IJCE; INTERNATIONAL JOURNAL OF CYBER ENGINEERING

E-ISSN: XXXX-XXXX
Vol. 01, No. 01, December, 2025, PP: 01-12
DOI:

ARTICLE

Analysis of Async RAT Malware Infection through Network Communication Using Wireshark

Prasetyo Mimboro [1,*], Mochammad Luthfi Rahmadi[2]

[1,2] Department of Informatic, Muhammadiyah Cyber University, Yogyakarta, Indonesia

***Corresponding author:** prasetyo20220100098@sibermu.ac.id

Abstract

This study investigates the infection process of the Async Remote Access Trojan (Async RAT) and demonstrates how its malicious activity can be identified through network traffic analysis using Wireshark. Async RAT is a remote access malware that enables adversaries to gain complete control over compromised systems. The case study utilizes PCAP files and infection artifacts obtained from malware-traffic-analysis.net to reconstruct the infection chain, beginning with the execution of a suspicious ISO file containing a WSF script and culminating in the malware's communication with Command and Control (C2) servers. The findings reveal that Async RAT employs both HTTP and HTTPS protocols to retrieve payloads and maintain interaction with remote servers. Furthermore, Indicators of Compromise (IoCs)—including domains, IP addresses, and file hashes—were identified and validated to provide insight into the attack vector. This research contributes to the broader understanding of malware infection patterns and supports the development of effective detection and mitigation strategies in cybersecurity practice.

Keywords: **Async RAT, malware, Wireshark, Command and Control, network infection, indicators of compromise**

1. Introduction

Network security is a critical element in safeguarding the integrity, confidentiality, and availability of digital data. One significant threat is the Remote Access Trojan (RAT), such as Async RAT, which enables attackers to gain remote control over compromised systems. This study analyzes the infection process of Async RAT based on network log data and malware artifacts to understand attack patterns and the malware's communication methods with Command and Control (C2) servers. information security has become increasingly crucial in the digital era, where the internet connects diverse sectors and facilitates the exchange of personal and organizational data (Ramdan et al., 2022). The widespread use of internet-based technologies introduces new challenges in protecting personal information, particularly during collection, utilization, and distribution. Cyberattacks, including fraud exploiting technological vulnerabilities, pose serious risks to institutions and organizations that rely on computer networks for operational activities (Parulian et al., 2021; Ikhwanul Uzlah & Saputra, 2024; Rabbani & Diana, 2023).

Growing dependence on digital technologies amplifies vulnerabilities in information security, making cyber protection a strategic priority to defend against evolving threats (Sandriana & Maulana, 2022; Afifah Rodhiyatun Nisa et al., 2024; Sutra & Haryanto, 2023). Malware continues to advance in complexity, capable of infiltrating systems, stealing sensitive data, and disrupting network operations (Pajar Setia et al., 2018). Effective tools and methods are therefore required to detect and mitigate such threats (Nugroho & Prayudi, 2015). Wireshark, a widely used network analysis tool, enables packet capture and inspection of data traffic. Its capabilities include monitoring multiple protocols, filtering device communications, and detecting anomalies indicative of malware activity. By observing abnormal traffic patterns, analysts can identify malware presence and gain insights into its operational methods. This research aims to evaluate the use of Wireshark in detecting malware by analyzing suspicious network communication patterns. The study includes forensic analysis of Async RAT infection using PCAP files obtained from the Malware Traffic Analysis platform. The results provide a comprehensive forensic report, identification of Indicators of Compromise (IoCs), and reconstruction of the attack chain based on analyzed network logs.

2. Literature Review

2.1. Malware

Malware, or malicious software, refers to any type of program intentionally designed to compromise computer systems and violate security policies, particularly those related to confidentiality, integrity, and availability of information. It can also be defined as code that performs harmful actions. Malware exists in diverse forms, including scripts, executable files, and other digital objects, and can spread through multiple vectors such as websites, email messages, or external storage devices like USB drives (Mariyah Fairuz et al., 2021).

2.2. Async RAT

Async RAT is a .NET-based malware that enables attackers to gain full control over compromised systems, including data theft and the installation of additional malicious software. As an open-source Remote Access Trojan (RAT), it is frequently modified by threat actors to perform harmful activities such as data exfiltration, keylogging, screenshot capture, and direct command execution via terminal access. The malware is commonly distributed through phishing emails, malicious attachments, or deceptive websites designed to trick users into downloading and executing the infectious payload. Its primary strength lies in its ability to operate covertly in the background while maintaining encrypted communication with attacker-controlled Command and Control (C2) servers, making detection by traditional security systems challenging. From a network forensic perspective, Async RAT activity can be identified through abnormal communication patterns, suspicious domain or IP connections, and the use of specific ports characteristic of C2 traffic. These indicators provide valuable insights for detecting and mitigating the presence of Async RAT in compromised environments.

2.3. Wireshark

Wireshark is a network analysis tool used to capture and examine data traffic in real time, supporting a wide range of protocols and display filters. Formerly known as Ethereal, it was first developed by Gerald Combs in 1988 and has since become a standard application for network troubleshooting and analysis. The software runs on multiple operating systems, including Windows and UNIX, and is pre-installed on several Linux distributions such as Kali Linux. Wireshark supports both command-line (CLI) and graphical user interfaces (GUI), providing detailed insights into network activities down to the packet level. Due to its comprehensive capabilities, Wireshark is widely adopted across educational institutions, commercial organizations, and non-profit entities (Iqbal & Naaz, 2019).

2.4.Previous Research

Sikos (2020) emphasizes that packet analysis enables traceback of network activity and can reveal malware infections, data breaches, and Command and Control (C2) communications. Wireshark is highlighted as a primary tool due to its ability to capture and display network packets in detail. Zhang et al. (2023) propose a machine learning approach based on Graph Attention Networks (GAT) to detect encrypted malicious traffic. While Wireshark supports manual inspection, their work provides a useful comparison to traditional methods such as manual IoC identification in detecting malware communications like Async RAT. Barik et al. (2021) identify Wireshark as a widely used forensic tool, supporting both graphical and command-line interfaces (tshark), which facilitates manual analysis of Async RAT infections through network traffic. Hilgert et al. (2024), however, note limitations in Wireshark's ability to extract files from traffic compared to more comprehensive approaches that reconstruct entire file systems from PCAPs, including metadata and hierarchical structures. This supports the argument for broader perspectives in understanding malware communication patterns.

Research on network forensics for malware analysis has been extensive. Luber and Schiller (2020) focus on detecting Remote Access Trojans (RATs) through anomaly analysis of network traffic, proposing machine learning methods to classify hidden C2 communication in encrypted flows. Singh and Singh (2017) demonstrate both static and dynamic malware analysis techniques as foundational approaches in cybersecurity incident investigations. Specifically regarding Async RAT, industry reports such as Zscaler (2021) describe its evolution and tactics, including the use of dynamic DNS services and file camouflage techniques. Collectively, these studies underscore the importance of network traffic analysis as both a proactive and reactive strategy against modern malware threats, forming the basis for the case study presented in this paper.

3. RESEARCH METHOD

This study applies a case study design with network forensic analysis using PCAP and malware artifacts (ISO, WSF, and scripts) obtained from malware-traffic-analysis.net. Wireshark is used to inspect network communications, extract Indicators of Compromise (IoCs), and support infection reconstruction. IoCs are validated by cross-referencing file hashes, domains, and IP addresses with trusted threat intelligence sources. The methodology stages are summarized in Figure 1.

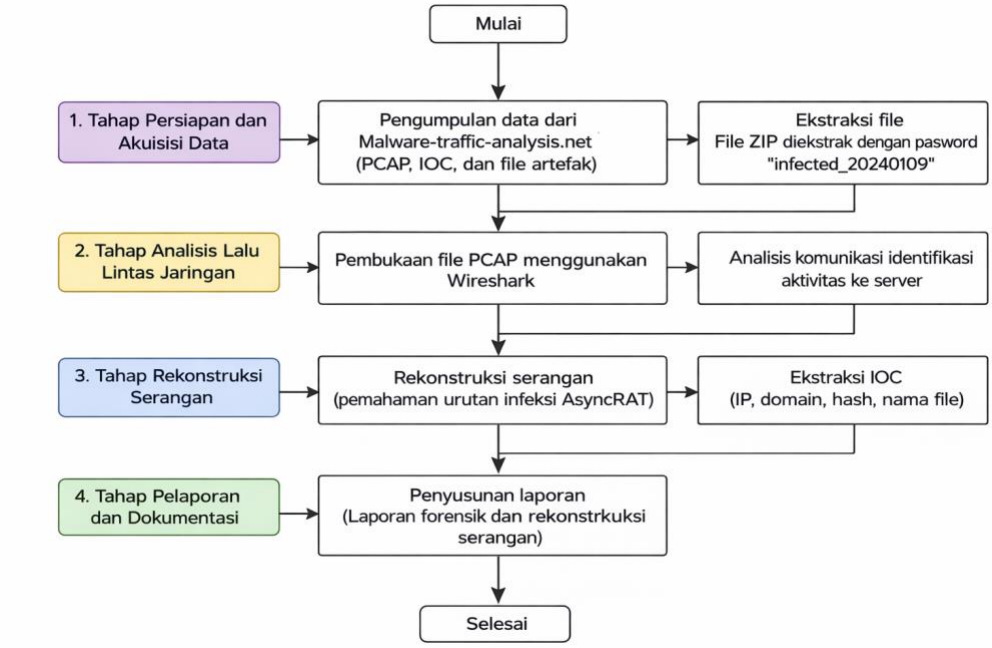

Figure 1. Flow diagram of the methodology for AsyncRAT network forensic analysis.

Figure 1 presents a four-stage methodology for AsyncRAT network forensic analysis, starting from data acquisition and ending with reporting. The process begins by collecting PCAP files, IOCs, and related artifacts from Malware-traffic-analysis.net, followed by extracting the ZIP archive using the provided password to enable access to the evidence set. The workflow then opens the PCAP in Wireshark to examine network sessions and to identify

communication patterns that indicate suspicious activity directed to a server. Next, the method reconstructs the attack by deriving the infection sequence of AsyncRAT from observed traffic artifacts and temporal event relations. In parallel, the analysis extracts actionable indicators of compromise, including IP addresses, domains, file hashes, and filenames, to support detection and correlation. The methodology concludes by compiling a structured forensic and attack reconstruction report that consolidates findings, extracted indicators, and the inferred sequence of events.

4. IMPLEMENTATION AND RESULTS

4.1. Reconstruction of the Infection Process

This section reports the results of a network forensic analysis of an AsyncRAT infection using PCAP data and supporting artifacts obtained from the Malware Traffic Analysis repository. The analysis identifies infection stages, examines communication with command-and-control (C2) infrastructure, and extracts Indicators of Compromise (IoCs) observed during the infection process through systematic packet-level inspection in Wireshark, as shown in Figure 2.

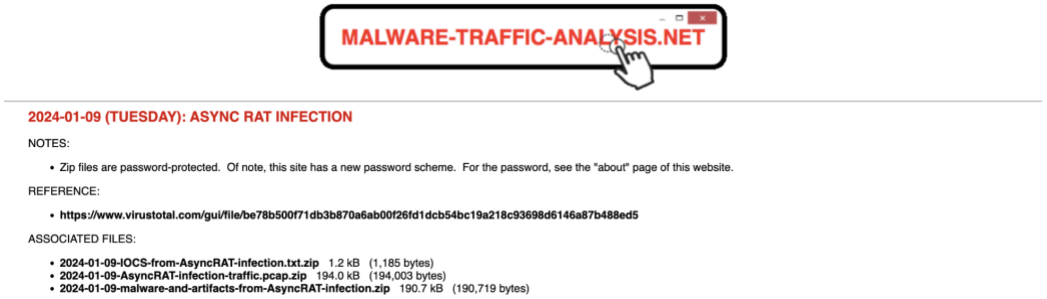

Figure 2. Data sources.

Figure 2 shows that the dataset was obtained from malware-traffic-analysis.net (2024) and consists of three ZIP archives containing distinct incident artifacts. The archive 2024-01-09-IOCs-from-AsyncRAT -infection.txt.zip (1.2 kB) provides IoCs generated during the infection process. The archive 2024-01-09-AsyncRAT -infection-traffic.pcap.zip (194.0 kB) contains the PCAP trace used for packet-level analysis. The archive 2024-01-09-malware-and-artifacts-from-AsyncRAT-infection.zip (190.7 kB) contains malware samples and execution artifacts. After extraction, the evidence set includes the PCAP file, IoC records, and malware artifacts such as xlm.txt and mdm.jpg, as shown in Figure 3.

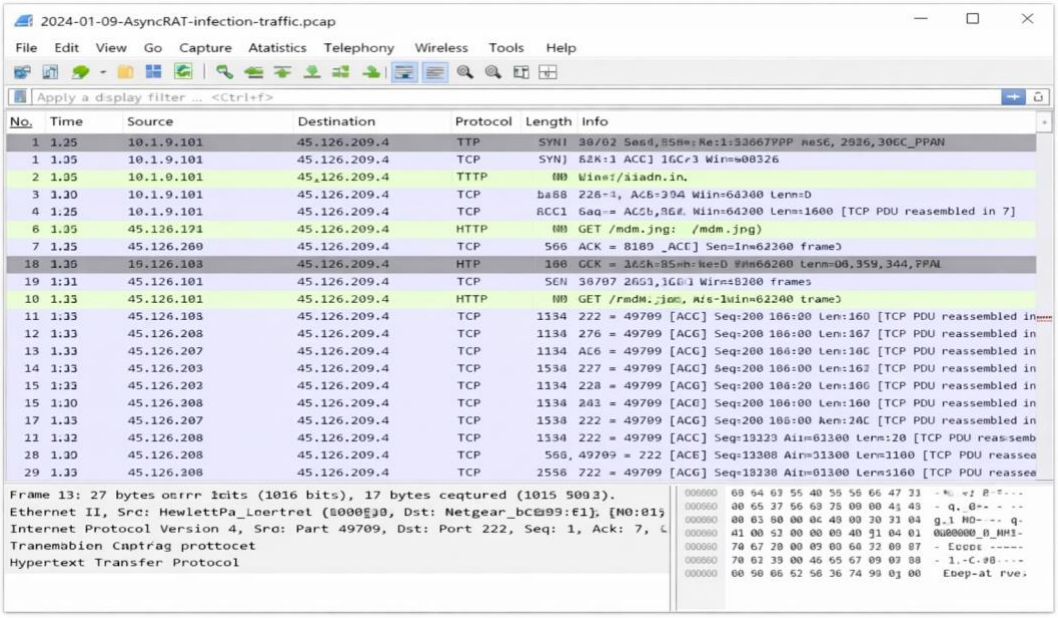

Figure 3. PCAP file view in Wireshark.

Figure 3 shows that Wireshark enables chronological tracing of sessions and packet streams to isolate suspicious communications and link traffic evidence to observed artifacts. The analysis prioritizes communication between the infected host and external infrastructure, with particular attention to traffic directed toward the suspicious endpoint 45.126.209.4:222. Chronological inspection of these interactions supports reconstruction from initial artifact downloads to persistent external connectivity and C2-oriented behavior, as shown in Figure 4.

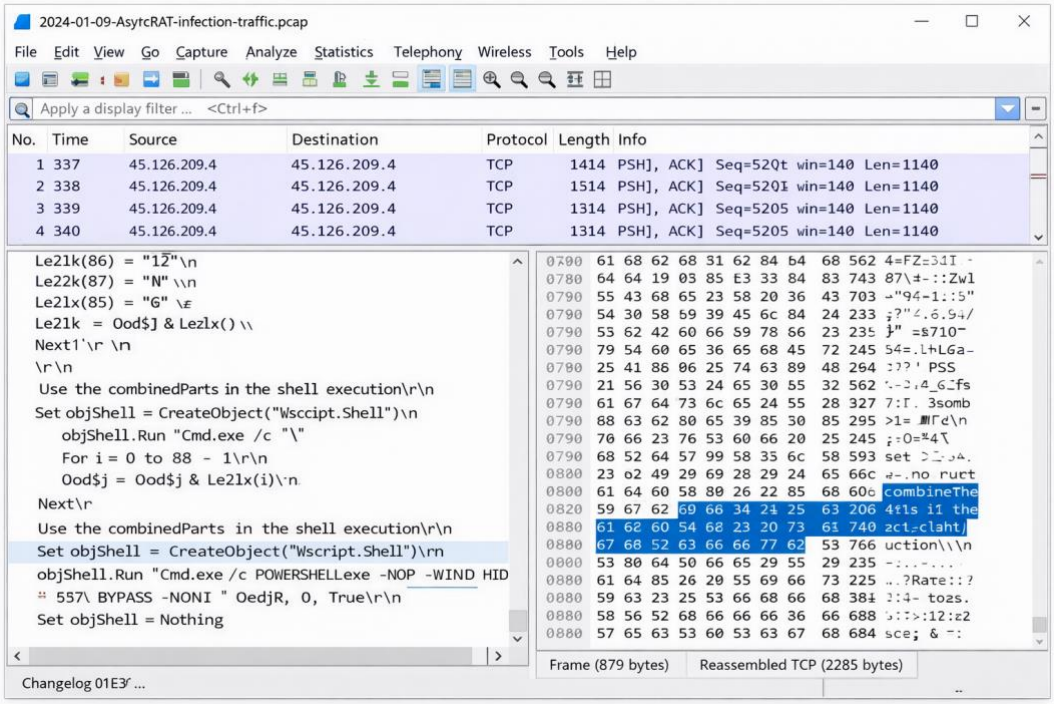

Figure 4. Reconstruction of communication to understand the malware infection process.

Figure 4 shows that traffic to 45.126.209.4:222 functions as a critical indicator of AsyncRAT activity and provides evidence of how the malware maintains contact with external infrastructure. The reconstructed flow links early-stage artifact retrieval to subsequent persistent communication, indicating sustained remote control capability. The investigation then documents IoCs extracted from the traffic and artifacts, including the suspicious endpoint 45.126.209.4:222, execution-related files such as xlm.txt and mdm.jpg, and supporting identifiers such as file hashes and domains where available.

4.1. Date and Source of Infection

The investigation analyzes PCAP traffic and supporting malware artifacts from an incident dated Tuesday, 9 January 2024. This subsection establishes the initial delivery source and the first trigger component recorded in the evidence, as summarized in Table 1.

Table 1. Event context and initial delivery source.

Elemen	Nilai
Incident date	Tuesday, 9 January 2024
Initial delivery container	ISO file of unknown origin
Trigger component	Windows Script File (WSF) inside the ISO
Function	Initiates the execution chain leading to AsyncRAT deployment

Table 1 shows that the incident is anchored to a specific date, Tuesday, 9 January 2024, which provides a fixed temporal reference for reporting the case. Table 1 also shows that the initial delivery mechanism is an ISO container

of unknown origin, meaning the evidence confirms the container type but does not provide a verified provenance for the source. In addition, Table 1 identifies a Windows Script File (WSF) inside the ISO as the trigger component. The "Function" entry in Table 1 explicitly defines the role of this trigger: it initiates the execution chain leading to AsyncRAT deployment. Together, the table entries establish the minimum evidence-backed starting point of the incident: when the case occurred, what container delivered the initial content, what component triggered execution, and what that trigger accomplishes.

4.2. Chronology of Infection

The reconstructed infection timeline begins when the victim system accesses the ISO container and executes the embedded WSF trigger. The trigger retrieves external artifacts, stages execution components on the host, and then establishes persistent remote communication, as summarized in Table 2.

Table 2. Reconstructed chronology of the AsyncRAT infection.

Step	Elemen	Nilai
1	ISO file accessed	Tuesday, 9 January 2024
2	WSF trigger executed	ISO file of unknown origin
3	External artifacts downloaded	Windows Script File (WSF) inside the ISO
4	Host artifacts created	Initiates the execution chain leading to AsyncRAT deployment
5	Persistent connectivity established	Communication with madmrx.duckdns.org:8808 over HTTPS/TLSv1.0

Table 2 shows a sequential chain that links the initial container and script trigger to observable network retrieval and host-side staging. Table 2 also shows that the activity transitions from artifact delivery to persistent external communication, which supports reconstruction of the infection flow using consistent evidence across files, URLs, and network endpoints.

4.3. Stages of AsyncRAT Infection

4.3.1 Initial Infection

The initial stage starts when the victim system accesses the ISO container invoice#5487214847577.iso and executes the embedded WSF trigger invoice#5487214847577.wsf. These two entry-point artifacts provide deterministic identifiers—file names, sizes, and SHA-256 hashes—that support unambiguous correlation between the delivery container and the first executable trigger, as reported in Table 3.

Table 3. Initial infection artifacts and cryptographic identifiers.

Artifact	Size	SHA-256
invoice#5487214847577.iso	129,024 bytes	be78b500f71db3b870a6ab00f26fd1dcb54bc19a218c93698d6146a87b488ed5
invoice#5487214847577.wsf	65,593 bytes	39ce0b953f3831429fa1c971ad0da741877ad2c932406e43f64874e65f82a238

Table 3 shows that the infection entry point is fully specified by two artifacts with fixed sizes and SHA-256 hashes. The ISO file uniquely defines the initial delivery container, while the WSF file uniquely defines the trigger responsible for initiating the execution chain. These identifiers enable consistent verification of the same entry artifacts across repeated examinations and facilitate reliable evidence correlation in subsequent stages.

4.3.2 WSF File Execution

Executing the WSF trigger transitions the activity from a local execution event into a network-mediated delivery stage, where the host establishes a session to 45.126.209.4:222 and automatically retrieves two external artifacts, as shown in Figure 5.

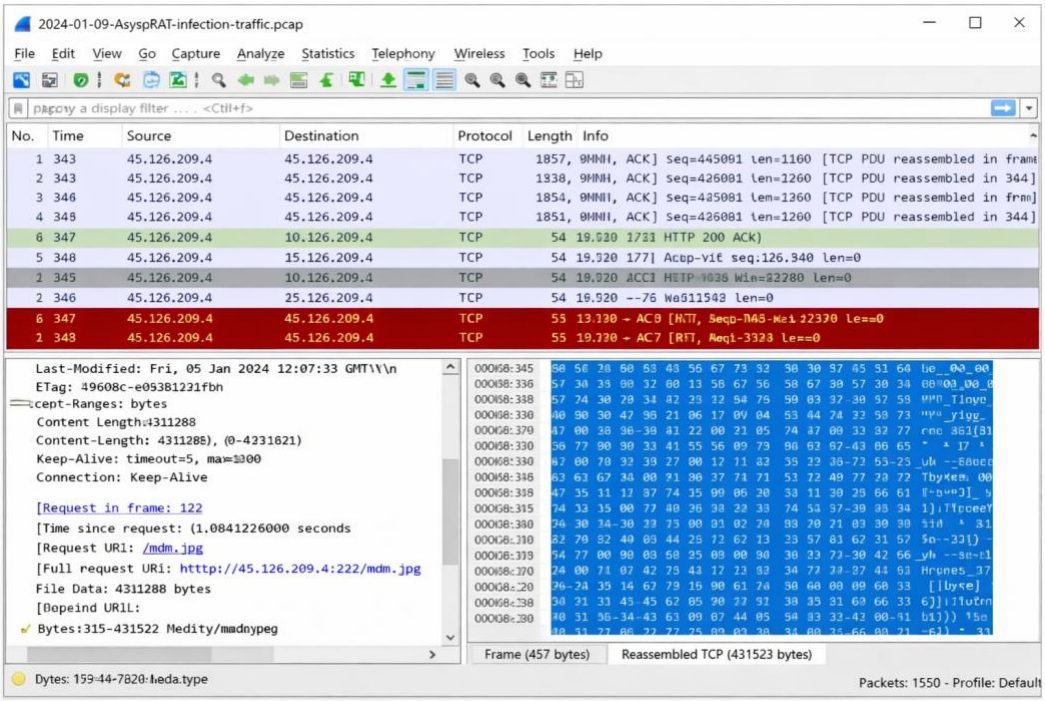

Figure 5. WSF file execution process.

Figure 5 provides protocol-level evidence that the WSF execution is directly coupled with artifact acquisition from an external endpoint. The traffic trace captures the retrieval as explicit HTTP transactions (e.g., a request URI such as /mdm.jpg) followed by the corresponding server response and payload transfer, which can be inspected through reassembled TCP segments. Importantly, the exchange exposes stable retrieval metadata—such as the endpoint, requested resource path, and response headers (e.g., content descriptors and length)—that can be used to anchor the downloaded artifacts to a specific network event. These file-level identifiers support deterministic verification and repeatable validation of the delivery behavior, as summarized in Table 4.

Table 4. Artifacts downloaded during WSF execution.

File	Size	SHA-256	URL	Note
xlm.txt	1,974 bytes	1e9c29d7af6011ca9d5609cb93b55496 5c61105a42df9fe0c36274e60db71b1d	http://45.126.209.4:222/xlm.txt	Staging component
mdm.jpg	431,208 bytes	83babee77db36512c0eab8ea6b35e981 aa4288a4095985d69b3841f8b684fe11	http://45.126.209.4:222/mdm.jpg	Disguised payload carrier

Table 4 shows that the delivery endpoint hosts two distinct artifacts with complementary roles: a small text-based component (xlm.txt, 1,974 bytes) labeled as a staging component, and a substantially larger file presented as an image (mdm.jpg, 431,208 bytes) noted as a disguised payload carrier. Both downloads originate from the same server and are uniquely anchored by their SHA-256 digests and URLs, enabling reproducible identification across repeated runs. The use of an image extension for the larger artifact is consistent with file-type masquerading, which can reduce casual visual suspicion while still supporting the continuation of the execution chain.

4.3.3 Infections in the Windows Host System

After the download phase completes, the infection transitions from remote delivery to host-side staging by materializing additional artifacts under the Windows Public directory (C:\Users\Public). These host-resident files establish the local execution path and persistence mechanism, as illustrated in Figure 6.

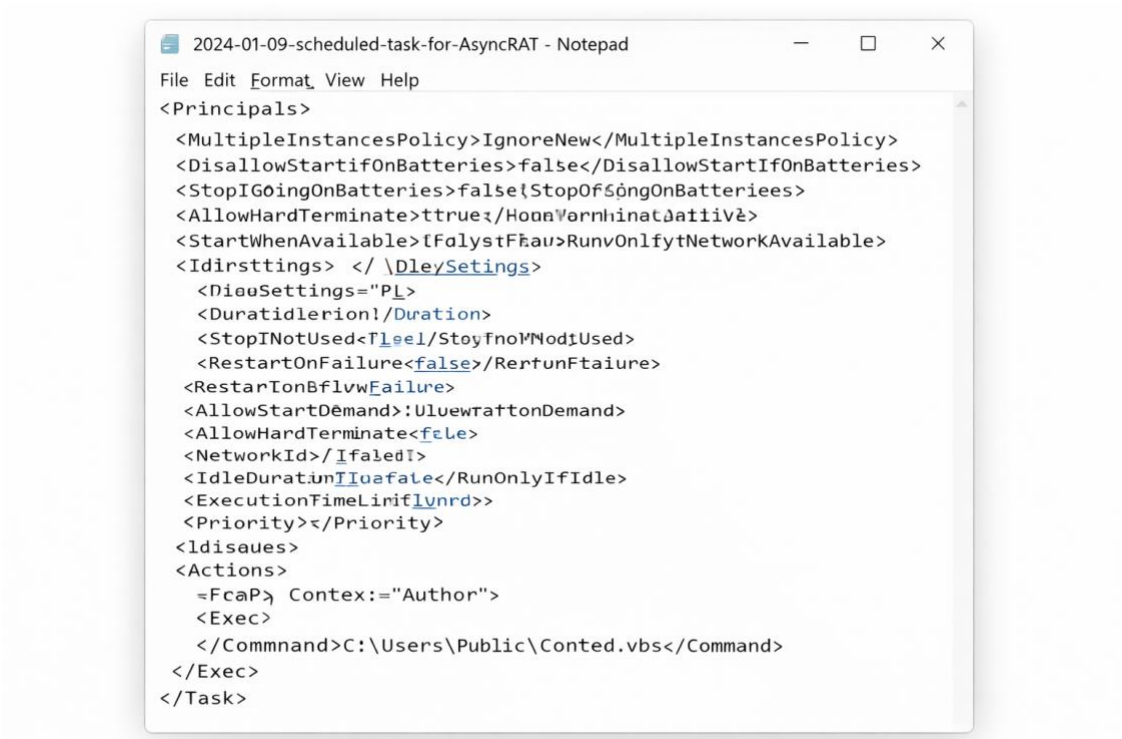

Figure 6. Malicious files from an external server that are downloaded when running WSF.

Figure 6 indicates that, once the external retrieval finishes, the workflow is continued on the Windows host through script-based components rather than through further direct downloads. The captured configuration shows an execution reference to a script placed in the Public directory (e.g., C:\Users\Public\Conted.vbs), demonstrating how the delivered content is converted into runnable host artifacts and tied to a repeatable launch mechanism. Together, this evidence supports the interpretation that the infection pipeline progresses into a local staging phase where execution is orchestrated via multiple script types, with the corresponding artifact set summarized in Table 5.

Table 5. Host artifacts created during staging and persistence.

File	Location	Size	SHA-256	Description
Conted.bat	C:\Users\Public\Conted.bat	205 bytes	cba344447d8228d88c93d64ffdcda1de8562ef41adc4901191548e00bbfc5f19	Batch staging file
Conted.ps1	C:\Users\Public\Conted.ps1	429,283 bytes	3a0a477030eaba84883193ede461d8595c3ca4345811632e295d9c2d136c1593	PowerShell payload derived from mdm.jpg
Conted.vbs	C:\Users\Public\Conted.vbs	688 bytes	a31dbd6f7416f150403c19be69f02d5e8608f5e7fae88a29831d40db15849b60	VBScript launcher

Table 5 shows that the staging phase relies on a multi-script toolchain located under a common Public path, with each artifact uniquely identifiable by its SHA-256 hash. The chain includes a compact batch staging file

(Conted.bat, 205 bytes), a PowerShell script (Conted.ps1, 429,283 bytes) explicitly noted as being derived from the previously retrieved mdm.jpg, and a VBScript launcher (Conted.vbs, 688 bytes) used to invoke the next step in the execution path. The use of batch, PowerShell, and VBScript in combination provides a structured and flexible execution sequence using built-in Windows scripting interfaces, while the shared directory placement and hashes enable deterministic verification of file presence and integrity across repeated analyses.

4.3.4 Network Communication

Traffic evidence indicates that the observed communication involves two external infrastructure roles: an initial payload delivery endpoint and a separate persistent C2 endpoint, with operator-facing connectivity maintained over HTTPS using TLSv1.0, as summarized in Table 6.

Table 6. External infrastructure roles observed in network communication.

Endpoint	Role	Evidence
45.126.209.4:222	Payload delivery	Serves xlm.txt and mdm.jpg
madmrx.duckdns.org:8808	Persistent C2	Maintains operator communication over HTTPS/TLSv1.0

Table 6 shows a clear functional separation between early-stage delivery and ongoing remote control. The delivery server (45.126.209.4:222) is directly associated with distributing the staged artifacts (xlm.txt and mdm.jpg), whereas the domain madmrx.duckdns.org:8808 is characterized as a persistent C2 endpoint that sustains operator communication over HTTPS/TLSv1.0. This division reduces dependence on a single endpoint for the full infection lifecycle and supports sustained access by decoupling artifact hosting from long-lived command-and-control.

4.4 Indicators of Compromise (IoC)

The investigation extracts high-fidelity IoCs that bind the delivery artifacts, infrastructure endpoints, and host-side scripts into a single traceable chain, confirming AsyncRAT activity across delivery, execution, and persistence stages as summarized in Table 7.

Table 7. Description of elements and values of analysis results.

Element	Value
File ISO	invoice#5487214847577.iso
Hash ISO	be78b500f71db3b870a6ab00f26fd1dcb54bc19a218c93698d6146a87b488ed5
IP Address	45.126.209.4:222
C2 Server	madmrx.duckdns.org:8808 (HTTPS over TLSv1.0)
Execution File	Conted.bat, Conted.ps1, Conted.vbs
Download File	xlm.txt, mdm.jpg
Attack Techniques	File masking, abuse of scripting (WSF, PowerShell, VBScript), TLSv1.0

Table 7 shows that the compromise can be traced from the initial containerized lure to sustained remote access using a consistent set of identifiers. At the delivery stage, the ISO lure (invoice#5487214847577.iso) is anchored by its SHA-256 hash and is linked to the delivery endpoint (45.126.209.4:222) that hosts the retrieved artifacts (xlm.txt and mdm.jpg). Following delivery, the infection transitions into host-side staging through a script set placed under the Public directory (Conted.bat, Conted.ps1, and Conted.vbs), each uniquely verifiable via SHA-256. Finally, persistence and operator control are represented by the C2 endpoint (madmrx.duckdns.org:8808) communicating over HTTPS/TLSv1.0. Collectively, these IoCs provide deterministic matching for incident

response and threat hunting by enabling direct correlation between observed network transactions, created host artifacts, and the external infrastructure used for ongoing control.

4.5 Impact

AsyncRAT enables full remote control of infected systems, including arbitrary command execution, additional payload delivery, file manipulation, and continuous activity monitoring. Persistent C2 connectivity enables covert access and increases the risk of long-term surveillance and follow-on compromise. Effective mitigation therefore requires blocking delivery and C2 endpoints, validating IoCs across telemetry sources, and enforcing proactive controls to prevent escalation and sustained adversary control.

5. DISCUSSION

This study provides an in-depth examination of the infection process of the AsyncRAT malware through a network forensic approach using Wireshark. The analysis focuses on PCAP files containing traces of network traffic to identify malicious activities, particularly communication with Command and Control (C2) servers. AsyncRAT, a Remote Access Trojan, enables covert remote control of victim systems. By analyzing datasets from Malware Traffic Analysis, this research demonstrates how infections can be forensically traced from seemingly normal traffic that conceals hidden malware interactions.

Wireshark was employed as the primary tool for packet analysis due to its ability to capture, read, and interpret network data. Functions such as filtering, following TCP streams, and protocol identification were crucial in uncovering the stages of infection. As emphasized by Sikos (2020), packet analysis is a fundamental technique in network forensics, serving as a primary method for detecting breaches, malware infections, and intrusions. Accordingly, this study positions packet analysis as the central approach for tracing AsyncRAT communication pathways.

The significance of Wireshark in PCAP analysis is further highlighted by Barik et al. (2021), who note its widespread use in network forensics due to its intuitive interface, flexible filters, and detailed visualization capabilities. In practice, these features enabled the identification of initial communication triggered by a .wsf file, which downloaded xlm.txt and mdm.jpg. These files subsequently injected the AsyncRAT payload into the victim system. The activity was later marked by communication with the dynamic domain madmrx.duckdns.org on port 8808 via TLSv1.0, indicating C2 interaction.

A major challenge in modern malware traffic analysis is the use of encryption to obfuscate malicious activity. In this case, AsyncRAT employed TLSv1.0, which hindered direct payload inspection. This limitation underscores the constraints of traditional methods that rely solely on header and metadata information. To address such challenges, Zhang et al. (2023) proposed TB-Graph, a machine learning approach leveraging relational graph attention networks to classify malware traffic even under encryption. Although not directly applied in this study, TB-Graph represents a promising avenue for extending manual Wireshark-based analysis.

The novelty of this research lies in integrating manual Wireshark analysis with infection sequence reconstruction based on artifacts and timestamps within PCAP files. This approach not only detects suspicious communication but also reveals the chronological attack flow: beginning with the execution of the .iso file, the invocation of the .wsf script, the download of malicious files from IP 45.126.209.4, and the creation of subsequent scripts such as conted.ps1 and conted.vbs in the victim's public directory. By tracing these artifacts, the study reconstructs how AsyncRAT infiltrates and operates as a remote controller. This principle aligns with Hilgert et al. (2024), who demonstrated that network traffic can be used not only to detect threats but also to reconstruct user activity, including file system manipulation via SMB protocol analysis.

Validation of Indicators of Compromise (IoCs) formed an integral part of the forensic process. IoCs such as IP addresses, domains, SHA256 file hashes, and script structures were verified against external references including VirusTotal and public threat intelligence databases. This validation ensured accuracy and provided contextual linkage between the AsyncRAT infection and previously documented malicious activities. Thus, PCAP-based analysis proved not only reactive but also proactive in supporting threat intelligence and security decision-making.

Methodologically, this study contributes by documenting replicable procedures for other researchers. Steps such as Wireshark configuration, communication filtering, payload analysis, artifact identification, and infection chronology reconstruction were systematically outlined and presented in a workflow diagram. This positions the research not merely as a technical case study but also as a methodological reference for future investigations of malware based on network communication.

In conclusion, the forensic network analysis approach using Wireshark proved effective in uncovering communication patterns and infection traces of AsyncRAT. Despite challenges posed by encrypted traffic and limitations in payload inspection, the combination of manual analysis, IoC validation, and systematic PCAP examination yielded a comprehensive understanding of the attack stages. These findings enrich digital forensic practices and open opportunities for hybrid approaches combining manual and automated methods for malware detection and reconstruction.

6. CONCLUSION

AsyncRAT infections can be effectively identified through network traffic analysis using Wireshark. The reconstructed attack sequence revealed the use of an ISO file containing a WSF script that downloaded malicious artifacts before executing shell scripts and establishing encrypted communication with the dynamic domain madmrx.duckdns.org via TLSv1.0. Wireshark successfully uncovered domains, IP addresses, and associated files, while the validation of Indicators of Compromise (IoCs) ensured accurate identification and contextual linkage with documented malicious activities. This study contributes a replicable methodology for forensic analysis, combining manual packet inspection, artifact tracing, and IoC validation. Future development should emphasize automation of IoC extraction, integration of machine learning for encrypted traffic analysis, and application in enterprise-scale environments. Expanding this approach to other malware families employing similar obfuscation techniques will further strengthen classification patterns and enhance proactive threat detection.

Reference

Afifah Rodhiyatun Nisa, A., Wijayanto, A. D., Priana, A. P. J., & Setiawan, A. (2024). Analisis log server untuk mendeteksi serangan DDoS pada keamanan jaringan di website. Journal of Internet and Software Engineering, 1(3), 17. https://doi.org/10.47134/pjise.v1i3.2612

Beale, J., Orebaugh, A., & Ramirez, G. (2006). Wireshark & Ethereal toolkit. Elsevier.

Barik, K., Das, S., Konar, K., Banik, B. C., & Banerjee, A. (2021). Exploring user requirements of network forensic tools. Global Transitions Proceedings, 2, 350–354. https://doi.org/10.1016/j.gltp.2021.08.043

Fairuz, M., et al. (2021). Cyber exercise penanganan malware. Jurnal Info Kripto.

Hilgert, J.-N., Mahr, A., & Lambertz, M. (2024). Mount SMB.pcap: Reconstructing file systems and file operations from network traffic. Forensic Science International: Digital Investigation, 50, 301807. https://doi.org/10.1016/j.fsidi.2024.301807

Ikhwanul Uzlah, L., & Saputra, R. A. (2024). Deteksi serangan siber pada jaringan komputer menggunakan metode random forest. Jurnal Mahasiswa Teknik Informatika, 8(3).

Iqbal, H., & Naaz, S. (2019). Wireshark as a tool for detection of various LAN attacks. International Journal of Computer Sciences and Engineering, 7(5), 833–837. https://doi.org/10.26438/ijcse/v7i5.833837

Luber, T., & Schiller, J. (2020). Detecting C&C traffic in encrypted channels using machine learning. Journal of Network and Computer Applications, 158, 102589.

Malware-traffic-analysis.net. . (2024). AsyncRAT infection dataset.

Fairuz, M., Yusuf, G., & Setiadji, B. (2021). Pembuatan bahan cyber exercise sebagai sarana latihan penanganan insiden malware (Studi kasus: Instansi XYZ). Jurnal Info Kripto, 15(3), 123–131.

Nugroho, H. A., & Prayudi, Y. (2015). Penggunaan teknik reverse engineering pada malware analysis untuk identifikasi serangan malware. Konferensi Nasional Sistem Informasi (KNSI 2014).

The Hacker News. (n.d.). Retrieved from www.thehackernews.com

Setia, P. T., Widiyasono, N., & Putra Aldya, A. (2018). Analysis malware Flawed Ammyy RAT dengan metode reverse engineering. Jurnal Informatika: Jurnal Pengembangan IT, 3(3), 371–379. https://doi.org/10.30591/jpit.v3i3.1019

Parulian, S., Pratiwi, D. A., & Yustina, M. C. (2021). Ancaman dan solusi serangan siber di Indonesia. TELNECT Journal. http://ejournal.upi.edu/index.php/TELNECT/

Rabbani, S., & Diana, D. (2023). Prediksi kategori serangan siber dengan algoritma klasifikasi random forest menggunakan Rapidminer. SMATIKA Jurnal, 13(2), 284–293. https://doi.org/10.32664/smatika.v13i02.934

Ramdan, A., Widyasono, N., & Mubarok, H. (2022). Prediksi jaringan TOR dan VPN menggunakan algoritma K-Nearest Neighbour pada trafik darknet. Jurnal Sistem Cerdas.

Sandriana, A., & Maulana, F. (2022). Klasifikasi serangan malware terhadap lalu lintas jaringan Internet of Things menggunakan algoritma K-Nearest Neighbour (K-NN). E-JOINT (Electronica and Electrical Journal of Innovation Technology), 3(1), 12.

Sutarti, Siswanto, & Bachtiar, A. (2023). Analisis web phishing men ggunakan metode network forensic dan block access situs dengan router Mikrotik. Jurnal PROSISKO, 10(1), 71–83.

Sutra, S. M. S., & Haryanto, A. (2023). Upaya peningkatan keamanan siber Indonesia oleh Badan Siber dan Sandi Negara (BSSN) tahun 2017–2020. Global Political Studies Journal, 7(1), 56–59.

Sikos, L. F. (2020). Packet analysis for network forensics: A comprehensive survey. Forensic Science International: Digital Investigation, 32, 200892. https://doi.org/10.1016/j.fsidi.2019.200892

Singh, A., & Singh, M. (2017). Malware analysis: A survey. Journal of Network and Computer Applications, 78, 1–18.

Wireshark.org. . (2024). Wireshark network protocol analyzer. Retrieved from https://www.wireshark.org

Zhang, Y., Xie, T., Zhang, Y., Liu, H., Zhao, Y., Liu, Y., … & Xie, D. (2023). TB-Graph: Enhancing encrypted malicious traffic classification through relational graph attention networks. Computers & Security, 132, 103376. https://doi.org/10.1016/j.cose.2023.103376

Zscaler ThreatLabz. (2021). The rise of AsyncRAT: A deep dive into its tactics and infrastructure. Retrieved from https://www.zscaler.com