A Lightweight Rule-Based Detection Approach for ARP Flooding Malware in Office Networks

Rizal Fathoni Aji, Heri Kurniawan, Nilamsari Putri Utami
Faculty of Computer Science, Universitas Indonesia, Indonesia

***Abstract*—Address Resolution Protocol (ARP) is a standard protocol used to map an IP address to its MAC address so the network can send packets to its destination. Office networks, which typically have limited network resources, are vulnerable to ARP flooding attacks launched by malware. ARP flooding can be used by malware to create network disruption and jam the networks. This study presents a rule-based detection method, Time Density ARP Thresholding with Binding Consistency Monitoring (TDCM), to identify ARP flooding using a simple mechanism, making it suitable for use in networks with limited hardware. To detect flooding anomalies, the TDCM algorithm monitors the flow of ARP packets and the consistency between MAC IP bindings in ARP packets. In this study, a series of experiments was conducted and repeated multiple times. On average, the experiment shows that the system performs well under high-volume ARP attack conditions. This proposed method offers an alternative to machine learning techniques, making it more suitable for deployment in resource-constrained office networks. Future work will focus on improving detection in low-volume attack scenarios, validating performance in real-world environments, and implementing on devices with limited computing resources.**

***Keywords*—ARP flooding; cybersecurity detection; rule-based detection; lightweight intrusion detection**

I. Introduction

Today, office networks play a crucial role in our daily activities. High usage of office networks and internet connectivity introduces cybersecurity threats, particularly from malware. Malware can exploit vulnerabilities in protocols to disrupt network operations [1]. For example, the Address Resolution Protocol (ARP) flooding technique can be used by malware to cause network and device congestion. ARP flooding is a network-based attack that exploits the mechanism in the ARP protocol. It has a function that maps IP addresses to MAC addresses at the data link layer [2]. In an ARP flooding attack, an attacker sends a large number of malicious ARP packets into the network, causing ARP Table overflow on network devices. This results in incorrect or missing address resolutions and disruption of network services [2].

A high volume of ARP flooding attacks can cause network denial-of-service (DoS) attacks, data loss, and potential breaches of sensitive information [3]. ARP flooding attacks generate a high volume of ARP request/reply packets from malicious sources within a short time frame [4]. These traffic spikes sometimes exceed the number of ARP activities observed in typical LAN environments, which typically maintain stable ARP caches and low ARP exchange rates. Attackers often used spoofed or randomized MAC and IP addresses, generating inconsistent ARP Table updates that lead to Table exhaustion or overflow [5]. Sometimes, forged ARP packets are broadcast across the subnet to maximize the effect [6]. Those characteristics can be used as a foundation for constructing rule-based detection mechanisms [7].

Some approaches have been developed to detect and mitigate ARP flooding attacks, from traditional fingerprint-based techniques to machine learning and deep learning methods. Conventional methods rely on static rules, such as port security, static ARP entries, dynamic ARP inspection, and a whitelist based on IP MAC binding validation [2]. Other methods, such as monitoring ARP request rates, tracking consistency between MAC and IP addresses, and identifying bulk packet broadcasts from non-gateway devices, could help administrators quickly detect ARP flooding attempts. However, such approaches may lack adaptability to evolving threats.

Currently, ARP attack detection is shifting toward more intelligent and adaptive mechanisms. Using machine learning and deep learning models, such as Random Forest, Convolutional Neural Networks, and Bi-directional Long Short-Term Memory (Bi-LSTM) architectures, achieves high accuracy and low false-positive rates in identifying malicious ARP traffic [5], [7], [8]. In industrial contexts, especially in the Internet of Things environments, hybrid CNN and Bi-LSTM models can detect large-scale ARP-based attacks [7]. The adoption of Software-Defined Networking (SDN) can also enhance ARP security by enabling centralized binding verification and automated mitigation strategies [6]. Another solution employs blockchain-based immutable address-resolution records and cloud-based ARP monitoring systems, thereby enhancing scalability and resilience for ARP detection in distributed environments [5].

Although machine learning and deep learning show promising results, they have limitations. They require large datasets for training, which are challenging to obtain in real-world ARP flooding scenarios. Machine learning and deep learning also make detection more complex for administrators to interpret [9]. These models consume substantial computing resources during training and inference, making deployment less practical [6]. Modified packets may also bypass those models by exploiting a threshold in the machine-learning algorithm. In contrast, fingerprint-based mechanisms can offer deterministic, explainable, and lightweight detection that can be used with minimal resources [4].

This study proposes an algorithm, Time Density ARP Thresholding with Binding Consistency Monitoring (TDCM), for detecting ARP flooding attacks using a rule-based detection

approach. With low computational requirements, TDCM can be deployed on resource-constrained devices. The algorithm detects common ARP flooding characteristics, such as bulk ARP packet broadcasts within short intervals and spoofing signs, in which a single IP address is associated with multiple MAC addresses. This mechanism is evaluated through simulations involving both regular network traffic and ARP attack scenarios.

II. Theoretical Foundation and Related Works

An office network designed to connect multiple devices within a work environment, enabling communication, data sharing, and resource sharing among employees and departments. The network infrastructure typically comprises both wired and wireless local area networks. Wired connections often rely on Ethernet cables connected to switches and routers, providing stable, high-speed connectivity for desktop computers, servers, and network printers. Wireless networks, by contrast, offer flexibility and mobility, enabling laptops, tablets, and mobile devices to connect without physical cabling. A central switch or a set of interconnected switches serves as a backbone that manages traffic flows between devices. These switches are often connected to one or more routers that provide internet access and enable communication with wide-area networks. Depending on the organization's size and structure, the network may be divided into segments or virtual LANs to separate traffic for security and performance reasons [10].

Another important aspect of office networking is sharing resources. File servers are commonly used to store organizational data and to enable employees to share files and documents. Network printers and scanners are connected through print servers, enabling centralized management and reducing the need for individual device setups. In larger organizations, directory services are often implemented to manage users, devices, and policies in a unified manner. This centralized approach simplifies administration while maintaining a consistent security framework.

Network administrators must ensure that bandwidth is allocated appropriately to prevent congestion, especially during peak hours when many users are online simultaneously. Quality-of-service configurations may be applied to prioritize critical traffic, such as voice over IP calls or video conferencing, over less time-sensitive data transfers. Regular network assessments help identify bottlenecks and potential failures before they affect productivity. Finally, office networks are designed for scalability. As businesses grow, additional devices, users, and services need to be integrated without causing significant disruptions.

The Address Resolution Protocol (ARP) is a fundamental component of computer networks. It operates at the data link layer and plays an essential role in maintaining network connectivity. Its primary function is to map IP addresses to corresponding MAC addresses. Each device on the network maintains an ARP Table, which contains a list of IP-to-MAC address mappings for other devices. When a computer application wants to send data, it uses a system call in the operating system. The operating system first checks its ARP Table to see whether the destination IP address already has a corresponding MAC address. If it does not, the operating system broadcasts an ARP request on the network, asking which MAC address corresponds to that IP address. Upon receiving the ARP request, the corresponding device responds with its MAC address. However, if the IP address is in a different subnet, the router responds instead, using its own MAC address. Upon receipt of the reply, the ARP Table is updated with the new information. Over time, if an entry is no longer valid or the time to live expires, the IP-to-MAC pair is removed from the Table [10].

The ARP packet flows through the network without any encryption or authentication mechanism [11]. Any device on the network can generate a bogus or fake ARP packet, which can lead to network errors and disruption. This lack of protection means that any device connected to the network can generate a fake or bogus ARP packet. When this happens, it can result in serious problems such as network errors or even the complete disruption of communication between devices [3]. This weakness has long been recognized as a major security vulnerability, and attackers often take advantage of it to intentionally cause malfunctions and interruptions in computer networks. However, because ARP is a widely used protocol and an integral part of the network layer, the vulnerability still exists. Hong et al. [11] and other researchers have already proposed ideas to enhance security in the ARP protocol, but changing the current ARP protocol requires significant effort, especially changing the code embedded in millions of network devices worldwide.

ARP flooding is a type of network attack that exploits a vulnerability in the ARP protocol. In this attack, the attacker generates a large number of ARP packets and sends them across the network [12]. The continuous flow of these fake packets can overwhelm the ARP tables in the targeted devices, filling them with unnecessary or false ARP entries. When the ARP table becomes overloaded, the device may no longer function properly, leading to errors in communication between hosts on the network. Another major effect of ARP flooding is the excessive traffic it creates on the network. Because so many packets are being transmitted at once, the network bandwidth becomes congested. As a result, legitimate traffic, such as actual data transmissions between users, may be delayed or even completely blocked. This may lead to denial of service or upper-layer service disruption [5]. ARP flooding is not only a direct attack on specific devices but also a threat to the stability and availability of the entire network.

Certain types of malware exploit weaknesses in the ARP protocol by using ARP flooding to launch denial-of-service (DoS) attacks or to position themselves for man-in-the-middle (MITM) attacks [3]. One common method involves sending a large number of ARP request packets into the network, which forces devices with the corresponding IP addresses to repeatedly respond. In some cases, the malware may also send a large volume of ARP reply packets, which can overwhelm the targeted devices. This continuous flooding puts heavy pressure on the ARP cache and consumes a significant amount of CPU resources, eventually disrupting normal network operations and causing a denial of service. If the attack is directed at critical infrastructure such as routers or switches, the impact can be even more severe. It could shut down the entire network and interrupt communication for all connected devices.

ARP flooding attacks typically exhibit distinct traffic patterns that can be detected using rule-based mechanisms. One primary characteristic is a high volume of ARP request or reply packets, often originating from a single host or multiple forged sources within a short time frame [12]. This abnormal packet volume is usually much higher than the expected ARP traffic in a typical local area network, which typically maintains stable ARP caches and exhibits low ARP exchange rates [4]. Additionally, ARP flooding often involves spoofed or randomized source MAC and IP addresses, resulting in continuous updates to the ARP tables of switches or hosts, eventually leading to table exhaustion [5]. Another notable feature is the broadcast nature of forged ARP packets, as they are typically sent to the entire subnet to maximize network disruption [6]. These traits, such as packet rate threshold violations, inconsistent or unknown MAC and IP bindings, and repetitive broadcast ARP replies, form the basis for rule-based detection mechanisms. By implementing static thresholds, whitelist-based validation of IP and MAC pairs, and monitoring for excessive ARP broadcasts from non-gateway devices, network administrators can define deterministic rules to promptly identify and block ARP flooding attempts without relying on complex models or machine learning [7].

Alongside intelligent detection models, recent research also underscores the importance of integrating detection with prevention frameworks. For instance, deploying Dynamic ARP Inspection in SDN-controlled environments has enabled rapid identification and isolation of malicious nodes [4]. Furthermore, cloud-based ARP monitoring systems that use real-time data analysis have shown scalability and effectiveness in large, distributed networks [5]. Integrating IP and MAC static binding in industrial SDN settings and using blockchain for immutable address resolution records represent emerging frontiers in ARP security [6]. These developments reflect a broader trend toward building resilient, autonomous systems that not only detect but also actively adapt to evolving ARP-based threats across enterprise, cloud, and IoT networks.

Despite the growing adoption of machine learning and deep learning for ARP flooding detection, these approaches have notable limitations compared with traditional fingerprint-based mechanisms [9]. A major drawback is reliance on large, well-labeled datasets for effective training. In practice, collecting real-world ARP flooding data can be challenging, leading to models that may generalize poorly across different network environments [5]. Additionally, machine learning and deep learning models often act as black boxes, making it difficult for network administrators to interpret or justify detection outcomes. By contrast, fingerprint-based systems are rule-driven and inherently explainable. Furthermore, these techniques tend to consume significant computational resources, especially during training and real-time inference, which can hinder deployment in resource-constrained environments. Another concern is their vulnerability to adversarial inputs; carefully crafted packets can potentially bypass detection by manipulating model behavior. In contrast, fingerprint-based mechanisms, while limited to known attack signatures, offer fast, deterministic, and lightweight detection that is easier to implement and maintain [4]. These tradeoffs highlight the need for a balanced approach, where intelligent systems are supported by traditional methods to ensure comprehensive and resilient ARP flooding defense.

In parallel with network-level attacks, recent studies have highlighted the growing threat of side-channel attacks, in which attackers exploit indirect information leakage, such as timing behavior, power consumption, cache access patterns, or electromagnetic emissions, to infer sensitive data [13]. Side-channel attacks have been demonstrated against both classical cryptographic implementations and lightweight cryptography, which is increasingly adopted in resource-constrained environments such as IoT and embedded systems [14],[15]. While lightweight cryptographic primitives reduce computational overhead, several works report that simplifications in control flow and reduced noise may unintentionally amplify side-channel leakage. Similarly, emerging post-quantum cryptography (PQC) algorithms, although designed to resist quantum adversaries, are vulnerable to timing and cache-based side-channel attacks due to complex polynomial operations and memory-intensive structures [16][17]. Prior research emphasizes that security mechanisms designed for constrained environments must consider not only algorithmic robustness but also implementation-level leakage. These findings reinforce the relevance of lightweight, deterministic, and transparent security mechanisms—such as rule-based detection approaches—that minimize attack surfaces and reduce exploitable side-channel behaviors in real-world deployments.

III. Research Method

This research starts with a literature study, the development of a simulation, an experiment, and the creation of conclusions. First, a literature review was conducted by analyzing existing ARP flooding detection techniques, both traditional and machine learning based, comparing the pros and cons of both mechanisms. This provided the theoretical foundation for the proposed rule-based detection framework. Second, a Python-based simulation environment was developed to model ARP traffic in a controlled office network scenario. The simulator was designed to generate both normal and attack traffic. The detection module was implemented to monitor ARP packet density and MAC IP binding anomalies. The burst packets are identified with a short timestamp among packets. The traffic simulation algorithm can be seen in Algorithm 1. Third, experiments were performed by executing multiple simulations with varying attack intensities. Each scenario was repeated five times, and detection performance was measured using metrics such as the number of undetected attack packets, true positives, and false positives. Results were averaged and visualized to reveal detection trends.

Algorithm 1: Traffic Simulation

```
SimulateTraffic():
now ← current time
attack_count = 100, 150, 200, …, 1000
normal_count = attack_count/2
For each in 1 to normal_count:
Create packet with:
src_ip = fixed unique IP
```

```
src_mac = fixed unique MAC
timestamp = now + randomFloatTime(0,5)
is_attack = False
Add to packets

For each in 1 to attack_count:
Create packet with:
src_ip = fixed IP
src_mac = unique or duplicate MAC
timestamp = now + randomFloatTime(0,5)
is_attack = True
Add to packets
```

The proposed mechanism is called Time Density ARP Thresholding with Binding Consistency Monitoring (TDCM). It combines analysis of ARP packet density with tracking of IP and MAC binding behavior, without using statistical learning models or training data. The core of the TDCM system comprises two parallel components, as shown in Fig. 1.

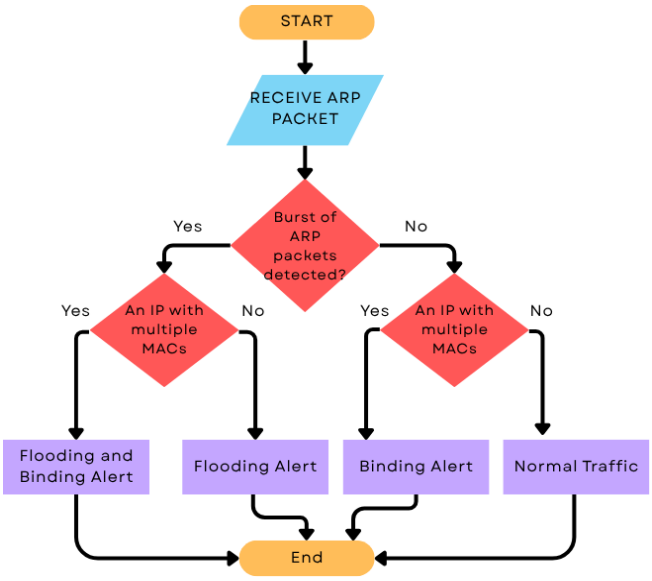

Fig. 1. Flow of the TDCM method.

The first component focuses on time-density monitoring, measuring the arrival rate of ARP packets within a predefined time window. A burst of ARP packets is detected when the number of ARP packets from a single host or targeting a particular IP address exceeds a set threshold within a short period. In this case, the system raises a potential flooding alert or a combination of flooding and binding alerts. If incoming packets contain an IP address with multiple MAC addresses, the attack is categorized as an ARP flooding and spoofing attack. Otherwise, the attack is categorized as ARP flooding only, without spoofing. This approach is based on the insight that ARP flooding attacks generate unusually high volumes of ARP traffic over a short period, which deviates significantly from typical network behavior. By calibrating this threshold based on historical network baselines, false positives can be reduced without the need for adaptive learning. Malware sometimes combines ARP flooding with a spoofed MAC address to initiate a man-in-the-middle attack.

The flooding alert component tracks the number of ARP requests and replies received per source MAC address over fixed intervals. If a particular source exceeds a predefined packet-per-interval threshold, the system flags it as a flooding candidate. Unlike static rate limiting, this approach uses sliding windows to account for short bursts while avoiding false positives from legitimate activity.

The second component, binding consistency monitoring, monitors the consistency of IP-MAC pairings over time. In a normal network, a specific IP address is usually bound to a fixed MAC address within a session or subnet. During ARP spoofing or flooding scenarios, MAC-IP bindings in ARP packets change rapidly. Multiple MAC addresses may be associated with the same IP address, or vice versa. TDCM tracks these inconsistencies and flags them as malicious packets.

Each incoming ARP packet is checked against a list of verified IP and MAC pairs. If a device continuously uses different MAC addresses with the same IP address in a short time frame, it is marked as suspicious. This check is also performed after packets are detected as burst ARP packets.

This approach offers a reliable and efficient method to detect ARP-based attacks. By using deterministic rule-based logic, it enhances interpretability and operational transparency, two aspects that often lack in machine learning approaches. Network administrators can trace each alert back to a specific condition or packet pattern, facilitating faster incident response and better threat understanding.

An optional quarantine queue can be activated when both modules flag a source: traffic from the suspicious node is rate-limited or redirected to an isolated VLAN for further inspection or logging. This multi-aspect detection, based on time density and binding logic, enables fast, deterministic identification of ARP flooding attacks using simple thresholds and lookups.

IV. Results and Discussion

To evaluate the detection effectiveness of the proposed TDCM algorithm, we conducted a series of simulations emulating ARP flooding attacks of varying intensities. The simulations generated two distinct traffic patterns. The first was legitimate ARP traffic from a single device with a fixed IP-to-MAC pairing, and the second was malicious flooding traffic that imitated a spoofing attack by sending a large volume of ARP packets from a single IP address with randomly generated MAC.

The legitimate traffic simulated a normal device transmitting ARP requests or responses at a moderate rate. Conversely, the flooding traffic was designed to trigger TDCM alerts by exceeding the time density threshold and violating binding consistency. The number of malicious ARP packets injected into the network increased from 100 to 1000 in steps of 50. This allowed testing both low-volume and high-volume attack scenarios, representative of stealthy and aggressive attack profiles.

Each configuration was executed five times with randomized MAC address generation and minor timing fluctuations during packet injection. The detection engine recorded the number of undetected packets for each run, and the results were averaged. The key metric analyzed was the

percentage of undetected malicious packets relative to the total number injected. The results in Table I represent the average number of undetected attack packets across these five runs.

TABLE I. SIMULATION RESULTS

Total packet	Flooding alert	Binding alert	False Positive (%)	False Negative (%)
150	0	91	0.00	9.00
225	0	140	0.00	6.67
300	0	191	0.00	4.50
375	0	241	0.00	3.60
450	0	291	0.00	3.00
525	0	341	0.00	2.57
600	10	391	0.00	2.25
675	34	439	5.56	2.00
750	55	491	10.00	1.80
825	80	541	13.64	1.64
900	108	591	16.67	1.50
975	136	641	19.23	1.38
1050	157	691	21.43	1.29
1125	186	741	23.33	1.20
1200	215	791	25.00	1.13
1275	234	841	26.47	1.06
1350	259	891	27.78	1.00
1425	289	941	28.95	0.95
1500	312	990	30.00	0.90

The number of packets in this simulation ranged from 150 to 1500 total packets. The total packet count consisted of one- third normal packets and two- thirds attack packets. The goal was to observe the system's ability to trigger flooding and binding alerts and measure its accuracy through false- positive and false-negative rates. At lower traffic volumes (below 600 packets), the system issued no flooding alerts, and only binding alerts were triggered. This indicates that low- volume attacks can be handled by the binding consistency module alone. However, as the total packet count increased beyond 600, the system began generating flooding alerts, starting with 34 alerts at 675 packets and increasing progressively to 312 alerts at 1500 packets.

The detection system maintained a zero false- positive rate up to 600 packets. As the packet volume increased, however, the false- positive rate gradually rose, reaching 30% at 1500 packets. This tradeoff reflects the system's sensitivity under stress, as aggressive ARP flooding patterns resemble high-volume traffic spikes that may sometimes be misclassified. On the other hand, the false- negative rate consistently decreased as the total packet count increased, from 9% at 150 packets to 0. 0.9% at 1500 packets, indicating that the system becomes more accurate at detecting attacks as flooding intensity rises.

The simulation results show the effectiveness of TDCM in detecting ARP flooding attacks. The method, which combines flooding threshold monitoring and MAC IP binding, can detect malicious ARP packet activity while maintaining a reasonable false- positive rate. These results also show that balancing packet data with traffic activity can maintain accuracy in flooding detection.

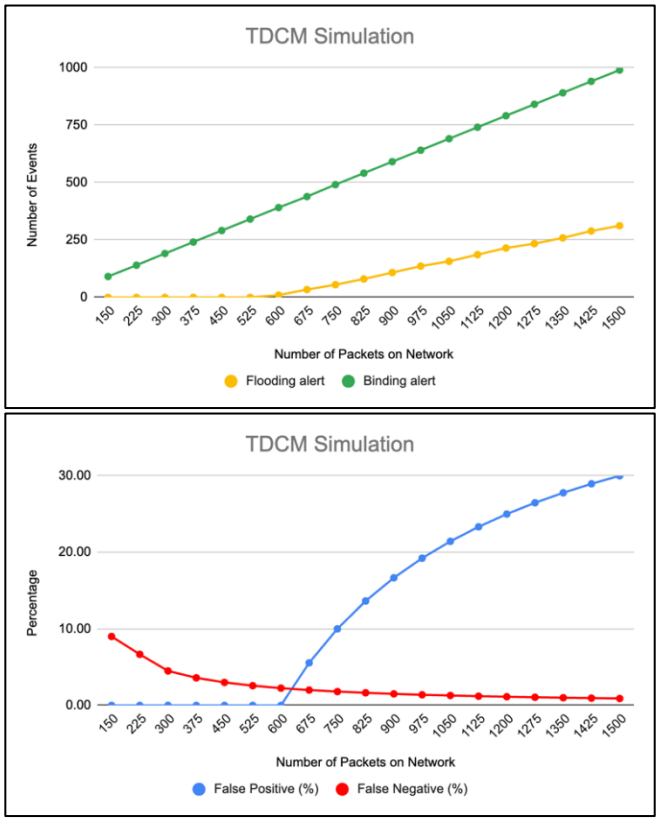

Fig. 2. Graph of simulation result.

Fig. 2 shows a trend with larger attack volume. This trend indicates that the TDCM algorithm will be more effective at recognizing ARP flooding as the number of packet attacks increases. The reduction in undetected packets suggests that the system's time density and MAC IP consistency thresholds will likely be triggered more often in high- packet- attack scenarios. This shows that TDCM may have poor performance in low-volume attacks, but it will increase its performance on large-scale ARP flooding attacks.

Compared to machine learning-based approaches, the TDCM mechanism offers a comparable alternative in high-volume ARP flooding attack scenarios. As demonstrated in the latest simulation, TDCM achieved a detection rate of up to 99% with a false-negative rate as low as 0.9% at 1500 total packets, without relying on model training or feature engineering. This indicates that TDCM can match the effectiveness of ML models in identifying large-scale flooding attacks while maintaining low computational complexity. Furthermore, while recent studies [4] acknowledge the scalability and adaptability of machine learning and deep learning techniques, they also highlight concerns about their black-box nature and limited interpretability during real-time deployment.

In contrast, TDCM's rule-based approach offers transparent detection logic, making it easier for administrators to trace and understand alerts during live network monitoring. TDCM's

approach, based on defined rules and simple logic, offers transparency and low overhead. It doesn't need a powerful computer or a GPU-powered device, making it ideal for resource-limited hardware. Because of its simplicity, TDCM can be implemented using the C programming language and deployed on low-cost IoT hardware. Technically, TDCM can also be implemented using a shell script running common networking tools on a tiny Linux system, making it feasible to deploy in resource-limited office network environments.

V. Conclusion

This study introduced a simple rule-based mechanism, Time Density ARP Thresholding with Binding Consistency Monitoring (TDCM), to detect ARP flooding malware in office networks. The proposed method combines two strategies: monitoring the number of ARP packets in the network and checking the consistency of IP-MAC bindings. TDCM does not require model training, large datasets, or high computational resources to implement.

Simulation and experiments show that the TDCM algorithm performs well under high-volume ARP flooding. MAC and IP binding detection plays a significant role in detecting most attacks. Combined with its monitoring strategies, TDCM can maintain a reasonable false-negative rate. The number of detected attack packets increased as the volume of malicious traffic increased. At a low injection level of 100 packets, the system missed an average of 10 packets. However, as the attack volume reached 1000 packets, the undetected rate dropped to less than 1%. This shows that TDCM is effective in identifying flooding attempts, making it suitable for practical deployment in infrastructure-limited office environments.

TDCM's transparency and deterministic rule structure provide an advantage in interpretability, so network administrators can readily understand, audit, and tune detection logic. This makes the system ideal for environments with limited staffing. Furthermore, its low overhead ensures compatibility with legacy systems, unmanaged switches, and embedded devices, where machine learning-based solutions are often impractical. Despite its strengths, TDCM showed reduced sensitivity to low-rate ARP flooding scenarios.

VI. Future Work

Future work will focus on enhancing detection granularity during such attacks. One promising direction is to integrate adaptive thresholding techniques that adjust detection parameters based on observed traffic baselines. Additionally, field testing in production networks is essential to assess performance under real-world conditions, including traffic noise, heterogeneous devices, and mixed protocol loads. Another possible direction for future research is to apply the algorithm in wireless environments with limited computing resources, such as IoT networks. IoT systems are an excellent setting for testing and applying TDCM because they typically comprise numerous small devices with limited processing power and memory. Experimenting with TDCM in this type of environment could yield valuable insights into how well the algorithm performs under real-world limitations. Over time, studies across different device types and network conditions could further refine the algorithm and enable more efficient, practical implementations of TDCM.

Acknowledgment

This research is supported by the Faculty of Computer Science at Universitas Indonesia, Grant no: NKB-14/UN2.F11.D/HKP.05.00/2024.

Declaration on Generative AI

The authors acknowledge using ChatGPT (OpenAI, 2025) to improve the clarity, grammar, and structure of the manuscript. The content, analysis, and conclusions remain the sole responsibility of the authors.

References

[1] T. Alsmadi and N. Alqudah, "A Survey on malware detection techniques," 2021 International Conference on Information Technology, ICIT 2021 - Proceedings, pp. 371–376, Jul. 2021, doi: 10.1109/ICIT52682.2021.9491765.

[2] D. R. Raviya, D. Satasiya, H. Kumar, and A. Agrawal, "Detection and prevention of ARP poisoning in dynamic IP configuration," 2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings, pp. 1240–1244, Jan. 2017, doi: 10.1109/RTEICT.2016.7808030.

[3] S. Oei, Y. Suyanto, and R. Pulungan, "A Comprehensive Approach for Detecting and Handling MitM-ARP Spoofing Attacks," IEEE Access, vol. 13, pp. 115503–115519, 2025, doi: 10.1109/ACCESS.2025.3585463.

[4] Y. Lu, C. Zheng, and C. Tang, "Design of ARP Attack Defense System Based on SDN Architecture in Industrial Internet," 2024. [Online]. Available: https://h-tsp.com/://creativecommons.org/licenses/by/4.0/

[5] M. Kumar and C. S. Dash, "Detecting and Preventing ARP Spoofing Attacks Using Real-Time Data Analysis and Machine Learning," International Journal of Innovative Research in Engineering & Management, vol. 12, no. 5, pp. 47–55, Sep. 2024, doi: 10.55524/IJIRCST.2024.12.5.7.

[6] Q. Li and Y. Dong, "Advanced approaches to prevent ARP attacks," Applied and Computational Engineering, vol. 44, no. 1, pp. 124–137, Mar. 2024, doi: 10.54254/2755-2721/44/20230410.

[7] J. Wang, "Industrial Internet of Things ARP Virus Attack Detection Method Based on Improved CNN BiLSTM," Journal of Cyber Security and Mobility, vol. 13, no. 5, pp. 1173–1206, Sep. 2024, doi: 10.13052/JCSM2245-1439.13516.

[8] A. Tran, X. P. L. Ngô, Q. C. Nguyễn, N. Bui Trung, and T. M. T. Dinh, "Using different machine learning models for address resolution protocol spoofing attack detection in software-defined network architecture," International Journal of Pervasive Computing and Communications, 2025, doi: 10.1108/IJPCC-03-2024-0081/1250800/USING-DIFFERENT-MACHINE-LEARNING-MODELS-FOR.

[9] 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, doi: 10.1016/J.JNCA.2015.11.016.

[10] J. F. Kurose et al., "Computer Networking A Top-Down Approach Seventh Edition," 2017, Accessed: Oct. 03, 2025. [Online]. Available: www.pearsoned.com/permissions/.

[11] S. Hong, M. Oh, and S. Lee, "Design and implementation of an efficient defense mechanism against ARP spoofing attacks using AES and RSA," Math Comput Model, vol. 58, no. 1–2, pp. 254–260, Jul. 2013, doi: 10.1016/J.MCM.2012.08.008.

[12] H. Prajapati and Z. Noorani, "A Survey on ARP Poisoning and Techniques for Detection and Prevention," International Journal of Advance Research and Innovative Ideas in Education, 2017.

[13] Y. Yarom and K. Falkner, "FLUSH+RELOAD: A high resolution, low noise, L3 cache side-channel attack," in USENIX Security Symp., 2014.

[14] M. M. Kermani, R. Azarderakhsh and Jiafeng Xie, "Error detection reliable architectures of Camellia block cipher applicable to different

variants of its substitution boxes," 2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST), Yilan, Taiwan, 2016

[15] M. M. Kermani, R. Azarderakhsh, "Efficient Fault Diagnosis Schemes for Reliable Lightweight Cryptographic ISO/IEC Standard CLEFIA Benchmarked on ASIC and FPGA," in IEEE Transactions on Industrial Electronics, vol. 60, no. 12, pp. 5925-5932, Dec. 2013

[16] A. Jalali, R. Azarderakhsh, M. M. Kermani, D. Jao. "Towards Optimized and Constant-Time CSIDH on Embedded Devices". 2019.

[17] D. J. Bernstein et al., "Post-quantum cryptography," Nature, vol. 549, no. 7671, pp. 188–194, 2017