IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 August 2024, accepted 8 October 2024, date of publication 17 October 2024, date of current version 30 October 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3482728

RESEARCH ARTICLE

Data Fingerprinting and Visualization for AI-Enhanced Cyber-Defence Systems

CHRISTIAAN KLOPPER AND JAN H. P. ELOFF
Department of Computer Science, University of Pretoria, Pretoria 0002, South Africa
Corresponding author: Jan H. P. Eloff (jan.eloff@up.ac.za)

ABSTRACT Artificial intelligence (AI)-assisted cyber-attacks have evolved to become increasingly successful in every aspect of the cyber-defence life cycle. For example, in the reconnaissance phase, AI-enhanced tools such as MalGAN can be deployed. The attacks launched by these types of tools automatically exploit vulnerabilities in cyber-defence systems. However, existing countermeasures cannot detect the attacks launched by most AI-enhanced tools. The solution presented in this paper is the first step towards using data fingerprinting and visualization to protect against AI-enhanced attacks. The AIECDS methodology for the development of AI-Enhanced Cyber-defense Systems was presented and discussed. This methodology includes tasks for data fingerprinting and visualization. The use of fingerprinted data and data visualization in cyber-defense systems has the potential to significantly reduce the complexity of the decision boundary and simplify the machine-learning models required to improve detection efficiency, even for malicious threats with minuscule sample datasets. This was validated by showing how the resulting fingerprints enable the visual discrimination of benign and malicious events as part of a use case for the discovery of cyber threats using fingerprint network sessions.

INDEX TERMS Cyber-defense, cyber security, data fingerprint, data visualization, intelligent system.

I. INTRODUCTION

One of the biggest challenges of the 21st century is defending cyber assets from cyber-attacks [1], [2], [3], and [4]. Worldwide events such as COVID-19 and the Russian invasion of Ukraine gave threat actors the opportunity to significantly increase cyber-attacks [1], [3], and [4]. The main defensive challenges are advanced and complex attacks, unprotected data, poor cybersecurity practices, and defenses based on vulnerability management, thereby exposing the cyber-defense perimeter [2], [3]. Simultaneously, threat actors are continually finding innovative means to deploy Artificial Intelligence (AI)-enhanced cyber-attacks [3] and increase their capabilities to target critical vulnerabilities more quickly [1]. This has led to more frequent cyber-attacks with greater complexity and smaller time windows to perform critical vulnerability patching. Consequently, by 2021, zero-day attacks have nearly doubled [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

A study performed by the University of New South Wales (UNSW) and Australian Cybersecurity Center (ACSC) unintentionally demonstrated the ineffectiveness of state-of-the-art cyber-defense systems in 2015 [5]. The majority of these countermeasures are based on machine learning models, particularly anomaly detection techniques. Threat actors target instabilities in machine learning (ML) and AI by poisoning input data, using adversarial attacks to confuse models during inference [1], [6], and using generative adversarial networks (GAN) to enhance cyber-attacks [7]. Cybercriminals have advanced their ability to execute highly sophisticated AI-enhanced attacks, which are becoming increasingly difficult to discover, detect, and protect [7], [8].

Among the many reasons for the poor performance of UNSW projects, the data problem is particularly noteworthy. This could be attributed to the fact that the data used were created in a laboratory, not real-world data or real-time data. These data properties are of prime importance when building cyber-defence systems. AI-enhanced ML cyber-defense solutions should be trained using real-world attack data [9]. However, there are multiple shortcomings

in obtaining real-world data for ML-based cybersecurity solutions: (i) the lack of availability of large real-world attack datasets; (ii) the sensitive nature of the data; and (iii) security, confidentiality, and privacy concerns [10], [11]. Research conducted by [12, 6, 18] has determined that most ML cybersecurity studies have not been tested or trained in real-time environments. This is critical for determining the detection efficiency in practical scenarios in which cyber-defence systems are intended to be deployed. In [13], the authors (s) concluded that high-quality real-world and real-time data are required to counter cyber threats.

Training machine learning models on visualized data has proven to be more successful than training on raw data [14]. This is because researchers have identified that visualizations can represent complex, large, and multimodal datasets as simple datasets [14], which simplifies the learning task for AI models. This opens up an opportunity for developers of cyber-defense systems to develop AI-enhanced tools that can be trained using visualized data. Furthermore, visualized representations of data create an opportunity to extract more meaningful real-world data from threat-related environments such as computer networks.

This study represents the first step in addressing certain aspects of the data problem by proposing a methodology for the development of AI-enhanced cyber-defence tools that include tasks for data fingerprinting and data visualization. The remainder of this paper is organized as follows. First, related work on threats to the cyber-defense lifecycle and the efficiency of state-of-the-art cyber-defense machine-learning models are discussed. This was followed by the proposal of a methodology for the development of AI-enhanced cyber-defence solutions. This methodology includes data fingerprinting tasks that are discussed in more detail. The application of this methodology is demonstrated using a use case study that focuses on the discovery of cyber threats through fingerprint network sessions.

II. RELATED WORK

Current cyber-defense research outputs that are important for the research at hand include the following.

A The cyber-defense lifecycle.

B State-of-the-art ML-based cyber-defense tools.

A. CYBER-DEFENSE LIFE CYCLE

The cyber-defense lifecycle stipulates that the phases of a cyber-attacker must be completed to infiltrate the organization. The cyber-defense lifecycle phases [7] are as follows: *Reconnaissance*, which collects information and intelligence for the planned cyber-attack; *Weaponization*, which focuses on the effectiveness of the cyberattack; *Delivery*, which bypasses existing safeguards; *Exploitation*, which infiltrates; *Installation*, which opens the network for malicious attacks; *Command & control*- remote control of the network; *Actions* that execute the intended malicious activity. All phases of the cyber-defense life cycle should be considered when developing methodologies for the development of AI-enhanced cyber-defense systems. For the research conducted for this paper the reconnaissance phase is of particular interest for the encoding and visualization of data for the development of AI-enhanced cyber-defense countermeasures.

Researchers have identified cyber threats that demonstrate the use of AI-enhanced attack tools within different phases of the cyber-defense lifecycle [15, 7]. Consider, for example, the reconnaissance phase in which an AI-enhanced tool such as MalGAN can be deployed. MalGAN generates concealed adversarial malware that can successfully bypass ''black-box'' malware detectors [16]. Another tool, DeepLocker [17], conceals its malware payload to activate it only when triggered. This is achieved by training adversarial samples that mutate the payload to obfuscate the normal appearance. DeepLocker represents advances in AI-enhanced tools that can be deployed in the Command & Control phases of the cyber-defence lifecycle.

B. STATE-OF-THE-ART CYBER-DEFENSE TOOLS

The ML-based cybersecurity countermeasures detailed in the literature show that there is still a significant gap in achieving a cyber-defense system to overcome the current cybersecurity threats. Researchers have concluded that most datasets used in ML-based detection systems research are outdated and do not typically reflect real-world traffic or the latest cyber-attacks accurately [18, 19, 6, and 12]. The findings indicate that legacy datasets used in Intrusion Detection Software (IDS) ML research represent 88% of the dataset distribution [6]. In addition, this study indicated that in ML research on Malware Detection Software (MDS), customized datasets represented 33% of the dataset distribution, and 20% of the datasets were created before 2012. Findings in [18] determined that legacy datasets had the highest majority, representing 56% of the dataset distribution in ML research. According to [18], although experimental results on legacy datasets are excellent, they decrease significantly when tested on more recent datasets, including real-world datasets.

The lack of real-world datasets is compounded by the inability to extract meaningful information from real-world systems such as computer network environments. Other studies [13, 20, 6, and 12] have indicated that most experiments for prototyping network-based cyber-defense systems use simplified calculated features based on data telemetry and averaging statistics. According to [6], simplified calculated features result in increased inference sensitivity and time delay for classifying cyber-attacks.

ML-based IDS countermeasures have evolved from techniques that are heavily dependent on feature engineering to Deep Learning, which is less dependent on feature engineering. This results in more complex models with incremental performance improvement. However, the detection of threats with minuscule malicious samples has not yet been improved. Although several attempts have used dataset rebalancing, no advancements have been made in the techniques that

perform better in detecting threats with minute malicious samples. This, combined with real-time timing differences, is most likely why IDS systems perform worse in real-time environments than in laboratories. In addition, most IDS research has been conducted on outdated datasets without current threats. Similar to ML-based IDS countermeasures, [21] conducted a comprehensive review of ML-based MDS approaches, considering signature, behavior, heuristic, model checking, DL, cloud, mobile, and Internet-of-Things-based detection. According to this study, although advancements have been made in every approach, no ML-based MDS detection approach has successfully detected all malware types.

Malialis [22] was one of the only researchers to develop a Reinforced Learning model for real-time network intrusion detection and response. The input source was real-time network packets [22]. The authors (s) proposed the use of a distributed RL defence system to throttle DDoS attacks. This was modelled using a mesh network with a distributed group of routers configured by the author(s) as RL agents. The agents were then trained to limit the amount of DDoS traffic that could be passed through the network based on the flow of packets through each router.

In a survey by [23] on adversarial ML in cyber warfare, the author(s) concluded that there are serious concerns regarding vulnerabilities in ML-based cyber-defense systems. According to the author(s), faulty assumptions during ML model training are the main cause of vulnerabilities. The author(s) further noted that AI, which confuses models during inference, is a direct result of assuming that data in datasets are linearly separable and solvable using linear functions. This was indirectly verified in practice by [3], who reported in 2022 that there will be an increase in sophisticated cyber threats, enabling threat actors to repeat cyberattacks on a greater scale and speed.

Based on the above discussion, the following requirements for methodologies that provide guidelines for the development of AI-enhanced cyber-defence systems are considered important.

- Improve detection rates and reduce detection time.
- Employ dynamic self-learning and RL approaches.
- Detect adversarial and unknown cyber-attacks.
- Detect threats and attacks with minute sample data sets.
- Training AI-enhanced countermeasures in real-world environments using real-time data.
- The focus is on extracting and encoding meaningful data from real-world systems, also referred to as fingerprinting.
- Visualize the data to overcome complexity in multi-modal threat related data.

III. THE AIECDS- METHODOLOGY FOR THE DEVELOPMENT OF AI-ENHANCED CYBER-DEFENSE SYSTEMS

Figure 1 depicts a so-called AIECDS (AI-Enhanced Cyber-defense System)-methodology developed by the same authors of the research at hand and adopted from previous research [24]. The AIECDS methodology provides guidelines for the development of AI-enhanced cyber-defence systems. However, this study presents a high-level overview of AIECDS methodology and discusses the fingerprints and visualization of the data in more detail. Furthermore, the application of AIECDS methodology is illustrated through a use case study for the discovery of cyber threats in fingerprinted network sessions.

A. PHASES OF THE AIECDS-METHODOLOGY

As shown in Figure 1 the AIECDS methodology consists of the following phases:

1) DATASET (REAL-TIME ENVIRONMENT)
2) EXTRACT FEATURES AND BUFFER DATA
3) DATA PREPARATION
4) FINGERPRINT SESSIONS
5) THREAT DETECTION

A high-level overview of each phase is provided below, with detailed attention to the fingerprinting phase.

1) DATASET (REAL-TIME ENVIRONMENT)

According to the criteria for AIECDS, the guideline is to train the AI-enhanced countermeasures in real-world environments using real-time data.

Use case: Network packets are captured via Packet Capture (PCAP) technology, and data are extracted in near real time, limited to information that is available to a firewall. A continuous stream of packets is processed by a separate system with minute processing delays, and is trained to detect threats prior to completing data transfer. This enables the proposed solution to detect threats within a live network while maintaining a low computational complexity, thereby reducing delays in threat detection. The last mentioned is one of the criteria for the AIECDS methodology. This is achieved using a PCAP dataset, which contains data extracted from a real-time network environment and starts to detect threats as packets are received. An example of a PCAP dataset is DARPA's UNSW-15 dataset [5], which has been used in many cyber security machine learning research projects. These datasets contain attacks, including DoS, worms, backdoors, fuzzers, and zero-day attacks, among others. For example, the UNSW-15 [5] dataset contains 100 GB of network packets (in PCAP format) with 82 million network packets in the training dataset alone. In addition, this dataset contains threat labels for threat categories rather than actual attacks, which is more meaningful for training high-performance threat detection algorithms.

2) EXTRACT FEATURES AND BUFFER DATA

According to AIECDS criteria, packets must be processed as they are received. This is achieved by extracting the key features for each packet received, and storing the results for each packet in a buffer. The buffer is periodically or fully input into the data preparation phase, after which it is cleared.

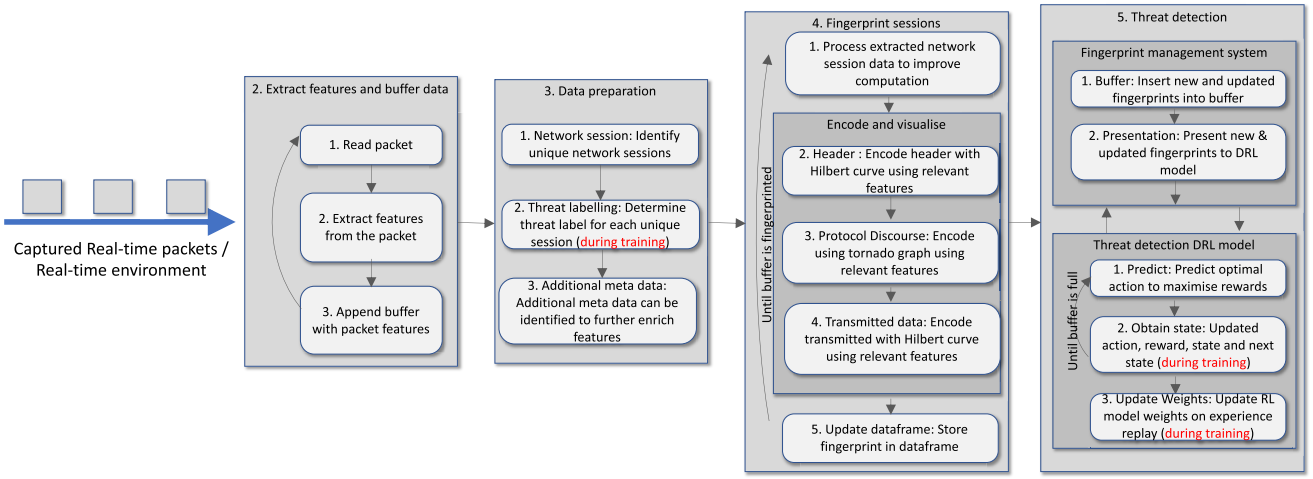

FIGURE 1. AIECDS-methodology adapted for the use case that fingerprints network sessions.

Use case: Network session datasets, such as the UNSW-15 PCAP dataset, contain a wide assortment of packets, including IP and address resolution protocols (ARP), network protocols, and a wide array of transport protocols. In use cases in which the focus is on network sessions, the following are examples of meaningful features: IP source, IP destination, IP length, TCP flags, source port, destination port, protocol, ARP p-source, ARP p-destination, and transmitted data. The extraction process is completed by extracting data in batches from the PCAP files during training, or by buffering real-time packets as they are received.

3) DATA PREPARATION

Data preparation involves the preparation of the extracted features for fingerprinting. It uses data frames from the extracted features and the buffer data phase. The fingerprint data frame represents extracted real-world data or events. Threat labelling was performed to link the available metadata from the features to the extracted dataset (during training). Additional information may enrich the understanding of these features.

Use case: The data preparation phase uses the data frame output extracted from the PCAP dataset, which was subsequently prepared for fingerprinting. The fingerprint data frame represents the extracted network sessions. Threat labelling was performed to link the available metadata from the features to the extracted PCAP dataset. Additional information includes applications and services that enrich the selected features. These features are then added to the corresponding fingerprints.

4) FINGERPRINT SESSIONS

The following criteria (see Section II) were specifically addressed in the "fingerprint session phase":

- Extract and encode (fingerprinting) meaningful data from real world systems.
- Visualize the data to overcome complexity in multi-modal threat related data.

Examples of space-filling [25] encoding and visualization techniques [26] include natural ordering, line-by-line, column-by-column, Hilbert, and Morton. For the AIECDS methodology, Hilbert curves [27] and tornado graphs [28] were chosen to encode and visualize the real-world data. Briefly, a Hilbert curve is a continuous fractal space-filling curve [27], whereas a tornado graph is a special type of barchart [28]. The decision to use Hilbert curves is based on the fact that they maintain the relative positions of data elements within the overall data structure. For example, the positions of network packets within network sessions are significant. A detailed explanation of the use of the Hilbert curves can be found in [24].

The detailed design of the fingerprint representations depends on the specifics of the use case. As mentioned previously, the use case employed to demonstrate the construction of a fingerprint for the purpose of this research was the discovery of cyber threats using fingerprinted network sessions. The fingerprint design for the use case has three distinct sections with different encoding approaches: the header, protocol discourse, and transmitted data.

Header: The header of a fingerprint must be unique to each event or session. The reason that the header section is encoded in a specific manner is to enlarge its prominence within the final fingerprint, because the significance of behaviors for certain unique events or sessions may otherwise be missed.

Use case: The source and destination IP addresses, ports, and protocols are sufficient for representing a unique network session. This is illustrated in Figure 2.

Both the TCP and UDP port numbers range from zero to 65535, which can be encoded using four colors (from light gray to black) and two eight $\times$ eight Hilbert curves. This was achieved by counting 255 in the first Hilbert curve for each one in the second Hilbert curve. Protocols range from zero to 255 and are encoded using a similar approach to IP sections. The last eight $\times$ eight Hilbert curve is reserved for future use and is required to complete the 128 columns required for the 128 $\times$ 128 Hilbert curve used for the transmitted data section.

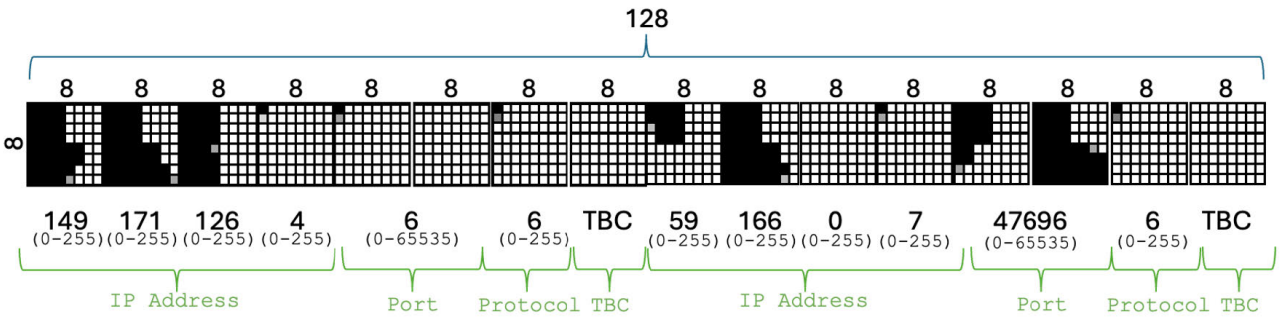

FIGURE 2. Fingerprint header design.

One possible future application for the reserved eight × eight space could be to record the frame sizes for IP, TCP, and UDP or metrics for other transport or application protocols.

Protocol discourse:

The protocol discourse section of a fingerprint must represent the communication sequence between multiple hosts. This is achieved using certain attributes and features originating from a sequence of communications. The use case is illustrated in Figure 3.

Use case: In Figure 3, the protocol exchange is visualized for a TCP session between 59.166.0.7 on port 53421 and 149.171.126.4 on port 80. The exchange is initiated with a request to synchronize (1), which is acknowledged (2), after which the initial setup is acknowledged, pushed, and acknowledged (3, 4, and 5). Large packets (6, 7, 9, 11, and 13) are then sent and acknowledged (8, 10, and 12). The session is finalized at the end with a finish and acknowledges (14, 15) before closing the exchange (16). The fingerprint has a sufficient capacity to capture 128 interactions between two hosts, which can contain multiple flows within the same unique session.

Transmitted data:

The fingerprint data section must encode relevant data within a unique session or event until the Hilbert curve is completed.

Use case: The data section of the fingerprint must encode the packet data for all packets within a unique session or until the 128 × 128 Hilbert curve is completed. The complete Hilbert curve is shown in Figure 4.

Data are transmitted in bytes, which are composed of eight bits. As a result, each byte can be converted into a decimal range from zero to 255, which can be encoded into grayscale colors. Therefore, each element of the 128 × 128 Hilbert curve can depict a byte using 256 grayscale colors. A 128 × 128 Hilbert curve was selected to develop dense transmitted data visualization to limit future changes in the fingerprint shape.

5) THREAT DETECTION

The criteria for AIECDS methodology include the use of dynamic self-learning and RL. Therefore, the threat detection phase of the AIECDS methodology was designed as shown in Figure 1. The threat detection phase consists of two main tasks, one for managing the fingerprint system and another for training the threat detection DRL (Detection Reinforced Learning) model. The purpose of the fingerprint management system is to buffer and maintain all fingerprints. This is achieved by recording a state for each fingerprint, which should include the available fingerprint space, when it was last presented to the threat detection DRL model, and when the fingerprint was last updated. The fingerprint-management system shown in Figure 1 is illustrated in Figure 5. The fingerprint management system involves inserting newly created fingerprints into the buffer and scheduling them to be presented to the threat detection DRL model as well as routinely scheduling existing fingerprints to be presented to the threat detection DRL model once updated.

Finally, once a fingerprint has been classified by threat detection DRL, the result is stored in the fingerprint, appropriate actions are taken to mitigate any risks, and the fingerprint is removed from the buffer.

The purpose of the threat-detection DRL model is to correctly detect cyber threats and threat types with as little information as possible. This refers to one of the criteria in the AIECDS methodology that states that it should be possible to detect threats and attacks within minute sample datasets. This can be achieved by presenting fingerprints to the threat detection DRL model in incremental steps as the fingerprints are updated over time. Higher rewards should be allocated to the threat-detection DRL model with early detection and the largest negative rewards should be allocated to incorrect threats or threat models with early detection. This is illustrated in Figure 5. The AIECDS criteria "Detect against adversarial and unknown cyber-attacks" is achieved by learning the patterns of adversarial attacks, however indistinct it may be.

Use case: Over time, a sufficient number of network session fingerprints visually profile the boundaries between benign and malicious network sessions. These fingerprints can then be used by DRL to learn the features that make each malicious attack type unique, and to detect unknown or new cyber-attacks. The RL algorithm operates and detects threats in real time when network packets are received. The proposed solution will have a visualized view of the transmitted data,

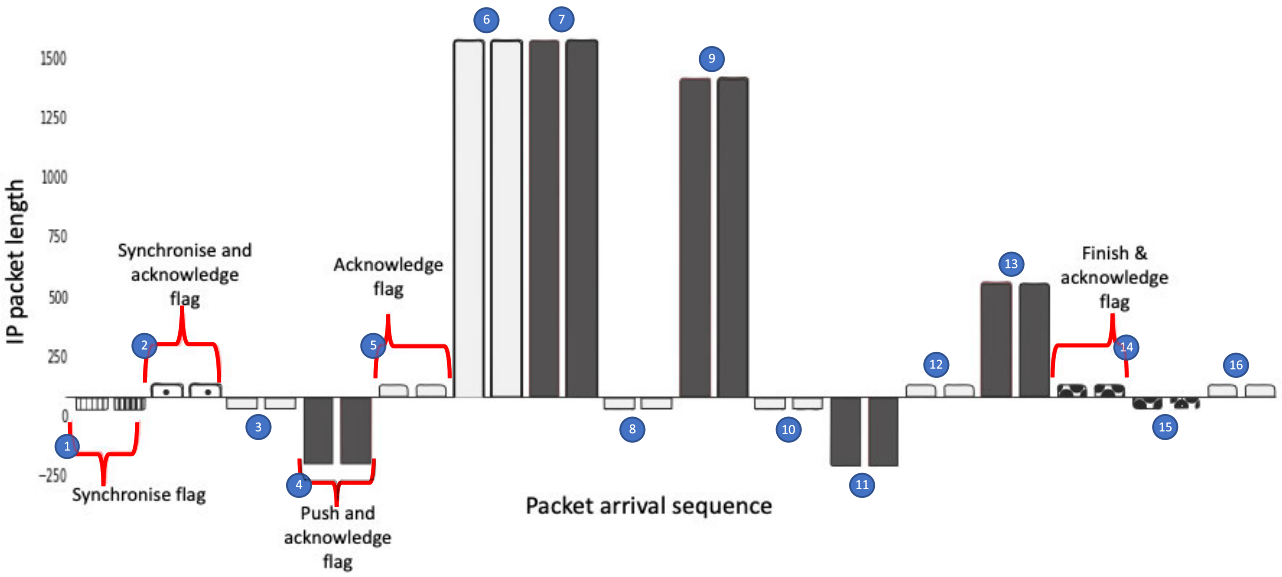

FIGURE 3. Protocol discourse design.

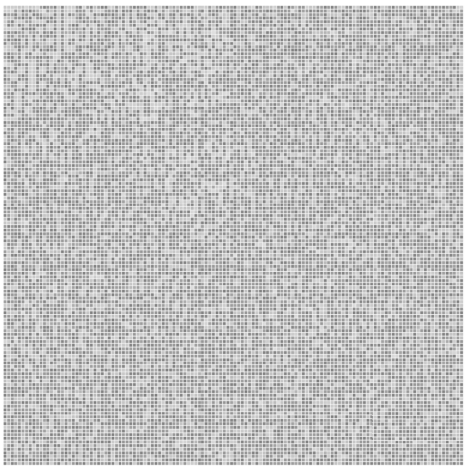

FIGURE 4. Transmitted data design.

irrespective of the nature of the packets sent, thereby being more resilient against temporal fluctuations. In addition, the fingerprint does not compare text, video, or music to existing formats but rather looks at what is different at the byte level. At the byte level, malicious intent is likely to be more visible because bytes are included in the network session for malicious code. Malicious code is revealed by fingerprinting the entire network session at the lowest possible information level, which is at the byte level.

IV. EXPERIMENTAL RESULTS

A prototype environment for the use case: ''discovery of cyber threats in fingerprinted network sessions'' was set-up and applied to the UNSW-15 dataset. A total of 10240 network sessions were fingerprinted, containing both benign and malicious fingerprints.

A. MALICIOUS FINGERPRINT ANALYSIS RESULTS

Malicious fingerprints were clustered to obtain the key fingerprints representing each malware threat category in the UNSW-15 dataset. Additionally, each of the closest benign fingerprints was selected by minimizing the element-wise distance between the body of the fingerprint (protocol discourse and transmitted data) and the malicious fingerprints. As shown in Figure 6, eight of the nine malicious cyber-attack categories simulated within the UNSW-15 dataset were identified. The differences between malicious fingerprints and their closest benign fingerprints are discussed separately for transmitted data and protocol discourse.

1) MALICIOUS FINGERPRINT CLUSTERS

To structure the selection of malicious fingerprint samples for analysis, malware threat categories with more than four fingerprinted network sessions are clustered using k-means clustering. The optimal elbow was identified for each fingerprint with the smallest Euclidean distance from each cluster center. All fingerprints were selected for malware threat categories with four or fewer network session fingerprints. The total number of malicious network sessions that were fingerprinted and the key cluster fingerprint totals are shown in Fig. 6.

2) FROBENIUS DISTANCE

The Frobenius distance measure [29] was determined for each pair of fingerprints analyzed because it measures the movement between elements, thereby capturing the difference between fingerprints with similar pattern motifs. The importance of this aspect of the Frobenius distance is illustrated in Figure 7, where both images resemble the same pattern.

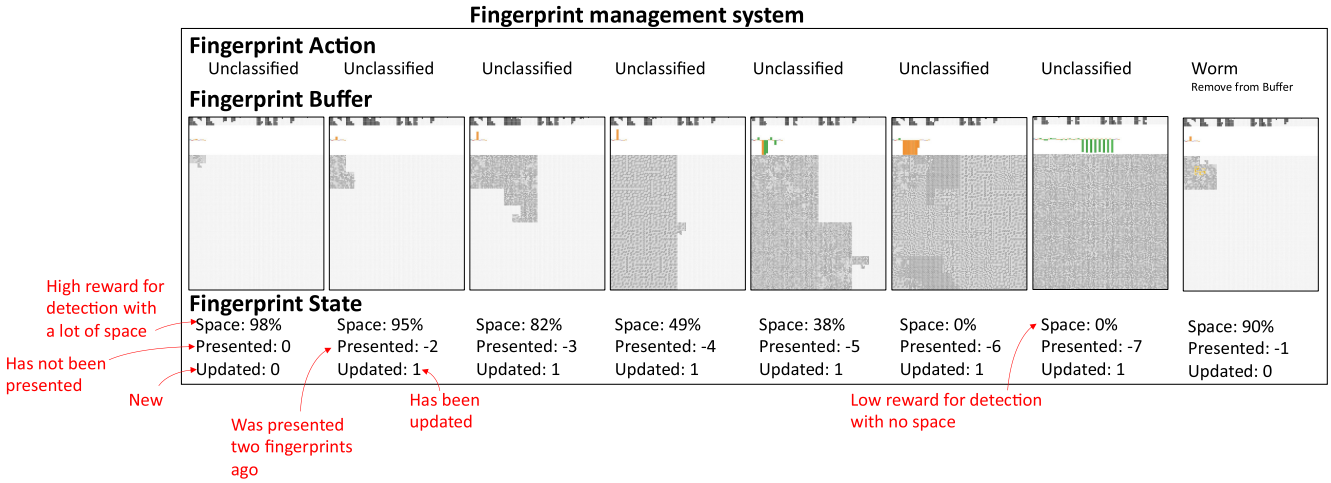

FIGURE 5. Illustration of the fingerprint management system.

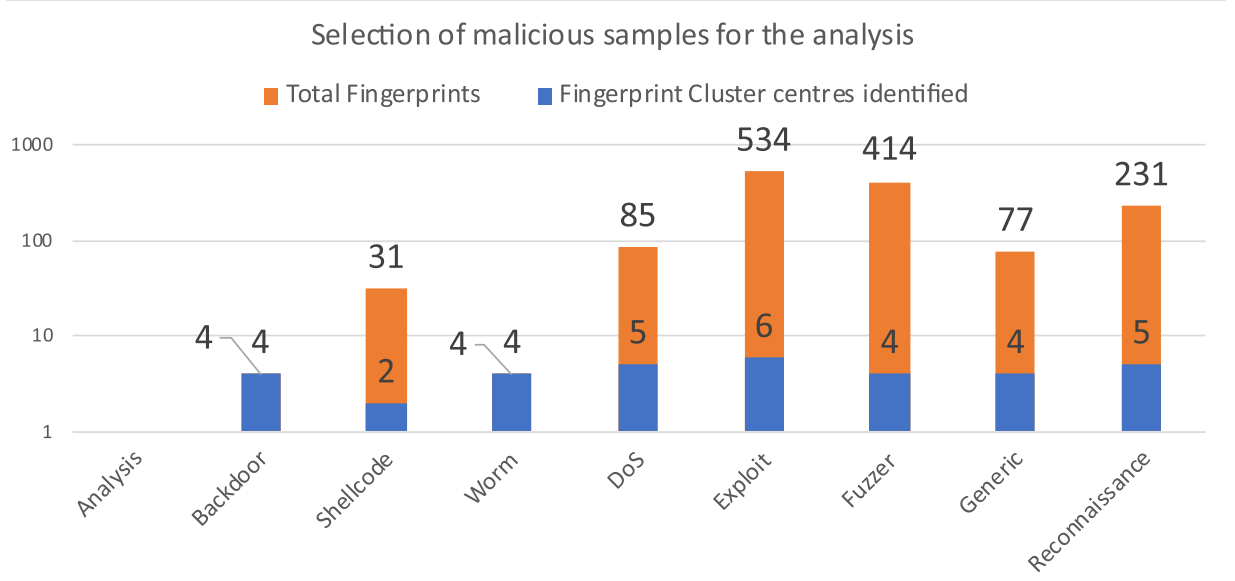

FIGURE 6. Fingerprint clustering per threat category.

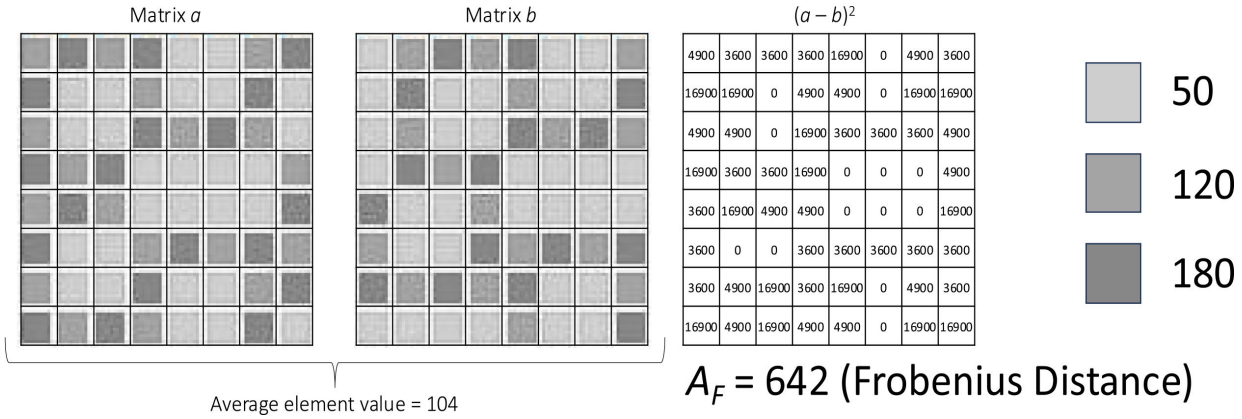

FIGURE 7. Frobenius distance illustration.

From the example illustrated in Figure 7, the square difference was calculated from the two 64-element matrices. The square root of the sum of differences was calculated as the Frobenius distance. This is 6.2 times the average element

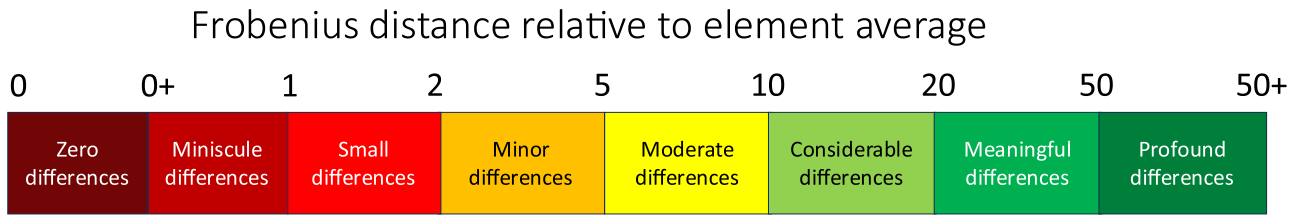

FIGURE 8. Frobenius distance significance.

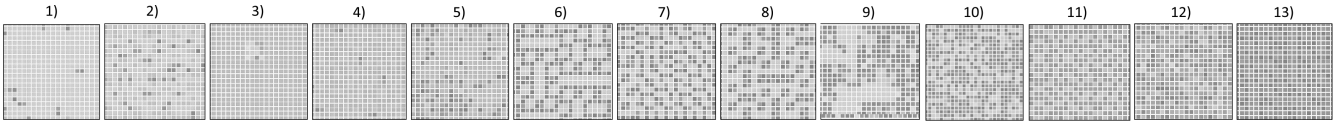

FIGURE 9. 13 unique patterns within transmitted data.

value in the example or, more simply, six additional elements in matrix 1 compared to matrix 2. This aspect is required for the comparison between fingerprints, because the DRL threat detection model considers fingerprints based on their element differences.

To determine the significance of the Frobenius distance for each fingerprint comparison, the gauge shown in Figure 8, which is based on the Frobenius distance relative to the average element value for the fingerprints, was used. The significance gauge is relevant to both the transmitted data and protocol discourse sections.

3) RESULTS OF TRANSMITTED DATA ANALYSIS

The transmitted data section of the fingerprint consists of a grid of 128 × 128 elements that can range from zero to 255, resulting in 4194304 factorial permutations that have infinite possibilities. Although infinite possibilities exist, it is clear from the analysis that there is a finite number of patterns. The focus here is to present the unique patterns discovered within the transmitted data, the results of the similarity analysis between the malicious and closest benign fingerprints, and the broader findings uncovered during the analysis. An evaluation guide (see Figure 9) was used to identify unique transmitted data patterns from the analysis results. Visual inspection of the fingerprints revealed 13 unique repeated patterns. The 13 different pattern types are shown in Figure 9.

The results of the transmitted data pattern analysis are listed in Table 1. The malicious and closest benign fingerprints are located in the same row to easily compare their pattern similarities. The columns include the row number (#), threat category, protocol, port, and pattern guide results for malicious fingerprints and the row number (#), protocol, port, and pattern guide results for the closest benign fingerprints. In addition, the Frobenius distance measure, mean, standard deviation, and significance were included.

The investigation of each threat type in Table 1 revealed interesting observations. For example, consider the shellcode and reconnaissance threat types. With reference to the shellcode threat type, both fingerprints 5.1 and 5.2 have the same transmitted data pattern. However, even though the transmitted data patterns were similar, the distance between the two was 1128 points. By contrast, fingerprints 6.1 and 6.2 do not have overlapping patterns with a smaller distance of 907 points owing to the small size of the transmitted data shape. The significance of these differences ranges from moderate to significant. The malicious reconnaissance and the closest benign fingerprints match completely in the transmitted data patterns. In addition, their fingerprints (30 – 34) had the smallest distances, with an average of 139 points and standard deviation of 28 points. The significance of these differences ranged from low to low.

Overall, the dominant pattern for both malicious and benign fingerprints was Pattern 9 (used 20 times for malicious fingerprints and 13 times for benign fingerprints). The second most dominant patterns for malicious fingerprints were Patterns 7 and 10 for benign fingerprints. Patterns 2 and 3 are frequently used by benign fingerprints but only once by a malicious fingerprint, whereas Pattern 4 is frequently used by malicious fingerprints but only once by a benign fingerprint. Finally, Pattern 11 is used by only one malicious fingerprint. The overall pattern analysis is shown in Figure 10.

It is clear from the transmitted data similarity analysis that the proposed solution provides a framework for identifying meaningful differences between malware and benign network sessions, and between malware threat categories. Not a single malware-transmitted data section was exactly the same as its closest benign-transmitted data section at a distance of zero.

All malware threat types, including malware categories that were undetectable in the UNSW-15 simulation (backdoor, shellcode, and worm), exhibited differences in patterns that could make these malware threats detectable using less complex algorithms. Even the reconnaissance malware with the smallest differences, which in the UNSW-15 simulation had the smallest detection ratio of 0.2%, had a consistent difference that could aid in the discovery of these threats. In addition, seven malicious fingerprints (7, 8, 9, 10, 11, 21, and 23) shared their closest benign fingerprints

TABLE 1. Transmitted data analysis results.

Malicious Fingerprints: Detail				Malicious Fingerprints: Patterns													Closest Benign Fingerprints: Detail			Closest Benign Fingerprints: Patterns																
#	Threat	Protocol	Port	1	2	3	4	5	6	7	8	9	10	11	12	13	#	Protocol	Port	1	2	3	4	5	6	7	8	9	10	11	12	13	Frobenius distance	Mean	Standard deviation	Significance
1.1	Backdoor	TCP	45947									■					1.2	UDP	43830			■									■		583	536	65	7.5
2.1		TCP	21554									■					2.2	TCP	80					■					■				440			5.6
3.1		TCP	47252									■					3.2	TCP	143			■										■	569			6.9
4.1		UDP	2140									■					4.2	TCP	80					■					■				551			6.8
5.1	Shellcode	TCP	18725										■				5.2	TCP	5582										■				1128	1018	156	12.8
6.1		UDP	17497												■		6.2	TCP	80									■				■	907			8.2
7.1	Worm	TCP	80									■					7.2	UDP	514		■								■				1294	2090	1600	15.8
8.1		TCP	80									■					8.2	TCP	80		■								■				1251			15.3
9.1		TCP	80									■					9.2	TCP	80		■								■				1324			16.1
10.1		TCP	80						■	■	■		■			■	10.2	TCP	80			■						■					4489			59.9
11.1	DoS	TCP	80						■			■		■			11.2	TCP	80			■						■					3135	3066	1373	42.9
12.1		TCP	25									■			■		12.2	TCP	25									■			■		3833			44.6
13.1		TCP	80	■								■	■				13.2	TCP	80					■		■	■	■				■	4864			63.2
14.1		TCP	80									■					14.2	TCP	80									■					1398			16.6
15.1		TCP	5001					■		■					■		15.2	TCP	8020			■											2101			29.2
16.1	Exploit	TCP	25	■				■				■					16.2	TCP	25									■			■		4233	3206	1245	54.3
17.1		TCP	110		■		■					■					17.2	TCP	110									■			■		4755			63.4
18.1		TCP	25				■					■					18.2	TCP	25				■	■				■			■		3054			40.7
19.1		TCP	110	■			■	■				■			■		19.2	TCP	110	■				■				■			■		3373			46.2
20.1		TCP	80									■	■				20.2	TCP	80						■		■						2579			31.5
21.1		TCP	80					■				■					21.2	TCP	80		■								■				1243			16.1
22.1	Generic	TCP	80	■				■		■		■				■	22.2	TCP	80									■			■		5056	3464	1620	70.2
23.1		TCP	80							■		■	■				23.2	TCP	80			■							■				4305			51.3
24.1		TCP	25			■		■				■					24.2	TCP	80						■		■						3161			37.2
25.1		TCP	6503							■			■				25.2	TCP	21							■	■					■	1333			16.9
26.1	Fuzzer	TCP	179				■									■	26.2	TCP	80						■	■							4118	2294	1555	47.9
27.2		TCP	1723	■												■	27.2	TCP	179					■				■					1235			17.2
28.1		TCP	179												■		28.2	TCP	179	■								■					3034			42.1
29.1		TCP	445						■	■			■				29.2	TCP	179						■				■				790			9.9
30.1	Recon-naissance	UDP	111	■													30.2	UDP	111	■													175	139	28	2.4
31.1		UDP	111	■													31.2	UDP	111	■													137			1.8
32.1		TCP	111							■	■						32.2	TCP	111							■	■						98			1.3
33.1		TCP	111							■	■						33.3	TCP	111							■	■						136			1.7
34.1		TCP	111							■	■						34.2	TCP	111							■	■						147			1.9

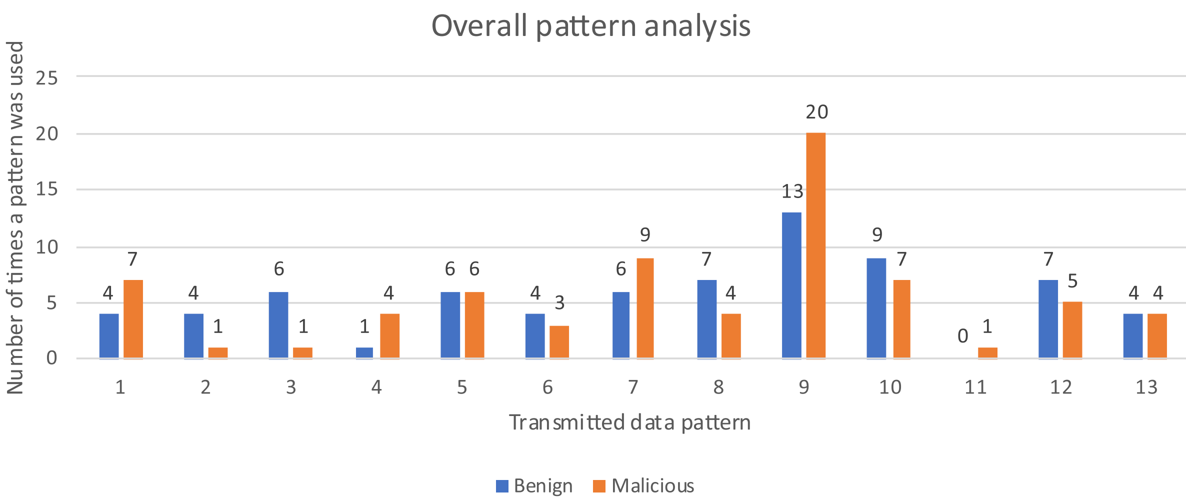

FIGURE 10. Overall pattern analysis.

(two unique fingerprints) with other fingerprints, further indicating the advancement of the proposed solution and its promising effectiveness in increasing the decision boundary between malware and benign classifications. Therefore, visual fingerprints can be developed for the transmitted data to differentiate between malicious and benign fingerprints.

4) RESULTS OF PROTOCOL DISCOURSE ANALYSIS

The protocol discourse section of the fingerprint consists of 128 values that range from −1500 to 1500, which has 384000 factorial permutations, resulting in infinite possibilities. From this analysis, it is clear that there are set packet ranges and phases that form patterns together. The focus of

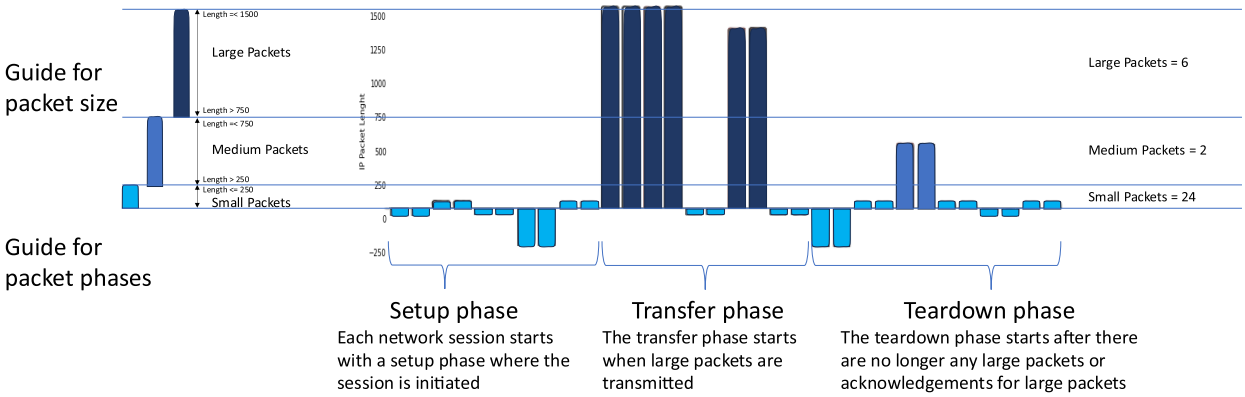

FIGURE 11. Protocol discourse packet guide.

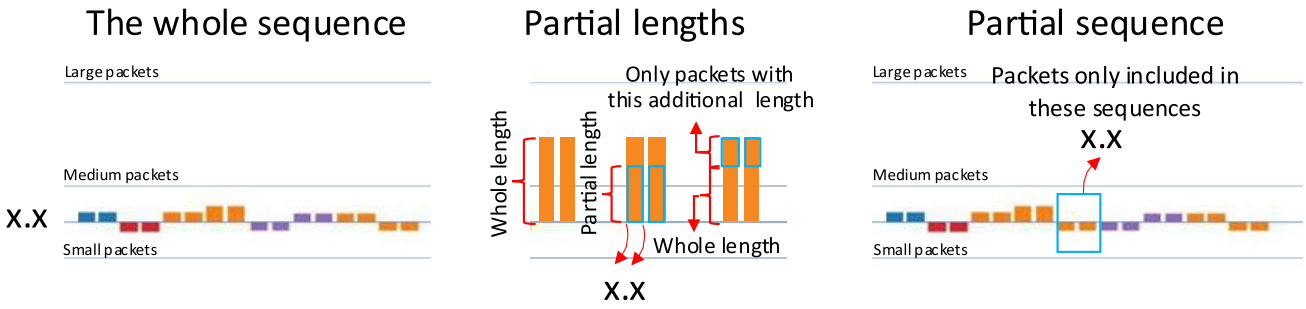

FIGURE 12. Protocol discourse annotation guide.

this subsection is to present the unique patterns, analysis of the protocol discourse for a few selected malicious and closest benign fingerprints, and broader findings uncovered during the analysis.

Two different pattern guides were used in the Protocol Discourse Results section. The first is a packet pattern guide that focuses on packet sizes and phases of engagement, and the second is a setup-phase packet length and sequence analysis for specific ports with repeating setup-phase patterns.

The following evaluation guide was used to identify unique patterns within the protocol discourse comprising the packet sizes of the phases. The guide for the different types of patterns is shown in Figure 11.

Three different phases were identified that corresponded to the typical flow of information and the sequence of events in a network session: setup, transfer, and teardown. In the example shown in Figure 11, there are ten small packets in the setup phase. Six large and four small packets were transmitted in the transfer phase, and the teardown phase contained ten small packets and two medium packets.

To illustrate and reveal patterns within the repeating set-up phase sequences, the following guide (Figure 12) was used to interpret the annotations, indicating how different sequences were combined.

In this example, three different protocol discourse setup phase sequences are overlaid onto one illustration on the left side of Figure 12. Using the annotation guide for the whole sequence, partial lengths, and partial sequences, three different sequences were identified, as depicted on the right-hand side of Figure 12.

5) PROTOCOL DISCOURSE PATTERN ANALYSIS

Table 2 presents the results of protocol discourse pattern analysis. The malicious and closest benign fingerprints were located in the same row to easily compare their pattern similarities. The columns include row numbers (#), threat categories, protocols, ports, and pattern guide results for malicious fingerprints, and row number (#), protocol, port, and pattern guide results for the closest benign fingerprints. In addition, the sum of differences, Frobenius distance measure, mean, standard deviation, and significance are included. The sum of the differences was included to highlight the overall differences, based on the protocol discourse guide.

The following are the conclusions from the results of the analysis in Table 2.

- In the reconnaissance section, it is clear that there is no difference between reconnaissance malware fingerprints and their closest benign fingerprints because all packets have the same sequence and packet size, except for fingerprint 33. In Fingerprint 33, the malicious fingerprint has the same number of transmitted packets, but the sequence is different, leading to a distance of 226 points.

TABLE 2. Protocol discourse analysis results.

Malicious Fingerprints											Closest Benign Fingerprints														
				Set up phase		Transfer phase			Teardown phase					Set up phase		Transfer phase			Teardown phase						
#	Threat	Protocol	Port	Small	Medium	Small	Medium	Large	Small	Medium	#	Protocol	Port	Small	Medium	Small	Medium	Large	Small	Medium	Sum of differences	Frobenius distance	Mean	Standard deviation	Significance
1.1	Backdoor	TCP	45947	16							1.2	UDP	43830	2							14	301	222	56	5.0
2.1	Backdoor	TCP	21554	18							2.2	TCP	80	18							0	176			3.7
3.1	Backdoor	TCP	47252	20							3.2	TCP	143	16							4	188			3.5
4.1	Backdoor	UDP	2140	4							4.2	TCP	80	16							12	222			4.6
5.1	Shellcode	TCP	18725	16							5.2	TCP	5582	16							0	7	125	167	0.1
6.1	Shellcode	UDP	17497	2							6.2	TCP	80	16							14	243			4.2
7.1	Worm	TCP	80	14	2						7.2	UDP	514	14	2						0	227	2072	3690	1.8
8.1	Worm	TCP	80	14	2						8.2	TCP	80	14	2						0	226			1.7
9.1	Worm	TCP	80	14	2						9.2	TCP	80	14	2						0	226			1.7
10.1	Worm	TCP	80	6	2	20	8	88			10.2	TCP	80	6	2	20	8	92			4	7607			124.7
11.1	DoS	TCP	80	6	2	20	8	92			11.2	TCP	80	6	2	20	8	92			0	2146	2236	1347	33.0
12.1	DoS	TCP	25	24		4	2	30	16		12.2	TCP	25	24		20	2	36	18		24	3890			4.4
13.1	DoS	TCP	80	6	2	2		14	8		13.2	TCP	80	6	2	2		16	8		2	3210			40.1
14.1	DoS	TCP	80	14	4						14.2	TCP	80	14	4						0	553			5.4
15.1	DoS	TCP	5001	14	2						15.2	TCP	8020	6			2	2	8		22	1380			5.3
16.1	Exploit	TCP	25	12		4		18	10		16.2	TCP	25	12		2		16	16		10	3608	2755	3012	5.0
17.1	Exploit	TCP	110	26		16	6	80			17.2	TCP	110	26		10	6	50	12		48	8096			176.0
18.1	Exploit	TCP	25	12		18	6	80			18.2	TCP	25	12		18	6	80			0	21			0.02
19.1	Exploit	TCP	110	26		2	2	20	12		19.2	TCP	110	26		4	4	26	12		10	3236			70.3
20.1	Exploit	TCP	80	14	14						20.2	TCP	80	14	2						12	1342			7.8
21.1	Exploit	TCP	80	14	2						21.2	TCP	80	14	2						0	227			1.7
22.1	Generic	TCP	80	10		2		20	8		22.2	TCP	80	14		2	2	16	8		10	3073	3310	2410	3.6
23.1	Generic	TCP	80	6	2	16	6	76	8		23.2	TCP	80	6	2	20	8	92			30	6081			36.0
24.1	Generic	TCP	25	12		2		6	8		24.2	TCP	80	6				2	8		12	3843			14.6
25.1	Generic	TCP	6503	22							25.2	TCP	21	14	2						10	243			3.0
26.1	Fuzzer	TCP	179	10		2		20	8		26.2	TCP	80	6		2		14	8		10	227	725	442	4.2
27.2	Fuzzer	TCP	1723	6		2		2	8		27.2	TCP	179	14	4						24	923			13.2
28.1	Fuzzer	TCP	179	6				2	8		28.2	TCP	179	6			4	2	8		4	1232			2.8
29.1	Fuzzer	TCP	445	18	4						29.2	TCP	179	18							4	519			7.9
30.1	Reconnaissance	UDP	111	2							30.2	UDP	111	2							0	0	45	101	0
31.1	Reconnaissance	UDP	111	2							31.2	UDP	111	2							0	0			0
32.1	Reconnaissance	TCP	111	18							32.2	TCP	111	18							0	0			0
33.1	Reconnaissance	TCP	111	18							33.3	TCP	111	18							0	226			4.1
34.1	Reconnaissance	TCP	111	18							34.2	TCP	111	18							0	0			0

This roughly aligns with the detection efficiency of 0.2% for the UNSW-15. In addition, all the reconnaissance network sessions used port 111, which was used for remote procedure calls.

- In the backdoor and shellcode sections, only small packets are exchanged and remain in the setup phase. Differences were observed in the number of packets exchanged in fingerprints 1, 3, 4, and 6, and all distances were nonzero. The average distances were 232 and 125 points with standard deviations of 56 and 167 points, respectively. The significance of the differences ranged from minor to moderate, except for fingerprint 5, which had minuscule significance.

From the results in Table 2, the protocol discourse analysis identified 14 fingerprints with no packet differences. These had an average distance of 288 points compared to fingerprints with one or more differences (20 of 34), with an average distance of 2473 points. Only four fingerprints had zero distances and zero packet differences, which were reconnaissance fingerprints (30, 31, 32, and 34) because the packet sequence and flags matched exactly.

In summary, except for four reconnaissance fingerprints (30, 31, 32, and 34), protocol discourse data and visual fingerprints can aid in differentiating malicious and benign fingerprints.

V. CONCLUSION AND FUTURE WORK

The AIECDS methodology discussed in this paper includes guidelines for the development of AI-enhanced cyber-defence systems. The focus was on extracting meaningful data and producing visualized fingerprints. This was achieved by designing a fingerprint that enabled the discovery of hidden patterns. Visually comparing malicious fingerprints with the closest benign fingerprints demonstrated a significant improvement in detecting malicious threats. Furthermore, the use of fingerprinted data and data visualization in cyber-defense systems can significantly reduce the complexity of the decision boundary and simplify the machine-learning models required to improve the detection efficiency, even for malicious threats with minuscule sample datasets.

Therefore, the contribution of this study is the improvement in the development of AI-enhanced cyber-defence systems. Furthermore, the application of AIECDS methodology is illustrated through a use case study for the discovery of cyber threats using fingerprinted data and visualized network sessions.

REFERENCES

[1] *ENISA Threat Landscape Report: July 2021 to July 2022*, Eur. Union Agency for Cybersecur. (ENISA), Athens, Greece, 2022.
[2] C. Gidi, "Vulnerability and threat trends report," Skybox Secur., San Jose, CA, USA, Tech. Rep., 2022.
[3] A. F. Police, "ACSC annual cyber threat report: July 2021 to June 2022," Austral. Criminal Intell. Commission (ACSC), Tech. Rep., 2022.
[4] R. Sobers. (May 2022). *89 Must-Know Data Breach Statistics 2022*. Accessed: Jun. 29, 2022. [Online]. Available: https://www.varonis.com/blog/cybersecurity-statistics
[5] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.
[6] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, and M. Xu, "A survey on machine learning techniques for cyber security in the last decade," *IEEE Access*, vol. 8, pp. 222310–222354, 2020.

[7] N. Kaloudi and J. Li, "The AI-based cyber threat landscape: A survey," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–34, Jan. 2021.
[8] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 8, pp. 3779–3795, Aug. 2021.
[9] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Comput. Netw.*, vol. 188, Apr. 2021, Art. no. 107840.
[10] J. K. F. Bowles, A. Silvina, E. Bin, and M. Vinov, "On defining rules for cancer data fabrication," in *Proc. Int. Joint Conf. Rules Reasoning*, in Lecture Notes in Computer Science, vol. 12173, 2020, pp. 168–176.
[11] Y. Lu, M. Shen, H. Wang, X. Wang, C. van Rechem, T. Fu, and W. Wei, "Machine learning for synthetic data generation: A review," 2023, *arXiv:2302.04062*.
[12] K. Shaukat, S. Luo, V. Varadharajan, I. Hameed, S. Chen, D. Liu, and J. Li, "Performance comparison and current challenges of using machine learning techniques in cybersecurity," *Energies*, vol. 13, no. 10, p. 2509, May 2020.
[13] A. Alshaibi, M. Al-Ani, A. Al-Azzawi, A. Konev, and A. Shelupanov, "The comparison of cybersecurity datasets," *Data*, vol. 7, no. 2, p. 22, Jan. 2022.
[14] A. Wu, Y. Wang, X. Shu, D. Moritz, W. Cui, H. Zhang, D. Zhang, and H. Qu, "AI4VIS: Survey on artificial intelligence approaches for data visualization," *IEEE Trans. Vis. Comput. Graphics*, vol. 28, no. 12, pp. 5049–5070, Dec. 2022.
[15] Threat Hunter Team. (Mar. 2022). *Daxin Backdoor: In-Depth Analysis*. Accessed: Dec. 20, 2022. [Online]. Available: https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/daxin-malware-espio
[16] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on GAN," in *Proc. Int. Conf. Data Mining Big Data*. Singapore: Springer, Nov. 2022, pp. 409–423.
[17] D. Kirat, J. Jang, and M. Stoecklin. (2018). *DeepLocker: Concealing Targeted Attacks With AI Locksmithing*. Black Hat USA. Accessed: Jul. 27, 2024. [Online]. Available: https://i.blackhat.com/us-18/Thu-August-9/us-18-Kirat-DeepLocker-Concealing-Targeted-Attacks-with-AI-Locksmithing.pdf
[18] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, p. e4150, Jan. 2021.
[19] K. Arshad, R. F. Ali, A. Muneer, I. A. Aziz, S. Naseer, N. S. Khan, and S. M. Taib, "Deep reinforcement learning for anomaly detection: A systematic review," *IEEE Access*, vol. 10, pp. 124017–124035, 2022.
[20] Y.-F. Hsu and M. Matsuoka, "A deep reinforcement learning approach for anomaly network intrusion detection system," in *Proc. IEEE 9th Int. Conf. Cloud Netw. (CloudNet)*, Nov. 2020, pp. 1–6.
[21] Ö. A. Aslan and R. Samet, "A comprehensive review on malware detection approaches," *IEEE Access*, vol. 8, pp. 6249–6271, 2020.
[22] K. Malialis, "Distributed reinforcement learning for network intrusion response," Doctoral dissertation, Dept. Comput. Sci., Univ. York, York, U.K., 2014.
[23] V. Duddu, "A survey of adversarial machine learning in cyber warfare," *Defence Sci. J.*, vol. 68, no. 4, p. 356, Jun. 2018.
[24] C. Klopper and J. Eloff, "Fingerprinting network sessions for the discovery of cyber threats," in *Proc. Int. Conf. Cyber Warfare Secur.*, Feb. 2023, vol. 18, no. 1, pp. 171–180.
[25] C. E. Heaney, Y. Li, O. K. Matar, and C. C. Pain, "Applying convolutional neural networks to data on unstructured meshes with space-filling curves," *Neural Netw.*, vol. 175, Jul. 2024, Art. no. 106198.
[26] D. A. Keim, "Pixel-oriented database visualizations," *ACM SIGMOD Rec.*, vol. 25, no. 4, pp. 35–39, Dec. 1996.
[27] Y. Lei, X. Tong, D. Wang, C. Qiu, H. Li, and Y. Zhang, "W-Hilbert: A W-shaped Hilbert curve and coding method for multiscale geospatial data index," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 118, Apr. 2023, Art. no. 103298.
[28] T. G. Eschenbach, "Technical note: Constructing Tornado diagrams with spreadsheets," *Eng. Economist*, vol. 51, no. 2, pp. 195–204, Jul. 2006.
[29] E. Weisstein. (2024). *Frobenius Norm*. Accessed: Jul. 25, 2024. [Online]. Available: https://mathworld.wolfram.com/FrobeniusNorm.html

CHRISTIAAN KLOPPER received the B.Eng. degree in electronic engineering from the University of Pretoria, South Africa, in 2010, where he is currently pursuing the master's degree in IT focusing on big data science. His main research interests include data science, big data analytics, and developing a self-learning cyber defense system that can discover undetectable threats.

JAN H. P. ELOFF was the Research Director of the SAP Research in Africa, from 2008 to 2015. From 2016 to 2021, he was the Deputy Dean Research, and the Acting Dean of the Faculty of Engineering, Built Environment and IT, University of Pretoria, South Africa, in 2022. He is currently a Full Professor in computer science with the University of Pretoria. He holds a B2 rating from the National Research Foundation in South Africa indicating that he receives considerable international recognition for his research in safeguarding platforms against societal and organizational cyber-threats. He is also a leading international scholar in conducting research on the convergence of cyber-security and AI. He has published widely in leading international journals. In 2018, he published a scholarly book on software failure investigations. He is the co-inventor of a number of patents registered in the USA. He is a member of the governing and advisory board of the International Knowledge Centre for Engineering Sciences and Technology (UNESCO(IKCEST)), China. During his research career, he represented South Africa as an Expert on Technical Committee 11 (Information Security) IFIP and was a recipient of the IFIP Silver Core and Outstanding Services Award. He also served as the South African Representative for the International Standards Organization (ISO) and as a former President of South African Institute of Computer Scientists and Information Technologists (SAICSIT). In 2017, he received a SAICSIT award recognizing him as an individual who has played a pioneering role in promoting computer science and information technology as academic disciplines in South Africa. In 2020, he received the Chancellor's Medal for Research from the University of Pretoria. He is listed as a finalist for the NSTF Lifetime Award for exemplary life-long research in cybersecurity, in 2021. He is an Associate Editor of Computers and Security, the world's leading journal for the advancement of computer security.