61] Mattijs Jonker, Alistair King, Johannes Krupp, Christian Rossow, Anna Sperotto, and Alberto Dainotti. "Millions of targets under attack: a macroscopic characterization of the DoS ecosystem." In: *Proceedings of the 2017 Internet Measurement Conference*. 2017, pp. 100–113. DOI: https://doi.org/10.1145/3131365.3131383.

[62] *Joomscan*. https://github.com/OWASP/joomscan/releases. Online; retrieved on April 30, 2021. 2021.

[63] Sydney M Kasongo and Yanxia Sun. "Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset." In: *Journal of Big Data* 7.1 (2020), pp. 1–20. DOI: https://doi.org/10.1186/s40537-020-00379-6.

[64] Sanmeet Kaur and Maninder Singh. "Automatic attack signature generation systems: A review." In: *IEEE Security & Privacy* 11.6 (2013), pp. 54–61. DOI: https://doi.org/10.1109/MSP.2013.51.

[65] A. Kenyon, L. Deka, and D. Elizondo. "Are public intrusion datasets fit for purpose characterising the state of the art in intrusion event datasets." In: *Computers & Security* 99 (2020), p. 102022. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2020.102022. URL: https://www.sciencedirect.com/science/article/pii/S0167404820302959.

[66] Jawad Khalife, Amjad Hajjar, and Jesus Diaz-Verdejo. "A Multilevel Taxonomy and Requirements for an Optimal Traffic-Classification Model." In: *Netw.* 24.2 (2014), 101–120. ISSN: 0028-3045. DOI: 10.1002/nem.1855. URL: https://doi.org/10.1002/nem.1855.

[67] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. "Survey of intrusion detection systems: techniques, datasets and challenges." In: *Cybersecurity* 2.1 (2019), p. 20. ISSN: 2523-3246. DOI: 10.1186/s42400-019-0038-7. URL: https://doi.org/10.1186/s42400-019-0038-7.

[68] Jinoh Kim, Caitlin Sim, and Jinhwan Choi. "Generating labeled flow data from MAWILab traces for network intrusion detection." In: *Proceedings of the ACM Workshop on Systems and Network Telemetry and Analytics*. 2019, pp. 45–48. DOI: https://doi.org/10.1145/3322798.3329251.

[69] Paul Krystosek, Nancy M Ott, Geoffrey Sanders, and Timothy Shimeall. *Network Traffic Analysis with SiLK*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA PITTSBURGH United States, 2019.

[70] Deepak Kshirsagar and Sandeep Kumar. "An ensemble feature reduction method for web-attack detection." In: *Journal of Discrete Mathematical Sciences and Cryptography* 23.1 (2020), pp. 283–291. DOI: https://doi.org/10.1080/09720529.2020.1721861.

[71] Deepak Kshirsagar and Sandeep Kumar. "An efficient feature reduction method for the detection of DoS attack." In: *ICT Express* (2021). DOI: https://doi.org/10.1016/j.icte.2020.12.006.

[72] Vikash Kumar, Ayan Kumar Das, and Ditipriya Sinha. "Statistical analysis of the UNSW-NB15 dataset for intrusion detection." In: *Computational Intelligence in Pattern Recognition*. Springer, 2020, pp. 279–294. DOI: https://doi.org/10.1007/978-981-13-9042-5_24.

[73] *Kyoto Dataset*. http://www.takakura.com/Kyoto_data. Online; retrieved on May 10, 2021. 2021.

[74] Arash Habibi Lashkari, Gerard Draper-Gil, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. "Characterization of tor traffic using time based features." In: *ICISSp*. 2017, pp. 253–262. DOI: https://doi.org/10.5220/0006105602530262.

[75] Sam Leroux, Steven Bohez, Pieter-Jan Maenhaut, Nathan Meheus, Pieter Simoens, and Bart Dhoedt. "Fingerprinting encrypted network traffic types using machine learning." In: *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*. 2018, pp. 1–5. DOI: 10.1109/NOMS.2018.8406218.

[76] *Lets Encrypt Status Report*. https://letsencrypt.org/stats. Online; retrieved on December 17, 2021. 2021.

[77] Jieling Li, Hao Zhang, and Zhiqiang Wei. "The Weighted Word2vec Paragraph Vectors for Anomaly Detection Over HTTP Traffic." In: *IEEE Access* 8 (2020), pp. 141787–141798. DOI: 10.1109/ACCESS.2020.3013849.

[78] Lixiang Li, Hao Zhang, Haipeng Peng, and Yixian Yang. "Nearest neighbors based density peaks approach to intrusion detection." In: *Chaos, Solitons & Fractals* 110 (2018), pp. 33–40. ISSN: 0960-0779. DOI: https://doi.org/10.1016/j.chaos.2018.03.010. URL: https://www.sciencedirect.com/science/article/pii/S0960077918301073.

[79] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. "Intrusion detection system: A comprehensive review." In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 16–24. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2012.09.004. URL: https://www.sciencedirect.com/science/article/pii/S1084804512001944.

[80] Richard P Lippmann, David J Fried, Isaac Graf, Joshua W Haines, Kristopher R Kendall, David McClung, Dan Weber, Seth E Webster, Dan Wyschogrod, Robert K Cunningham, et al. "Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation." In: *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*. Vol. 2. IEEE. 2000, pp. 12–26. DOI: https://doi.org/10.1109/DISCEX.2000.821506.

[81] Jinxin Liu, Burak Kantarci, and Carlisle Adams. "Machine learning-driven intrusion detection for contiki-NG-based IoT networks exposed to NSL-KDD dataset." In: *Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning*. 2020, pp. 25–30. DOI: https://doi.org/10.1145/3395352.3402621.

[82] Michael J. De Lucia and Chase Cotton. "Identifying and detecting applications within TLS traffic." In: *Cyber Sensing 2018*. Ed. by Igor V. Ternovskiy and Peter Chin. Vol. 10630. International Society for Optics and Photonics. SPIE, 2018, pp. 179 –190. DOI: 10.1117/12.2305256. URL: https://doi.org/10.1117/12.2305256.

[83] Chaochao Luo, Le Wang, and Hui Lu. "Analysis of LSTM-RNN based on attack type of kdd-99 dataset." In: *International Conference on Cloud Computing and Security*. Springer. 2018, pp. 326–333. DOI: https://doi.org/10.1007/978-3-030-00006-6_29.

[84] Philipp M Lutscher, Nils B Weidmann, Margaret E Roberts, Mattijs Jonker, Alistair King, and Alberto Dainotti. "At home and abroad: The use of denial-of-service attacks during elections in nondemocratic regimes." In: *Journal of Conflict Resolution* 64.2-3 (2020), pp. 373–401. DOI: https://doi.org/10.1177/0022002719861676.

[85] *MITRE*. https://cve.mitre.org/. Online; retrieved on January 1, 2017. 2017.

[86] Gabriel Maciá-Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón. "UGR '16: A new dataset for the evaluation of cyclostationarity-based network IDSs." In: *Computers & Security* 73 (2018), pp. 411–424. DOI: https://doi.org/10.1016/j.cose.2017.11.004.

[87] *Malware Capture Facility Project*. https://mcfp.felk.cvut.cz/publicDatasets/datasets.html. Online; retrieved on May 10, 2021. 2021.

[88] *Mandiant Targeted Attack Lifecycle*. https://www.mandiant.com/resources/targeted-attack-lifecycle. Online; retrieved on January 12, 2022. 2022.

[89] David J Marchette. "A Statistical Method for Profiling Network Traffic." In: *Workshop on Intrusion Detection and Network Monitoring*. 1999, pp. 119–128.

[90] Verónica Mateos, Víctor A. Villagrá, Francisco Romero, and Julio Berrocal. "Definition of response metrics for an ontology-based Automated Intrusion Response Systems." In: *Computers & Electrical Engineering* 38.5 (2012). Special issue on Recent Advances in Security and Privacy in Distributed Communications and Image processing, pp. 1102–1114. ISSN: 0045-7906. DOI: https://

doi.org/10.1016/j.compeleceng.2012.06.001. URL: https://www.sciencedirect.com/science/article/pii/S0045790612001103.

[91] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[92] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed Representations of Words and Phrases and their Compositionality." In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Vol. 26. Curran Associates, Inc., 2013. URL: https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

[93] Nour Moustafa and Jill Slay. "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set." In: *Information Security Journal: A Global Perspective* 25.1-3 (2016), pp. 18–31. DOI: https://doi.org/10.1080/19393555.2015.1125974.

[94] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. "Deepcorr: Strong flow correlation attacks on tor using deep learning." In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2018, pp. 1962–1976. DOI: https://doi.org/10.1145/3243734.3243824.

[95] *National Cyber Security Centre: Stages of an attack*. https://www.ncsc.gov.uk/information/how-cyber-attacks-work. Online; retrieved on January 12, 2022. 2022.

[96] Yoav Nir, Simon Josefsson, and Manuel Pégourié-Gonnard. *Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier*. RFC 8422. 2018. DOI: 10.17487/RFC8422. URL: https://rfc-editor.org/rfc/rfc8422.txt.

[97] D. Ourston, S. Matzner, W. Stump, and B. Hopkins. "Applications of hidden Markov models to detecting multi-stage network attacks." In: *36th Annual Hawaii International Conference on System Sciences, 2003. Proceedings of the*. 2003, 10 pp.–. DOI: 10.1109/HICSS.2003.1174909.

[98] Atilla Özgür and Hamit Erdem. "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015." In: *PeerJ Preprints* 4 (2016), e1954v1. DOI: https://doi.org/10.7287/peerj.preprints.1954v1.

[99] ANDERSON J. P. "Computer security threat monitoring and surveillance." In: *Technical Report, James P. Anderson Company* (1980). URL: https://cir.nii.ac.jp/crid/1573950399661362176.

[100] Eva Papadogiannaki, Dimitris Deyannis, and Sotiris Ioannidis. "Head(er)Hunter: Fast Intrusion Detection using Packet Metadata Signatures." In: *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. 2020, pp. 1–6. DOI: 10.1109/CAMAD50429.2020.9209308.

[101] Andrea Pederiva. "The COBIT® maturity model in a vendor evaluation case." In: *Information Systems Control Journal* 3 (2003), pp. 26–29.

[102] Muhammad Shakil Pervez and Dewan Md Farid. "Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs." In: *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)* (2014), pp. 1–6.

[103] Paul Prasse, Lukáš Machlica, Tomáš Pevnỳ, Jiří Havelka, and Tobias Scheffer. "Malware detection by analysing encrypted network traffic with neural networks." In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2017, pp. 73–88.

[104] C Radhakrishnan, K Karthick, and R Asokan. "Ensemble Learning based Network Anomaly Detection using Clustered Generalization of the Features." In: *2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*. IEEE. 2020, pp. 157–162. DOI: https://doi.org/10.1109/ICACCCN51052.2020.9362791.

[105] Smitha Rajagopal, Katiganere Siddaramappa Hareesha, and Poornima Panduranga Kundapur. "Feature Relevance Analysis and Feature Reduction of UNSW NB-15 Using Neural Networks on MAMLS." In: *Advanced Computing and Intelligent Engineering*. Springer, 2020, pp. 321–

332. DOI: https://doi.org/10.1007/978-981-15-1081-6_27.

[106] Smitha Rajagopal, Poornima Panduranga Kundapur, and Katiganere Siddaramappa Hareesha. "A stacking ensemble for network intrusion detection using heterogeneous datasets." In: *Security and Communication Networks* 2020 (2020). DOI: https://doi.org/10.1155/2020/4586875.

[107] Guozheng Rao, Weihang Huang, Zhiyong Feng, and Qiong Cong. "LSTM with sentence representations for document-level sentiment classification." In: *Neurocomputing* 308 (2018), pp. 49–57. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2018.04.045. URL: https://www.sciencedirect.com/science/article/pii/S092523121830479X.

[108] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling with Large Corpora." English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, 2010, pp. 45–50.

[109] Matilda Rhode, Pete Burnap, and Kevin Jones. "Early-stage malware prediction using recurrent neural networks." In: *Computers & Security* 77 (2018), pp. 578–594. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2018.05.010. URL: https://www.sciencedirect.com/science/article/pii/S0167404818305546.

[110] Markus Ring, Alexander Dallmann, Dieter Landes, and Andreas Hotho. "IP2Vec: Learning Similarities Between IP Addresses." In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. 2017, pp. 657–666. DOI: 10.1109/ICDMW.2017.93.

[111] Martin Roesch et al. "Snort: Lightweight intrusion detection for networks." In: *Lisa*. Vol. 99. 1. 1999, pp. 229–238.

[112] S. Suthaharan and T. Panchagnula. "Relevance feature selection with data cleaning for intrusion detection system." In: *2012 Proceedings of IEEE Southeastcon*. 2012, pp. 1–6. DOI: 10.1109/SECon.2012.6196965.

[113] Roberto Saia, Salvatore Carta, Diego Reforgiato Recupero, and Gianni Fenu. "A Feature Space Transformation to Intrusion Detection Systems." In: *KDIR*. 2020, pp. 137–144.

[114] Roberto Saia, Salvatore Carta, Diego Reforgiato Recupero, Gianni Fenu, and Madalina Stanciu. "A Discretized Extended Feature Space (DEFS) Model to Improve the Anomaly Detection Performance in Network Intrusion Detection Systems." In: *KDIR*. 2019, pp. 322–329.

[115] Fadi Salo, Ali Bou Nassif, and Aleksander Essex. "Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection." In: *Computer Networks* 148 (2019), pp. 164–175. DOI: https://doi.org/10.1016/j.comnet.2018.11.010.

[116] Sangkatsanee, Phurivit and Wattanapongsakorn, Naruemon and Charnsripinyo, Chalermpol. "Practical Realtime Intrusion Detection Using Machine Learning Approaches." In: *Comput. Commun.* 34.18 (2011), pp. 2227–2235. ISSN: 0140-3664. DOI: 10.1016/j.comcom.2011.07.001. URL: http://dx.doi.org/10.1016/j.comcom.2011.07.001.

[117] S Anantha Sayana. "Approach to auditing network security." In: *Information Systems Control Journal* 5 (2003), pp. 21–23.

[118] *Selenium Python*. https://selenium-python.readthedocs.io. Online; retrieved on May 14, 2021. 2021.

[119] Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang. "Accurate, Scalable in-Network Identification of P2p Traffic Using Application Signatures." In: *Proceedings of the 13th International Conference on World Wide Web*. WWW '04. New York, NY, USA: Association for Computing Machinery, 2004, 512–521. ISBN: 158113844X. DOI: 10.1145/988672.988742. URL: https://doi.org/10.1145/988672.988742.

[120] Subhabrata Sen and Jia Wang. "Analyzing peer-to-peer traffic across large networks." In: *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. 2002, pp. 137–150. DOI: https://doi.org/10.1109/TNET.2004.826277.

[121] *Service Name and Transport Protocol Port Number Registry*. https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml. Online; retrieved on October 10, 2021. 2021.

[122] Iman Sharafaldin, Amirhossein Gharib, Arash Habibi Lashkari, and Ali A Ghorbani. "Towards a reliable intrusion detection benchmark dataset." In: *Software Networking* 2018.1 (2018), pp. 177–200. DOI: https://doi.org/10.13052/jsn2445-9739.2017.009.

[123] Anish Singh Shekhawat, Fabio Di Troia, and Mark Stamp. "Feature analysis of encrypted malicious traffic." In: *Expert Systems with Applications* 125 (2019), pp. 130–141. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2019.01.064. URL: https://www.sciencedirect.com/science/article/pii/S095741741930082X.

[124] Ali Shiravi, Hadi Shiravi, Mahbod Tavallaee, and Ali A Ghorbani. "Toward developing a systematic approach to generate benchmark datasets for intrusion detection." In: *computers & security* 31.3 (2012), pp. 357–374. DOI: https://doi.org/10.1016/j.cose.2011.12.012.

[125] Douglas C Sicker, Paul Ohm, and Dirk Grunwald. "Legal issues surrounding monitoring during network research." In: *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. 2007, pp. 141–148.

[126] Kamran Siddique, Zahid Akhtar, Farrukh Aslam Khan, and Yangwoo Kim. "KDD Cup 99 data sets: a perspective on the role of data sets in network intrusion detection research." In: *Computer* 52.2 (2019), pp. 41–51. DOI: https://doi.org/10.1109/MC.2018.2888764.

[127] Amrit Pal Singh and Arvinder Kaur. "Flower pollination algorithm for feature analysis of kyoto 2006+ data set." In: *Journal of Information and Optimization Sciences* 40.2 (2019), pp. 467–478. DOI: https://doi.org/10.1080/02522667.2019.1580886.

[128] *Snort IDS*. https://snort.org/. Online; retrieved on May 10, 2021. 2021.

[129] Jungsuk Song, Hiroki Takakura, Yasuo Okabe, Masashi Eto, Daisuke Inoue, and Koji Nakao. "Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation." In: *Proceedings of the first workshop on building analysis datasets and gathering experience returns for security*. 2011, pp. 29–36. DOI: https://doi.org/10.1145/1978672.1978676.

[130] Deris Stiawan, Mohd Yazid Bin Idris, Alwi M Bamhdi, Rahmat Budiarto, et al. "CICIDS-2017 dataset feature analysis with information gain for anomaly detection." In: *IEEE Access* 8 (2020), pp. 132911–132921. DOI: https://doi.org/10.1109/ACCESS.2020.3009843.

[131] Stratosphere. *Stratosphere Laboratory Datasets*. Retrieved March 13, 2020, from https://www.stratosphereips.org/datasets-overview. 2015.

[132] Liya Su, Yepeng Yao, Ning Li, Junrong Liu, Zhigang Lu, and Baoxu Liu. "Hierarchical Clustering Based Network Traffic Data Reduction for Improving Suspicious Flow Detection." In: *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. 2018, pp. 744–753. DOI: 10.1109/TrustCom/BigDataSE.2018.00108.

[133] Tongtong Su, Huazhi Sun, Jinqi Zhu, Sheng Wang, and Yabo Li. "BAT: deep learning methods on network intrusion detection using NSL-KDD dataset." In: *IEEE Access* 8 (2020), pp. 29575–29585. DOI: https://doi.org/10.1109/ACCESS.2020.2972627.

[134] Nasrin Sultana, Naveen Chilamkurti, Wei Peng, and Rabei Alhadad. "Survey on SDN based network intrusion detection system using machine learning approaches." In: *Peer-to-Peer Networking and Applications* 12.2 (2019), pp. 493–501. ISSN: 1936-6450. DOI: 10.1007/s12083-017-0630-0. URL: https://doi.org/10.1007/s12083-017-0630-0.

[135] Bayu Adhi Tama, Lewis Nkenyereye, SM Riazul Islam, and Kyung-Sup Kwak. "An enhanced anomaly detection in web traffic using a stack of classifier ensemble." In: *IEEE Access* 8 (2020), pp. 24120–24134. DOI: https://doi.org/10.1109/ACCESS.2020.2969428.

[136] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. "A detailed analysis of the KDD CUP 99 data set." In: *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE. 2009, pp. 1–6. DOI: https://doi.org/10.1109/CISDA.2009.5356528.

[137] Mahbod Tavallaee, Natalia Stakhanova, and Ali Akbar Ghorbani. "Toward credible evaluation of anomaly-based intrusion-detection methods." In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40.5 (2010), pp. 516–524. DOI: https://doi.org/10.1109/TSMCC.2010.2048428.

[138] *The Relevance of Network Security in an Encrypted World*. https://blogs.vmware.com/networkvirtualization/2020/09/network-security-encrypted.html. Online; retrieved on December 2, 2021. 2021.

[139] Luan Tran, Liyue Fan, and Cyrus Shahabi. "Outlier Detection in Non-Stationary Data Streams." In: *Proceedings of the 31st International Conference on Scientific and Statistical Database Management*. SSDBM '19. Santa Cruz, CA, USA: Association for Computing Machinery, 2019, 25–36. ISBN: 9781450362160. DOI: 10.1145/3335783.3335788. URL: https://doi.org/10.1145/3335783.3335788.

[140] *TrickBot: The multi-faceted botnet*. https://www.kaspersky.com/resource-center/threats/trickbot. Online; retrieved on January 12, 2022. 2022.

[141] *UMass Trace Repository*. http://traces.cs.umass.edu/index.php/Network/Network. Online; retrieved on May 10, 2021. 2021.

[142] Antoine Varet and Nicolas Larrieu. "Realistic Network Traffic Profile Generation: Theory and Practice." In: *Comput. Inf. Sci.* 7.2 (2014), pp. 1–16. DOI: https://doi.org/10.5539/cis.v7n2p1.

[143] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar. "A survey of methods for encrypted traffic classification and analysis." In: *International Journal of Network Management* 25.5 (2015), pp. 355–374. DOI: https://doi.org/10.1002/nem.1901.

[144] Fatema Bannat Wala and Chase Cotton. "Unconstrained Endpoint Security System: UEPTSS." In: *International Journal of Network Security & Its Applications (IJNSA) Vol 10* (2018).

[145] Jing Wang and Ioannis Ch Paschalidis. "Botnet detection based on anomaly and community detection." In: *IEEE Transactions on Control of Network Systems* 4.2 (2016), pp. 392–404. DOI: https://doi.org/10.1109/TCNS.2016.2532804.

[146] *Wikipedia*. https://en.wikipedia.org/wiki/Sleman_Regency. Online; retrieved on July 7, 2022. 2021.

[147] Xi Xiao, Shaofeng Zhang, Francesco Mercaldo, Guangwu Hu, and Arun Kumar Sangaiah. "Android malware detection based on system call sequences and LSTM." In: *Multimedia Tools and Applications* 78.4 (2019), pp. 3979–3999. ISSN: 1573-7721. DOI: 10.1007/s11042-017-5104-0. URL: https://doi.org/10.1007/s11042-017-5104-0.

[148] Ibrahim Yilmaz, Rahat Masum, and Ambareen Siraj. "Addressing imbalanced data problem with generative adversarial network for intrusion detection." In: *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*. IEEE. 2020, pp. 25–30. DOI: https://doi.org/10.1109/IRI49571.2020.00012.

[149] Arif Yulianto, Parman Sukarno, and Novian Anggis Suwastika. "Improving AdaBoost-based Intrusion Detection System (IDS) Performance on CIC IDS 2017 Dataset." In: *Journal of Physics: Conference Series* 1192 (2019), p. 012018. DOI: 10.1088/1742-6596/1192/1/012018. URL: https://doi.org/10.1088/1742-6596/1192/1/012018.

[150] Safaa Zaman, Mohammed El-Abed, and Fakhri Karray. "Features Selection Approaches for Intrusion Detection Systems Based on Evolution Algorithms." In: *Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication*. ICUIMC '13. Kota Kinabalu, Malaysia: Association for Computing Machinery, 2013. ISBN: 9781450319584. DOI: 10.1145/2448556.2448566. URL: https://doi.org/10.1145/2448556.2448566.

[151] Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. "A survey of intrusion detection in Internet of Things." In: *Journal of Network and Computer Applications* 84 (2017), pp. 25–37. ISSN: 1084-8045. DOI: https://doi.org/10.1016/j.jnca.2017.02.009. URL: https://www.sciencedirect.com/science/article/pii/S1084804517300802.

[152] *Zeek IDS*. https://zeek.org. Online; retrieved on May 10, 2021. 2021.

[153] *Zeus Github*. https://github.com/Visgean/Zeus/blob/c55a9fa8c8564ec196604a59111708fa8415f020/manual_en.html. Online; retrieved on December 1, 2021. 2021.

[154] *Zeus Trojan Analysis*. https://talosintelligence.com/zeus_trojan. Online; retrieved on January 12, 2022. 2022.

[155] Chongzhen Zhang, Fangming Ruan, Lan Yin, Xi Chen, Lidong Zhai, and Feng Liu. "A deep learning approach for network intrusion detection based on NSL-KDD dataset." In: *2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*. IEEE. 2019, pp. 41–45. DOI: https://doi.org/10.1109/ICASID.2019.8925239.

[156] Jun Zhang, Yang Xiang, Wanlei Zhou, and Yu Wang. "Unsupervised traffic classification using flow statistical properties and IP packet payload." In: *Journal of Computer and System Sciences* 79.5 (2013), pp. 573–585. ISSN: 0022-0000. DOI: https://doi.org/10.1016/j.jcss.2012.11.004. URL: https://www.sciencedirect.com/science/article/pii/S0022000012001729.

[157] Rongfeng Zheng, Jiayong Liu, Liang Liu, Shan Liao, Kai Li, Jihong Wei, Li Li, and Zhiyi Tian. "Two-layer detection framework with a high accuracy and efficiency for a malware family over the TLS protocol." In: *PLOS ONE* 15.5 (May 2020), pp. 1–22. DOI: 10.1371/journal.pone.0232696. URL: https://doi.org/10.1371/journal.pone.0232696.

[158] Xiaoyan Zhuo, Jialing Zhang, and Seung Woo Son. "Network intrusion detection using word embeddings." In: *2017 IEEE International Conference on Big Data (Big Data)*. 2017, pp. 4686–4695. DOI: 10.1109/BigData.2017.8258516.

[159] Richard Zuech, Taghi Khoshgoftaar, Naeem Seliya, Maryam Najafabadi, and Clifford Kemp. "A New Intrusion Detection Benchmarking System." In: (2015). URL: https://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS15/paper/view/10368.