Journal of Applied Informatics and Computing (JAIC)
Vol.x, No.x, Juli 202x, pp. xx~xx
e-ISSN: 2548-6861

Forensic Extraction of Network Artifacts: A Wireshark-Based Analysis of CTF Scenario

Josep Lois Castilo Gultom [1]*, Nelmiawati [2]**
* Cyber Security Engineering Study Program, Politeknik Negeri Batam
** Informatics Engineering Department, Politeknik Negeri Batam
josep.4332001014@students.polibatam.ac.id [1], nelmiawati@polibatam.ac.id [2],

Article Info

Article history:

Received …
Revised …
Accepted …

Keyword:

Digital Forensic, CTF, Network Analysis, SQL Injection

ABSTRACT

Network forensics is a branch of digital forensics concerned with monitoring and capturing traffic to uncover digital crime. The purpose of this report is to isolate network forensics artifacts (e.g. covert payload and data exfiltration) from Capture The Flag (CTF) network traffic PCAPs, which will be used to build a complete picture of attack events. Although attackers may hide their malware commands and control in plain sight via network traffic, post-incident forensic analysis may allow investigators to study the captured packet data and identify malicious activity. A post-incident analysis was executed on a simulated CTF network traffic PCAP file with the network packet analyser Wireshark. Post-incident forensic analysis was applied by performing statistical analysis to identify and filter network traffic and reconstructing TCP streams to extract specific malicious packets from network noise, uncover attacks, and the impact of a breach. This analysis involves the usage of Wireshark display filter expressions, statistics capability features, and more. This paper proves that with post-incident forensic analysis techniques, an investigator will be able to identify and isolate the network forensic artifacts, construct an event timeline, and gain an accurate understanding of the incident.

I. INTRODUCTION

Simply cloning a hard drive post-intrusion falls short because attackers utilize fileless malware and encrypted staging servers, leaving no permanent footprint on the physical hardware. Therefore, an incident response must also rely on live, in-motion data interception, typically in the form of Packet Capture (PCAP) files [1]. During an active incident, this data is captured as it traverses the network, preserving information that would otherwise disappear once the host is powered down. Every single instance of exfiltration, payload deliverance, or lateral movement generates data within these packets. By examining the PCAP, investigators can identify plaintext sensitive data, network sessions, malicious payloads, and determine the identity of active Command and Control servers or sources of attack, and other artifacts important to the investigation [2].

Capture The Flag (CTF) environments replicate this exact scenario. In order to survive within these environments, a defender needs the ability to identify hidden artifacts inside of massive packet captures. Attackers often attempt to camouflage malware commands within standard HTTP or DNS traffic, which are both normal parts of typical network traffic. Wireshark can be used to filter out the noise. Wireshark is a network protocol analyzer that displays raw packet traffic and interprets the logs into a structured format for the user [3].

Although, in order to create an appropriate display filter in Wireshark, one must have a general understanding of the TCP/IP protocol suite. Once an appropriate display filter has been created, Wireshark can easily display a packet from the network traffic with the characteristics of the malware artifact, distinguishing it from normal, background, network chatter, and other traffic. This type of context cannot be obtained using previous command-line-based sniffers.

Wireshark has a user friendly, graphical interface, and it supports hundreds of protocol dissectors. It continues to update with the addition of new protocols and standards. It is compatible with the Windows, Linux, and macOS operating systems, and its ability to correctly reconstruct a stream of packets makes it an incredibly valuable tool for forensic investigators [4]. This study will examine Wireshark's

e-ISSN: 2548-6861

functionality as a packet capturing tool with a network packet capture dataset retrieved from a Capture the Flag challenge available from CyberDefenders CTF Challenges.

CyberDefenders is a blue-team training platform that presents incident responders with realistic cyber attack scenarios. For our secondary dataset, we have used a retired Capture the Flag (CTF) Challenge exercise on CyberDefenders which provides a ground-truth PCAP file with pre-captured, documented malware activity. [5]

A. Research Focus

Building upon this problem, this paper explored the issue of discriminating between malicious and normal network activity. This paper seeks the following questions:

1. How post-incident network forensics approach is used to filter malicious network artifacts from large PCAP files?
2. How a deep packet inspection analysis of statistical analysis and filtering through Wireshark is used to isolate network artifacts?

B. Research Objective

This paper focuses on the extraction and identification of network artifacts, malicious indicators resulting from malware within a PCAP file. This research seeks to utilize a forensic workflow, consisting of data collection, analysis, and documentation, to isolate malicious traffic and document a timeline of the attacker's activity. The resulting extractive analysis will measure the overall impact of a network incident by reconstructing network sessions to evaluate whether the attack was successful and determine the extent of exfiltration in the victim based on the CTF scenario acquired.

C. Scope and Limitations

This paper has been limited to a simulation analysis conducted in an offline post-incident forensics lab configuration. The only ground-truth traffic capture dataset that has been used for this analysis is a single packet capture file, acquired from a retired CTF Challenge. For protocol analysis and network reconstruction purposes, the only network analysis tool that has been used in this study is Wireshark. Finally, this study has focused primarily on artifacts that are only present when the network data traffic consists of plaintext traffic, which in this case means plain HTTP. It should be noted, this study was not intended to include any analysis of network traffic that contains modern levels of encryption, because Wireshark is not able to automatically decrypt this traffic unless an encryption tool with the appropriate key is utilized in tandem.

II. LITERATURE REVIEW

A. Packet Capture

Packet Capture (PCAP) is essentially the process of capturing and recording data in real time as it flows through a network. Capturing this real-time data is important because when an infected device is shut down, traces of data leaks, attacks, and lateral intrusion routes disappear immediately. Forensic experts meticulously analyze PCAP files to uncover plaintext passwords, malware, communication logs, and even the currently active Command and Control servers [6].

B. Capture the Flag (CTF)

Capture The Flag (CTF) environments are realistic, simulation cybersecurity platforms designed to simulate real-time cyber attack scenarios, wherein participants, who represent the defenders, learn how to detect adversarial artifacts embedded in millions of network packet captures. These scenarios may be used to serve as ground-truth datasets of malware activity, with all activities documented for investigative study and analysis [7].

C. Network Protocol Analyzer

A protocol analyzer is a software application that shows the raw packet traffic, as well as displays the log in organized format and is readable for user. Wireshark is a well-known protocol analyzer that provides a graphical user interface with many hundreds of built-in protocol dissectors. It is actively maintained, meaning the application is updated to provide new protocols and protocols standards as they are defined and developed, while retaining a compatibility to run on a variety of platforms, including Windows, Linux, and macOS operating systems. One of the distinguishing features of a protocol analyzer, as compared to the command line-based sniffers is the ability to build or rebuild a packet stream in an intelligent fashion to give additional context, which provides a significant benefit for the forensic investigator who seeks to distinguish signal from noise, as well as isolate the artifacts of malicious activity [8].

D. Forensic Artifacts

Forensic artifacts are the digital evidence or IOCs created as part of adversary action in a network capture file. Since adversaries are using fileless and encrypted staging servers, there may not be any artifacts left on a physical device, which makes network artifacts extremely important. By examining data as it moves across the network, we can extract sensitive data in plaintext, network sessions, malware payloads, data exfiltration channels, and active Command and Control servers, which are all crucial to creating a comprehensive and accurate timeline of the adversary's actions during the event [9].

E. Post-Incident Forensic Analysis

In post-incident forensic analysis, an investigator will capture data when it is first noticed that a security breach has occurred and then look back at the stored packets to identify the suspicious behavior that would have likely been erased from the compromised system if the system were rebooted or if the machine were turned off. In this way, the offline

analysis will isolate the malicious traffic from what would otherwise be classified as normal or harmless traffic. The typical workflow for post-incident forensic analysis involves a specific and sequential series of forensic steps that are designed to understand how the attack occurred, what vulnerabilities were exploited, and what data or systems were compromised [10].

III. METHODOLOGY

This study utilizes a holistic, post-event network forensic analysis of a Packet Capture (PCAP) file created from a Capture The Flag (CTF) scenario. Using Wireshark to perform a deep-dive packet inspection, the analysis is conducted offline and follows an outlined forensic process of statistical profiling, traffic filtering, and TCP stream reconstruction. This method was implemented to determine the malicious or benign hosts in the PCAP file, extract and identify any malicious activities and indicators, establish the time sequence of the attacker activity, and ultimately decide if the attack was successful and to what extent data was exfiltrated from the victim(s) [11].

A. Environment Setup

An offline environment specifically designed for post incident digital forensics is used to investigate this event. In this forensic environment, Wireshark is installed on the local forensic workstation which provides deep packet analysis, protocol dissectors, and the TCP stream reassembly capabilities needed for the analysis. [12] The environment is intended to parse and investigate a single dataset, the WebInvestigation.pcap from a CyberDefenders CTF Challenges, that is now retired and available for public use [13].

Table 1. Environment Detail

No	Component	Specification/Version
1	Analysis Workstation	- OS: Windows - Compute: 8 GB RAM
2	Analysis Software	Wireshark Version 3.6.8
3	Dataset	PCAP file acquired from CyberDefenders retired CTF Challenges

B. Analysis

The PCAP file will be analyzed with a systematic forensic pipeline; since we are working with historical data and this study does not conduct live testing, our focus will be on extracting the relevant forensic artifacts, i.e., data evidence.

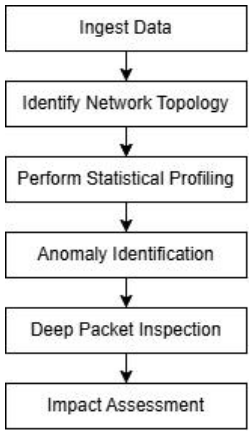

Figure 1. Network Analysis Pipeline

Analysis begin by examining the top level of the PCAP and determining baseline statistics regarding endpoints and communication protocols to identify outliers and anomalies in communications and to determine the most prevalent communication protocol [14].

The analysis then delves into the PCAP to analyze packets on a lower level in the TCP/IP protocol stack. Wireshark display filters will be constructed to analyze isolated objects. Various features of Wireshark will also be used as needed by the analysis. This will identify the attack being performed in the scenario (i.e., SQL Injection attack) [15].

Next, we follow the network stream of the attack to completion, using both the request and response data to correlate and analyze the corresponding activites. The evidence from this analysis will be sufficient to determine the impacts of the potential breach and evaluate the effectiveness of Wireshark to identify malicious activities visible in the network.

IV. RESULT AND ANALYSIS

A. Import Packet Capture File into Wireshark

First, the packet capture file to be analyzed is imported into Wireshark through the Open Capture File feature and selected. In this case, it is WebInvestigation.pcap file.

e-ISSN: 2548-6861

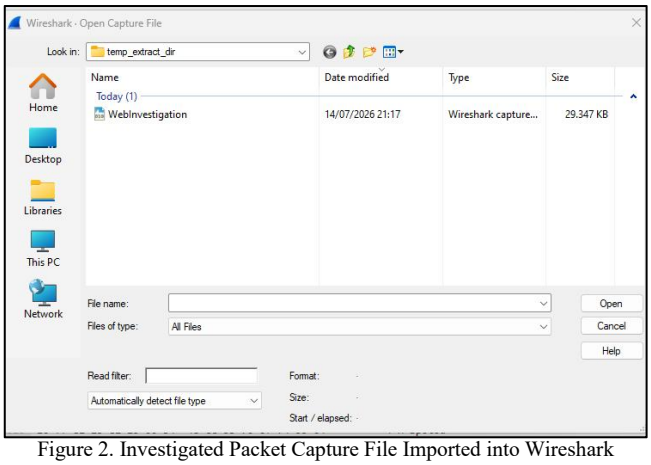

Figure 2. Investigated Packet Capture File Imported into Wireshark

B. *Identify Endpoints in the Network*

Then, to start isolating the evidence, topology of the network's endpoints must be identified. From the Endpoint Statistics feature of Wireshark, we get a list of all the IP addresses included in the capture and sorted by traffic usage. We can see there are two clear outliers from the rest of the network: 73.124.22.98 and 111.224.250.131. The unusually large one-way transfers in and out from these two IPs point towards a potential malicious activity being present, so we can use them as a jump off point to look closer.

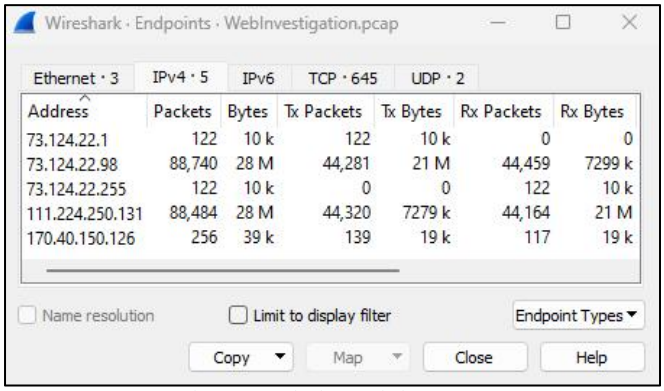

Figure 3. Endpoint Statistic Feature

C. *Identify Patterns in the Network's Protocol Statistics*

The analysis is then carried out on the protocol level of the packets captured. This analysis begins by displaying the statistics based on the Protocol Hierarchy Statistics feature in Wireshark. It was found that 93.9% of the packets and 79.5% of the data captured were HTTP. This may be indicative of web activity, requiring us to then inspect the HTTP packets more closely. Such activity would allow for the possibility of the HTTP streams being malicious with SQL injection attacks or the execution of the payload.

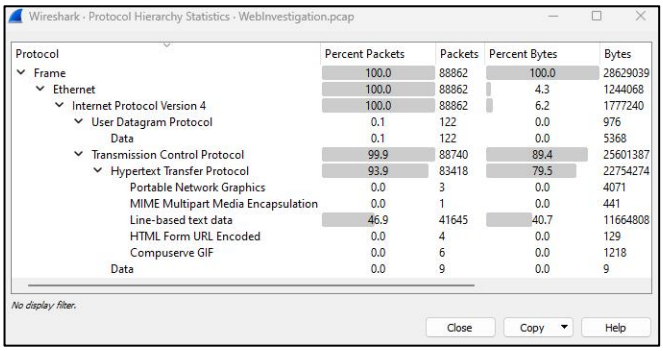

Figure 4. Protocol Hierarchy Statistics Feature

D. *Apply Filter Expressions to Investigate Network Communication Between Isolated Endpoints*

Then, a filter expression is used to exclusively include packets from or to 111.224.250.131 and the HTTP protocol, in order to isolate the investigated network traffic. Looking at this filter results, show a lot of http GET requests from 111.224.250.131 to 73.124.22.98. The GET request to search.php URL is identified to have a Structured Query Language injection (SQLi) [16]. In the query, SQLi is used to create a mathematical condition which is always logically true. This may allow a SQLi attack to access unauthorized databases as such condition can be exploited to bypass restrictions or cause intended effects. Confidently, we can state that 111.224.250.131 was the attacker to web server 73.124.22.98, and this activity targeted the search.php feature.

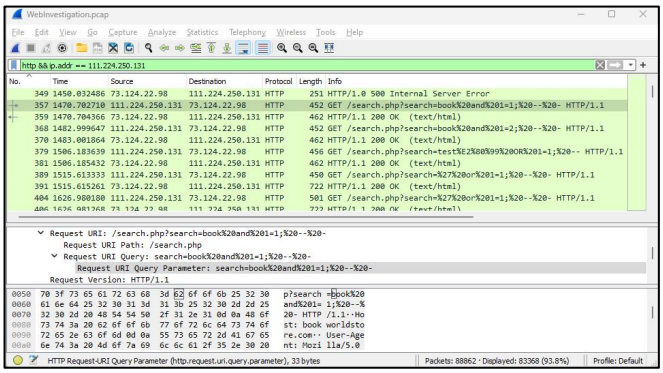

Figure 5. Filter Expression Feature

E. *Analyze Payloads of the Malicious Traffic*

Continuing with analysis of the exploit phase of the web server, the analysis is further refined by only viewing the HTTP GET requests that communicate with the search.php URI. Upon this filtration, many instances of encoded, URL parameters are observed which indicate the utilization of SQL injection techniques.

After further refinement to only these encoded URL parameters using the HTTP Object List feature, the attacker can be examined to transition to a UNION SELECT based SQL injection attack. Each payload injects the UNION SELECT keywords which allows the injection of a query that combines the original query result with the results

obtained from the selected database [17]. The use of sequential database and table names can also be observed in the query as the attacker attempts to get more information about the database being targeted. This is a common technique used to extract data from any table on the database. These payloads are also being partly encoded to prevent signature-based IDS system's detection.

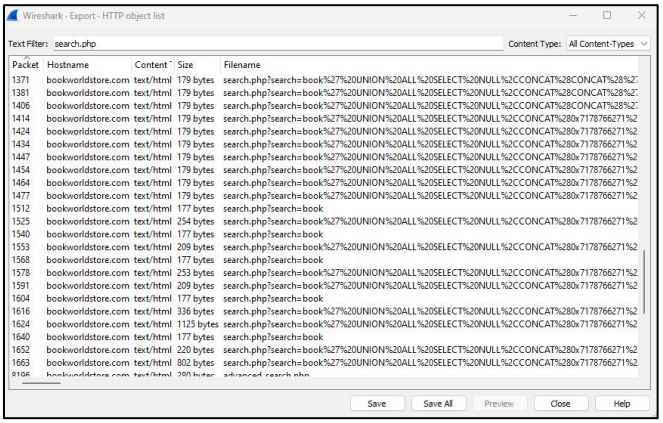

Figure 6. HTTP Object List Feature

F. Assess Impact of the Malicious Traffic

By further analyzing the HTTP server responses, impact of the breach is able to be determined. Correlating the malicious HTTP GET requests to the server response code 200 OK instances provide full visibility into the attacker's successful data extraction.

In the later stages of the attack, the attacker successfully enumerated the available databases using the INFORMATION_SCHEMA.SCHEMATA view. The response revealed all of the default schemas for mysql, information_schema, performance_schema, and sys, as well as the application database bookworld_db. Then the attacker targeted the bookworld_db database to identify a table named customers.

Following the HTTP communication stream through the Follow feature, it is then identified that the attacker was able to exfiltrate rows of sensitive user records from this table, definitively exposing Personally Identifiable Information (PII), including user names, email addresses, and contact details.

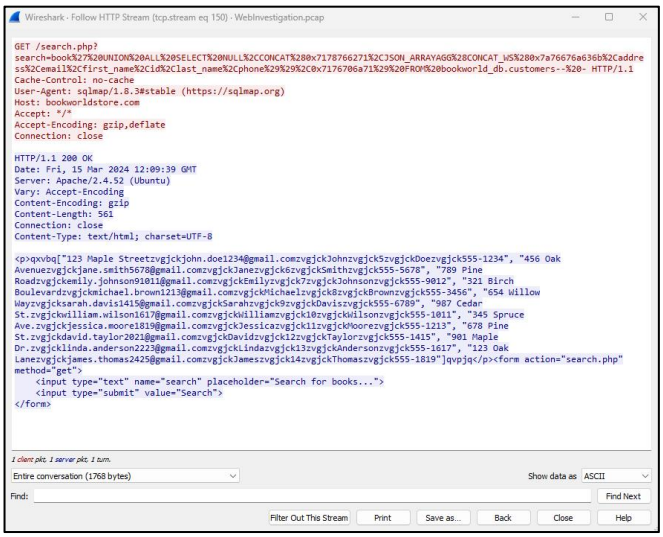

Figure 7. Follow HTTP Stream Feature

V. CONCLUSION AND RECOMMENDATION

A. Conclusion

Post-incident network forensic analysis was conducted on the CTF dataset using a forensic pipeline, which allowed for the successful extraction of malicious network artifacts from the large PCAP file. By applying this pipeline through an offline analysis, starting with data ingestion, moving to network topology identification, then identifying malicious behavior, and finally determining the impact on the network, the network artifacts belonging to the attacker were extracted from the network artifacts of other benign traffic on the host.

Through the use of packet analysis (DPI) and the use of packet filter expressions, the network artifacts associated with SQLi attacks were isolated from the packet trace data. Preliminary statistical profiling of the PCAP file showed that the potential anomalous hosts and their network protocol of interest. This information was applied to the filter expressions to further refine the results of the packet analysis to identify the particular attack and isolate it. An in-depth examination uncovered how the adversary managed to obfuscate their commands to perform covert enumeration and indexing of the internal datasets.

Our analysis showed that the attacker was able to exfiltrate data about customers, and with our instrumentation we reconstructed the entire dialogue, aligning the covert commands to the successful responses. Reviewing the complete dialogue gave us the precise scope of the data leak, enabling us to comprehend the scope and impact of the attack and construct a chronological account of the attack, successfully completing our research objectives of assessing the attack's impact.

B. Recommendation

Although we demonstrated the ability to pull hidden payloads with this methodology, there are obvious limitations in the analysis tool and the nature of our data that

e-ISSN: 2548-6861

we didn't explore. In our study, the network artifacts we were able to extract depended entirely on clear text data; that is to say, HTTP. Wireshark, as a passive analyzer tool, cannot decipher modern encryption on its own. To do so would require outside decryption tools or explicit Transport Layer Security (TLS) session keys [18].

The next step would be to figure out how to extract the same network artifacts but from encrypted traffic. In similar security education and network analysis studies in the future, it is important to compare findings of a hybrid methodology that pairs deep packet inspection with an automated Intrusion Detection System (IDS), like Suricata. This would let us compare the ways an IDS would alert or parse out the CTF data compared to manual extraction.

Although having a CTF dataset as our ground-truth sample on a closed network made for an effective test bed, this data did not represent a network running millions of connections like a live environment does. Thus, in future research applying these network artifact extraction techniques, a large enterprise PCAP file would be beneficial to test for how much of a performance lag graphical protocol analyzers might incur on significantly larger datasets.

REFERENCES

[1] M. Naas and J. Fesl, “A novel dataset for encrypted virtual private network traffic analysis,” Feb. 01, 2023, Elsevier. doi: 10.1016/j.dib.2023.108945.M. Naas and J. Fesl, “A novel dataset for encrypted virtual private network traffic analysis,” Feb. 01, 2023, Elsevier.

[2] M. Guven, “Dynamic Malware Analysis Using a Sandbox Environment, Network Traffic Logs, and Artificial Intelligence.,” IJCESEN, vol. 10, no. 3, Sept. 2024

[3] C. Sanders, “Practical Packet Analysis: using Wireshark to solve real-world network problems,” Network Security, vol. 2011, no. 8, p. 4, Aug. 2011

[4] G. Jain and A. Anubha, “Application of SNORT and Wireshark in Network Traffic Analysis,” IOP Conf. Ser.: Mater. Sci. Eng., vol. 1119, no. 1, p. 012007, Mar. 2021

[5] Y. Nagare et al., “From Simulation to Application: The Role of CTF Competitions in Cybersecurity Training,” in 2025 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), Gwalior, India: IEEE, Mar. 2025, pp. 1–6.

[6] R. K. Jolankarfyan, "Network Forensics of Contact Forms BazarLoader and Cobalt Strike Based on PCAP File Analysis," International Journal of Cyber Engineering, vol. 1, no. 1, pp. 28–37, 2025.

[7] S. Karagiannis and E. Magkos, “Adapting CTF challenges into virtual cybersecurity learning environments,” ICS, vol. 29, no. 1, pp. 105–132, May 2021.

[8] J. Beale, A. Orebaugh, and G. Ramirez, "Wireshark & Ethereal Network Protocol Analyzer Toolkit," Elsevier, 2006.

[9] M. Asassfeh et al., "An Overview of Tools and Techniques in Network Forensics," 2024 25th International Arab Conference on Information Technology (ACIT), Zarqa, Jordan, 2024, pp. 1-7

[10] A. G. AlBoloushi, F. A. Al-Meer, F. Nawshin, and D. Unal, “Network Forensics: Techniques, Challenges, and Incident Response,” in Information System Design: Communication Networks and Internet of Things, vol. 1538, V. Bhateja, Z. Reza Khan, M. Simic, and D. K. Sharma, Eds., in Lecture Notes in Networks and Systems, vol. 1538. , Singapore: Springer Nature Singapore, 2026, pp. 51–62. doi: 10.1007/978-981-96-9245-3_5.

[11] M. Macak, M. Stovcik, T. Rebok, M. Ge, B. Rossi, and B. Buhnova, “CopAS: A Big Data Forensic Analytics System,” Apr. 03, 2023

[12] I. V. Ticovan and G. Sebestyen, “Forensic Data Analysis Pipeline,” Acta Univ. Sapientiae Inform., vol. 17, no. 1, p. 3, Jul. 2025

[13] "Web Investigation Lab," CyberDefenders. [Online]. Available: https://www.techinnovations.com/quantum-computing. [Accessed: July. 13, 2026].

[14] M. Ahmed, A. Naser Mahmood, and J. Hu, “A survey of network anomaly detection techniques,” Journal of Network and Computer Applications, vol. 60, pp. 19–31, Jan. 2016

[15] R. Sarvaiya, Y. Trivedi, and R. Pal, “Capturing and Analysis of Data Packets Through Wireshark,” in Advances in VLSI, Communication, and Signal Processing, vol. 1457, R. A. Mishra, S. K. Gupta, V. K. Srivastava, and J. Mäkelä, Eds., in Lecture Notes in Electrical Engineering, vol. 1457. , Singapore: Springer Nature Singapore, 2025, pp. 225–237.

[16] V. Abdullayev and A. S. Chauhan, “SQL Injection Attack: Quick View,” Mesopotamian Journal of CyberSecurity, vol. 2023, pp. 30–34, Feb. 2023

[17] R. W. Setiawan and W. M. Ashari, “Infiltrasi Union: SQL injection untuk Ekstraksi Kredensial Admin,” ijcs, vol. 12, no. 6, Dec. 2023

[18] A. K. Cherukuri, S. T. Ikram, G. Li, and X. Liu, “Encrypted Network Traffic Analysis,” in Encrypted Network Traffic Analysis, in SpringerBriefs in Computer Science. , Cham: Springer International Publishing, 2024, pp. 19–45.