

Netshield: A Real-Time Arp Spoofing And Ddos Detection System With Automated Mitigation And Cloud-Based Visualization

Dr. Seetha J¹, Praveen M², Sarath Kumaran P³, Ram Prashanth S⁴

¹Professor, Department of Computer Science and Business Systems Panimalar Engineering College Chennai, India

^{2,3,4}Students, Department of Computer Science and Business Systems Panimalar Engineering College Chennai, India

Abstract

ARP spoofing continues to be a significant threat in local area networks because of the lack of inherent authentication, which permits man-in-the-middle and denial-of-service attacks. Current mitigation strategies frequently depend on specific hardware or resource-heavy machine learning models, restricting their feasibility for real-time applications. This document introduces NetShield, a lightweight system for intrusion detection and response that detects ARP spoofing and volumetric DDoS attacks through rule-based IP-MAC checks and protocol-specific traffic evaluation. NetShield actively observes live traffic via libpcap, identifies MAC discrepancies within a time frame, and automatically prevents verified attackers through firewall enforcement. A logging module facilitates the transmission of structured alerts to Firebase for immediate dashboard visualization. Test results show elevated detection precision, reduced response time, and minor system resource use, rendering NetShield ideal for small-to-medium business and educational networks

Keywords : ARP Spoofing, Intrusion Detection System, Network Security, libpcap, DDoS Detection, Firewall Automation, Firebase Logging, Network Visualization.

1. Introduction

The Address Resolution Protocol (ARP) is a fundamental component of IPv4 networking that enables devices to map logical IP addresses to physical MAC addresses within a local area network. Despite its critical role, ARP was designed without authentication or integrity validation, making it inherently vulnerable to spoofing attacks. In an ARP spoofing attack, a malicious host forges ARP replies to associate its own MAC address with the IP address of a legitimate device, thereby enabling man-in-the-middle interception, traffic redirection, session hijacking, and denial-of-service (DoS) conditions.

ARP spoofing remains a prevalent threat in both traditional enterprise networks and emerging software-defined networking (SDN) environments. Attackers frequently exploit this vulnerability to bypass perimeter security controls, inject malicious payloads, or disrupt critical services. Prior studies have demonstrated the feasibility of such attacks using commonly available tools such as arpspoof and dsniff, highlighting the urgent need for efficient and deployable defense mechanisms.

Several mitigation approaches have been proposed in the literature. Hardware-based solutions such as Dynamic ARP Inspection (DAI) enforce trusted IP-MAC bindings at the switch level but require managed network infrastructure and complex VLAN configuration, limiting adoption in small and cost-sensitive networks [7]. Software-defined networking (SDN)-based defenses centralize ARP validation at the controller layer, offering scalability but introducing architectural dependencies and compatibility constraints [9], [13], [16].

Machine learning-based intrusion detection systems have also been explored for ARP spoofing detection [8], [11], [14]. While these methods achieve high classification accuracy, they impose significant computational overhead, require extensive labeled datasets, and introduce latency during feature extraction and model inference. Such constraints reduce their suitability for real-time deployment on resource-constrained hosts and edge devices.

In parallel, modern network attacks increasingly combine ARP spoofing with volumetric denial-of-service (DDoS) techniques such as TCP SYN floods, UDP floods, and ICMP floods. Most existing ARP defense mechanisms focus solely on address mapping integrity and fail to provide integrated protection against multi-vector network attacks.

To address these limitations, this paper proposes NetShield, a lightweight, host-based intrusion detection and prevention system (IDPS) that combines real-time ARP spoofing detection with protocol-specific DDoS anomaly monitoring. NetShield employs a rule-based IP-MAC binding verification model that tracks ARP reply inconsistencies within a temporal window to identify spoofing behavior. Upon confirmation, malicious hosts are automatically blocked using firewall rules, enabling rapid containment without manual intervention.

In addition to detection and mitigation, NetShield enhances operational visibility by integrating with a Python-based logging service that forwards structured security events to Firebase. This enables a real-time web dashboard for attack visualization, traffic analytics, and historical incident tracking, providing administrators with actionable insights for real-time monitoring, rapid response, and post-incident analysis.

II. Literature Survey

Paper [1] The first paper presents a host-based security mechanism called ARP Guard that continuously monitors ARP packets and verifies IP-MAC bindings. The system detects spoofing by identifying inconsistencies in ARP responses and blocks malicious traffic in real time. Although effective, the approach requires installation on every host, making large-scale deployment difficult.

Paper [2] This paper proposes a WLAN-focused detection framework that analyzes ARP traffic patterns to identify spoofing attempts. It employs threshold-based detection and ARP table verification to prevent unauthorized address mapping. While the solution performs well in wireless environments, it lacks centralized logging and automated response features.

Paper [3] The authors introduce an active detection strategy that injects probe packets into the network to validate IP-MAC bindings. This approach improves detection accuracy by confirming legitimate hosts, but it introduces additional network overhead and is not suitable for high-speed enterprise networks.

Paper [4] This work focuses on securing smart microgrid networks against ARP spoofing by deploying a real-time intrusion detection system. The system monitors ARP packet inconsistencies to detect malicious behavior. Although highly effective for industrial control systems, it is not adaptable to general LAN environments.

Paper [5] This paper provides a comparative analysis of existing ARP spoofing detection techniques, including static ARP entries, IDS-based methods, and cryptographic solutions. It discusses the strengths and limitations of each approach and highlights the trade-offs between detection accuracy, deployment complexity, and system overhead.

Paper [6] The authors survey multiple machine learning-based techniques such as decision trees, SVMs, and neural networks for detecting ARP spoofing attacks. The study emphasizes improved accuracy but also points out issues related to high training cost, dataset dependency, and real-time deployment challenges.

Paper [7] This paper simulates Cisco's Dynamic ARP Inspection (DAI) mechanism using Cisco Packet Tracer to prevent ARP poisoning. The method effectively blocks spoofed ARP packets at the switch level but requires managed network hardware and VLAN configurations, limiting its usability in low-cost environments.

Paper [8] The authors propose a machine learning-based ARP spoofing detection model tailored for IoT networks. The study evaluates detection accuracy versus computational

complexity and concludes that lightweight ML models are more suitable for resource-constrained devices. However, deployment complexity remains a major challenge.

Paper [9] This paper introduces a deception-based intrusion detection system for software-defined networks (SDNs). It deploys fake network entities to lure attackers and identify malicious ARP behavior. While effective in SDN environments, the approach requires complex controller-based orchestration.

Paper [10] The study presents a traffic analysis-based framework that tracks abnormal ARP behavior patterns to identify spoofing incursions. The system is effective for attack detection but does not provide automated mitigation or blocking mechanisms.

Paper [11] This work combines real-time traffic monitoring with supervised machine learning classifiers to detect and prevent ARP spoofing. The system achieves high detection accuracy but depends heavily on labeled datasets and introduces higher computational overhead.

Paper [12] The authors provide a comprehensive survey of ARP poisoning techniques and defense mechanisms. The paper discusses limitations of existing solutions and highlights the need for hybrid detection models that combine rule-based and intelligent techniques.

Paper [13] This paper proposes an SDN-based framework that uses centralized controllers to detect and mitigate ARP spoofing attacks in real time. Although highly scalable, the solution requires SDN infrastructure and is incompatible with traditional LAN environments.

Paper [14] The authors design a machine learning-driven intrusion detection system that extracts features from ARP traffic to classify spoofing attacks. The method improves detection accuracy but introduces latency and requires frequent model retraining.

Paper [15] This paper presents a lightweight rule-based detection and firewall-based mitigation system for ARP spoofing. It focuses on fast response and low deployment cost but does not include visualization or cloud-based logging capabilities.

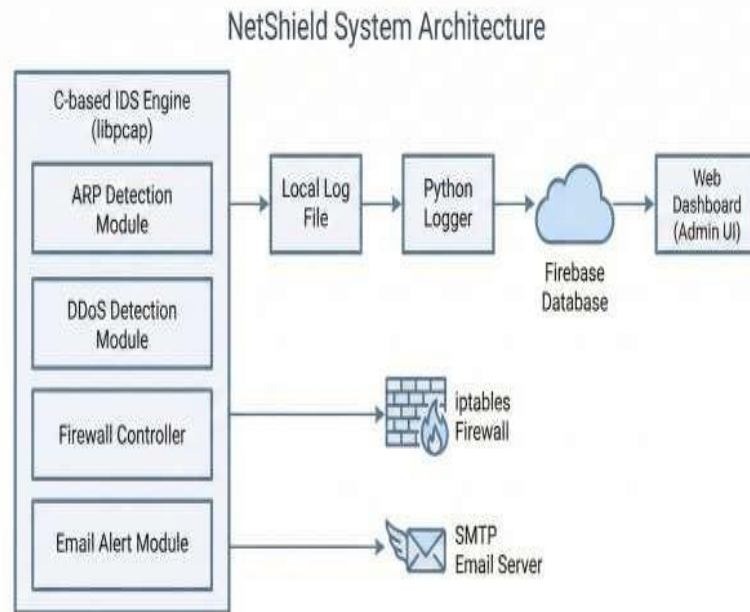
Paper [16] The authors propose an SDN-integrated ARP spoofing detection mechanism that leverages controller-level traffic analysis. While effective in virtualized environments, it is not suitable for standalone or resource-constrained systems.

III. Proposed System

This paper proposes NetShield, a lightweight Intrusion Detection and Prevention System (IDPS) designed to detect and mitigate ARP spoofing and volumetric DDoS attacks in real time. The solution is implemented as a host-based monitoring service that passively observes network traffic, applies rule-based detection logic, and performs automated mitigation without requiring specialized hardware or machine learning models.

NetShield follows a modular architecture, as illustrated in Fig. 1 (System Architecture Diagram), and consists of four primary components:

1. Packet Capture Module
2. ARP Spoofing Detection Engine
3. DDoS Anomaly Detection Engine
4. Mitigation, Logging, and Visualization Module



The system operates on a monitoring host connected to the network in promiscuous mode. Live traffic is captured using the libpcap library with a Berkeley Packet Filter (BPF) configured to process only ARP and IP packets. This selective filtering reduces computational overhead and ensures real-time processing capability.

The Packet Capture Module extracts essential metadata from each packet, including source IP address, source MAC address, destination IP address, protocol type (TCP, UDP, or ICMP), and relevant transport-layer fields such as TCP flags. These extracted attributes are forwarded to the corresponding detection engines based on protocol classification.

The ARP Spoofing Detection Engine maintains a dynamic in-memory table that maps IP addresses to their observed MAC addresses. For every ARP packet, the system verifies whether the IP-MAC binding is consistent with previously recorded values. If a mismatch is detected within a defined temporal window, the event is treated as a potential spoofing attempt. A confirmed attack is raised only when repeated mismatches exceed a predefined threshold, thereby reducing false positives caused by legitimate network changes such as DHCP renewals or hardware replacements.

The DDoS Anomaly Detection Engine analyzes IP traffic to detect volumetric attacks such as TCP SYN floods, UDP floods, and ICMP floods. For each source IP, the system tracks protocol-specific counters and total packet rates within a sliding time window. Alerts are triggered when traffic volumes exceed predefined thresholds or when SYN packets dominate the traffic profile, indicating a potential flooding condition.

Upon confirmation of either an ARP spoofing attack or a DDoS attack, NetShield initiates automated mitigation actions. The attacker's IP address is blocked using firewall rules at the operating system level, preventing further malicious traffic from reaching the network. Simultaneously, structured log entries are generated and forwarded to a Python-based logging service.

To enhance situational awareness, NetShield integrates with a cloud-based monitoring layer using Firebase. The Python logger pushes security events to Firebase in real time, enabling a web-based dashboard to visualize attacks, display historical trends, and provide administrators with actionable insights. This end-to-end design transforms

NetShield from a passive IDS into a fully operational intrusion detection and prevention platform.

NetShield assumes an attacker operating within the same local area network as the victim hosts. The attacker is capable of sending forged ARP replies, generating high-rate network traffic, and launching volumetric flooding attacks such as TCP SYN floods, UDP floods, and ICMP floods using commonly available tools. The attacker is assumed to have no administrative privileges on the NetShield monitoring host and cannot bypass operating system-level firewall enforcement once mitigation rules are applied. External internet-based attacks and physical compromise of network devices are considered outside the scope of this work.

IV. Workflow

The operational workflow of NetShield follows a multi-stage processing pipeline designed to enable real-time detection, automated mitigation, and continuous monitoring of ARP spoofing and DDoS attacks. The interaction among system components is illustrated in Fig. 1. The workflow begins with passive traffic acquisition and proceeds through protocol classification, attack detection, mitigation, and cloud-based logging for visualization and analysis.

NetShield continuously captures live network traffic using the libpcap library operating in promiscuous mode. To reduce processing overhead, a Berkeley Packet Filter is applied to restrict packet capture to ARP and IP traffic. Captured packets are parsed to extract essential metadata, including source and destination IP addresses, MAC addresses, protocol types, and control flags, which are then forwarded to the corresponding detection modules.

For ARP traffic, NetShield maintains a dynamic ARP table that records observed IP-MAC bindings along with timestamps. Incoming ARP packets are validated by comparing the sender MAC address against the stored binding for the corresponding IP address. Repeated MAC mismatches occurring within a predefined temporal window are treated as spoofing attempts. Once the alert threshold is exceeded, the source is classified as malicious, triggering immediate mitigation.

For IP traffic, NetShield performs protocol-aware traffic analysis by maintaining per-source-IP packet counters for TCP SYN, UDP, ICMP, and total packet volume. Traffic behavior is evaluated within a sliding time window using predefined thresholds to identify TCP SYN floods, UDP floods, ICMP floods, and generic high-rate flooding attacks. Sources that violate these thresholds are flagged as attackers.

Upon confirmation of either ARP spoofing or DDoS activity, NetShield initiates an automated response by dynamically inserting firewall rules to block all traffic from the offending IP address. Detected incidents are logged and forwarded to a Python-based logging service, which uploads structured event records to Firebase. A web-based dashboard retrieves this data to provide real-time alerts, historical attack timelines, and network security insights, enabling effective monitoring and post-incident analysis.

V. Detection Algorithms

Algorithm 1: ARP Spoofing Detection Algorithm

The ARP spoofing detection algorithm verifies the consistency of IP-MAC bindings observed in ARP packets within a defined temporal window. Repeated MAC address mismatches for the same IP address are treated as spoofing attempts and trigger automated mitigation.

Input: Live ARP packets

Output: Detection and blocking of ARP spoofing attacker

```

Initialize IP_MAC_Table
Initialize Mismatch_Counter
Set Mismatch_Threshold = T

For each captured ARP packet do:
    Extract Sender_IP and Sender_MAC
    If Sender_IP exists in IP_MAC_Table then
        If Sender_MAC  $\neq$  Stored_MAC then
            Increment Mismatch_Counter[Sender_IP]
            If Mismatch_Counter[Sender_IP]  $\geq$  Mismatch_Threshold then
                Block Sender_IP using firewall rules
                Log spoofing event
            End If
        Else
            Store Sender_IP  $\rightarrow$  Sender_MAC in IP_MAC_Table
        End If
    End For

```

Algorithm 2: DDoS Detection Algorithm

The DDoS detection algorithm monitors protocol-specific traffic rates within a sliding time window. Traffic sources exceeding predefined thresholds are classified as attackers and immediately mitigated.

Input: Live IP packets
 Output: Detection and blocking of DDoS attacker

```

Initialize Packet_Counters per Source_IP
Initialize Sliding_Time_Window
Set Protocol_Thresholds

For each captured IP packet do:
    Identify Source_IP and Protocol_Type
    Increment Packet_Counters[Source_IP][Protocol_Type]

    If Packet_Counters exceed Protocol_Thresholds within window then
        Block Source_IP using firewall rules
        Log DDoS event
    End If
End For

```

VI. Implementation

The NetShield intrusion detection and response system is implemented as a lightweight, host-based security solution for Linux environments. The core detection engine is developed in C to ensure high performance and low latency, while auxiliary logging and visualization components are implemented using Python and cloud-based services. Live network traffic is captured using the libpcap library operating in promiscuous mode. A Berkeley Packet Filter is applied to selectively process ARP and IP packets, reducing processing overhead. Packet headers are parsed to extract IP addresses, MAC addresses, protocol types, and transport-layer fields required for attack detection. ARP spoofing detection is implemented using a dynamic IP–MAC binding table that tracks observed mappings within a defined time window. Repeated MAC inconsistencies for the

same IP address within this window trigger spoofing alerts. DDoS detection is performed by monitoring protocol-specific traffic patterns, including TCP SYN, UDP, and ICMP packets, using per-IP counters and threshold-based analysis.

Upon attack confirmation, NetShield automatically blocks malicious IP addresses using operating system-level firewall rules. Security events are logged and forwarded to a Python-based service that uploads structured records to Firebase, enabling real-time visualization through a web-based dashboard.

VII. Experimental Evaluation

The proposed NetShield system was evaluated in a controlled virtual local area network environment to assess its effectiveness in detecting and mitigating ARP spoofing and volumetric DDoS attacks. The experimental setup consisted of multiple virtual machines running Kali Linux and Ubuntu, interconnected through a virtual switch to emulate a standard Ethernet LAN. Kali Linux was configured as the attacker node, while Ubuntu systems acted as the victim host and the NetShield monitoring node.

ARP spoofing attacks were generated by transmitting forged ARP reply packets to poison the victim's ARP cache and redirect network traffic. Volumetric DDoS attacks, including TCP SYN floods, UDP floods, and ICMP floods, were launched at varying packet rates to evaluate NetShield's detection accuracy, response time, and robustness under high traffic conditions. NetShield continuously monitored live traffic using the libpcap library and applied rule-based IP-MAC binding verification and protocol-specific traffic analysis in real time.

A. Evaluation Metrics

The system was evaluated using the following metrics: detection accuracy, false positive rate, detection latency, and system overhead. Detection accuracy represents the percentage of attack instances correctly identified by the system. False positive rate denotes the proportion of legitimate traffic incorrectly classified as malicious. Detection latency measures the time elapsed between the initiation of an attack and the enforcement of mitigation actions. System overhead was measured in terms of CPU utilization and memory consumption during active monitoring.

B. Detection Performance

Table I: Detection Performance of NetShield

Attack Type	Detection Accuracy (%)	False Positive Rate (%)	Avg Detection Time (ms)
ARP Spoofing	98.6	1.4	110
TCP SYN Flood	96.8	2.2	180
UDP Flood	95.9	2.6	200
ICMP Flood	97.3	1.9	170

The results indicate that NetShield achieves high detection accuracy across all evaluated attack scenarios. ARP spoofing attacks were detected with minimal latency due to the

lightweight IP-MAC binding verification mechanism. DDoS attacks were accurately identified based on protocol-specific packet rate thresholds, enabling timely mitigation before significant service degradation occurred. The low false positive rates demonstrate that legitimate network traffic was largely unaffected during normal operation.

C. System Overhead Analysis

Table II: System Resource Overhead

Metric	Average Value
CPU Utilization Increase	4–6%
Memory Consumption	~28 MB
Packet Processing Capacity	>50,000 packets/sec

The observed system overhead remained minimal, confirming that NetShield can operate continuously without imposing significant performance penalties on the monitoring host. The use of rule-based detection logic and selective packet capture enables efficient real-time processing even under high traffic loads.

D. Discussion

Overall, the experimental results demonstrate that NetShield provides effective real-time detection and automated mitigation of ARP spoofing and volumetric DDoS attacks while maintaining low computational overhead. The integration of cloud-based logging and visualization further enhances situational awareness by enabling real-time monitoring and historical analysis of security events. These characteristics make NetShield suitable for deployment in small-to-medium enterprise networks and academic environments.

VIII. Comparative Analysis

This section compares the proposed NetShield system with existing ARP spoofing detection and mitigation approaches reported in the literature. Traditional hardware-based solutions such as Dynamic ARP Inspection (DAI) provide effective protection against ARP spoofing but require managed switches and complex configuration, limiting their applicability in low-cost and academic environments. In contrast, NetShield operates entirely at the software level and does not depend on specialized network hardware.

Machine learning-based detection techniques achieve high detection accuracy in controlled environments; however, they typically require extensive training datasets, feature extraction, and model tuning, resulting in higher computational overhead and reduced real-time responsiveness. NetShield avoids these limitations by employing lightweight rule-based IP-MAC binding verification and protocol-specific traffic analysis, enabling faster detection with minimal system resource consumption.

Software-defined networking (SDN)-based ARP defense mechanisms centralize monitoring and enforcement but introduce additional architectural complexity and controller dependency. NetShield follows a decentralized host-based approach, making it easier to deploy and maintain in conventional networks without modifying existing infrastructure.

Overall, compared to existing approaches, NetShield offers a balanced trade-off between detection accuracy, deployment simplicity, and runtime efficiency. Its ability to provide real-time detection, automated mitigation, and cloud-based visualization makes it a practical alternative for small-to-medium enterprise networks and educational environments.

IX. Limitations And Future Work

Although NetShield demonstrates effective real-time detection and mitigation of ARP spoofing and volumetric DDoS attacks, certain limitations remain. The current implementation relies on predefined thresholds and rule-based logic, which may require tuning for highly dynamic or large-scale network environments. Additionally, the system primarily focuses on IPv4 networks and does not address Neighbor Discovery Protocol (NDP) spoofing in IPv6.

Future work will focus on extending NetShield to support IPv6-based attacks, adaptive thresholding based on traffic profiling, and integration of lightweight machine learning techniques to enhance detection accuracy under evolving attack patterns. Further evaluation in larger and heterogeneous network environments is also planned to assess scalability and robustness.

X. Conclusion

This paper presented NetShield, a lightweight intrusion detection and response system designed to detect and mitigate ARP spoofing and volumetric DDoS attacks in real time. By leveraging rule-based IP-MAC binding verification and protocol-specific traffic analysis, NetShield effectively identifies malicious behavior without relying on specialized hardware or computationally intensive machine learning models. The experimental evaluation timely detection, automated mitigation, and minimal system overhead in a virtualized network environment. With its simplicity, effectiveness, and real-time monitoring capabilities, NetShield represents a practical security solution for small-to-medium enterprise networks and academic environments.

XI. References

1. Navaneetha C. et al., "ARP Guard: Advanced Detection and Defence Against Spoofing Attacks," *TIJER*, 2025.
2. Balamurugan A., "An Effective Approach to Detect and Prevent ARP Spoofing Attacks on WLAN," *IJRASET*, 2025.
3. Ashok Bawge & Harish Joshi, "Identifying ARP Spoofing Through Active Strategies," 2025.
4. Kalinath Katuri et al., "Real-Time Detection and Identification of ARP Spoofing Attacks in Microgrids," *IEEE Access*, 2025.
5. R. Singh & P. Verma, "Analysis of ARP-Spoofing Attack Detection Methods in the Network," *IJNS*, 2024.
6. Buli Fakada & Gaurav Gupta, "Machine Learning Techniques for Detecting ARP Spoofing Attack: A Review," *JETIR*, 2025.
7. Fauzan P. E. Putra et al., "Implementation of Dynamic ARP Inspection," *Brilliance*, 2024.
8. Adeeb Alsaaidah et al., "ARP Spoofing Detection Model in IoT Networks," *JADS*, 2025.
9. Fabrice Mvah et al., "Deception-based IDS against ARP Spoofing Attacks," *ICNC*, 2024.
10. A. Kumar & R. Mehta, "Identifying ARP Spoofing Incursions," *IJCRT*, 2024.
11. Mrinal Kumar & C. S. Dash, "Detecting and Preventing ARP Spoofing Using ML," *IJIRCST*, 2024.
12. S. Patel & R. Sharma, "A Survey on ARP Poisoning," *IJCSR*, 2025.

13. Vidyanandan Hnamte & Jamal Hussain, "Enhancing Security in SDN," Elsevier, 2024.
14. K. Narayanan & S. Iyer, "Enhanced Cybersecurity: Detecting ARP Spoofing Using ML," IJAC, 2024.
15. Prema Arokiya Mary et al., "ARP Spoof Detection and Mitigation," IJERT, 2023.
16. Saritakumar N. et al., "Detection of ARP Spoofing Attacks in SDN," IEEE, 2023.