Journal of Harbin Engineering University
ISSN: 1006-7043

Vol 44 No. 9
September 2023

Proactive Defence Against ARP Poisoning: ARule-Based Solution

Abhijna PS
Department of Computer Science School of Computing, Mysore Amrita Vishwa Vidyapeetham India
Santosh Anand
Department of Computer Science School of Computing, Mysore Amrita Vishwa Vidyapeetham India
Akhil K M
Department of Computer Science School of Computing, Mysore Amrita Vishwa Vidyapeetham India

Abstract—Ensuring the security of data transmitted over a network is a critical aspect of modern-day internet usage. Man-in-the-Middle (MITM) is one of the prevalent form of network secu- rity vulnerability, where in the communication channel attacker listens between two hosts to obtain their data. When it comes to Local Area Network (LAN), as a destination address to send data Media Access Control (MAC) address is utilized, and Address Resolution Protocol (ARP) helps to resolve the IP to MAC mechanism. However, the ARP protocol's lack of authentication makes it vulnerable to attackers who can easily execute a MITM attack using the ARP poisoning technique. In this paper, proposes an algorithm that defends against ARP poisoning attacks by employing a Rule-Based solution implemented in one of the host in the network which will be considered as an Admin host. By having the Admin host, client-side implementation is not required. Once an ARP poisoning attack is detected, the Rule-Based algorithm aids in recovery by utilizing a legitimate ARP packet containing the appropriate contents. This proactive measure effectively prevents the ARP poisoning attack from further compromising the network.
Index Terms—ARP poisoning, Monitoring Host, Address res- olution protocol (ARP), Man in the Middle (MITM).

I. Introduction

The internet has become an essential tool for daily activi- ties in today's world. To ensure privacy and security during communication and data transfer across the globe, individuals and organizations require reliable protocols like Hypertext Transfer Protocol Secure (HTTPS) which is based on the Transport Layer Security (TLS) encryption [1]. However, as no technology or software is perfect, vulnerabilities still exist, and they must be addressed. One such type of attack that poses a significant challenge for prevention is known as the Man in the Middle (MITM) attack. So this MITM includes intercepting and monitoring the network traffic, by which an attacker can gain unauthorized access to data. One technique for carrying out a MITM attack is through ARP poisoning.

Whenever the host wants to send data to an another host or server outside its subnet, it sends data or requests to the host's IP address. When it comes to inside the subnet i.e Local Area Network (LAN), as a target/destination address Media Access Control (MAC)is considered, so which ever the host wants to send data to another host the data will be sent to destination host's MAC address. To resolve this

IP-to-MAC mechanism in LAN, Address Resolution Protocol(ARP) is utilized. When a host in a network intends to send data to a specific destination host, it initiates communication by sending an ARP request packet to all the hosts within the network stating, *Who as 'Target IP address'? Tell 'Target IP address'* [2]. The computer associated with the queried IPaddress replies MAC address as an ARP reply packet, the remaining host will just ignore the request. Upon receiving the ARP reply packet from the host that initiated the request, the MAC address is stored in the "ARP cache" table, whereit is associated with the corresponding target IP address. Consequently, whenever the host intends to transmit data, it can conveniently refer to the ARP cache to obtain the MAC address of the destination host and send the data accordingly. Since in the ARP protocol there is no authentication process [3], malicious actors in the network take this as the advantage to execute the MITM. The

motivation for this research is driven by the persistent vulnerability of networks to ARP poisoning attacks, which remains prevalent in the present day. These attacks, known as man-in-the-middle (MITM) attacks, can have severe consequences, including session hijacking, data manipulation, phishing attacks, eavesdropping, and mal- ware injection. Recognizing the risks associated with MITM attacks is crucial for the development of effective coun- termeasures that safeguard sensitive information and ensure the integrity and confidentiality of network communications. This paper presents a Rule-Based system implemented on the Admin host, eliminating the need for client-side changes. The system effectively detects and prevents ARP poisoning attacks without modifying the ARP protocol. It offers scalability, making it suitable for networks with numerous hosts. By centralizing the implementation, the system ensures consistent protection across the network infrastructure. This approach provides an efficient solution for mitigating ARP poisoning attacks without disrupting network operations.

The paper is structured as follows: Section II provides a comprehensive explanation of the problem statement, while Section III presents a literature survey. In Section IV, we introduce the proposed algorithm, followed by the description of its implementation in Section V. Finally, the results are discussed in Section VI, and the paper concludes with a conclusions in Section VII.

BACKGROUND

If a host in a network requires to transmit data to another host, it needs to first determine the destination address. In this scenario, when Host A needs to transmit data to Host B, it starts the procedure by sending a packet with an ARP request to all hosts within the network, requesting Host B's MAC address, as shown in Fig.1. The packet contains the message *Who has 'Host B IP'? and Tell 'Host B IP'*. Here Host D ignores the request because the IP address does not match its own, but Host B replies with the MAC address as an ARP reply stating, *'Host B IP' is at 'Host B MAC'*. Host A saves Host B's MAC address in its ARP cache after receiving the ARP reply packet. As a result, Host A can send data to Host B with ease by simply referencing the MAC address stored in the ARP cache table.

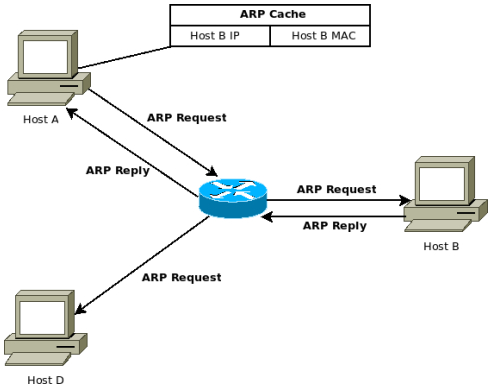


Fig. 1. Address Resolution Protocol

One issue with ARP is the lack of authentication, which makes it impossible to confirm whether the ARP reply packet came from the desired destination host that Host A wants to communicate with. This is because the ARP protocol simply maps the IP to MAC but it does not provide any means for verifying the authenticity of the ARP reply packet. Therefore, an attacker can easily change the contents of ARP reply packetleading to MITM.

A. ARP Poisoning Attack

In LAN network shown in Fig.2, Host D will act as the malicious actor. Now when the Host D wants to

Listen between the Host A and Host B by establish the MITM, Host D simply send the forged ARP request packet, saying "Who has 192.168.1.11, tell 192.168.13 is at 7C:0A:3B:54:31." Since there is no authentication in Address Resolution Protocol, Host A thinks it came from 192.168.1.13 and refreshes its ARP cache table. So now whenever Host A wants to transmit data to Host B (192.168.1.13), it looks for the IP in the cache table and sends it to the MAC associated with the IP. In this case as shown in Fig.2 the Host B IP is mapped to Host D's MAC, so any data that Host A sends to Host B will be sentto Host D where Host D can use routing protocols to redirect

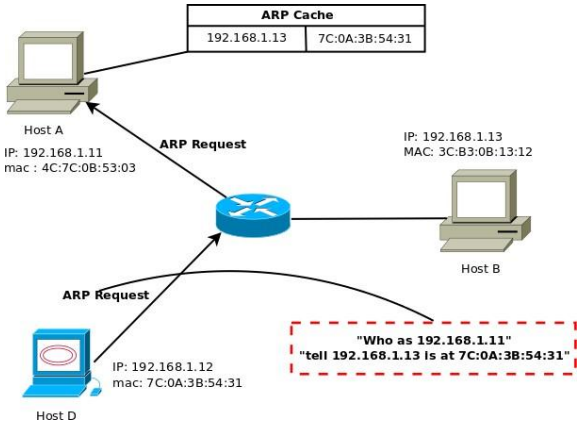


Fig. 2. ARP poisoning using ARP request packet

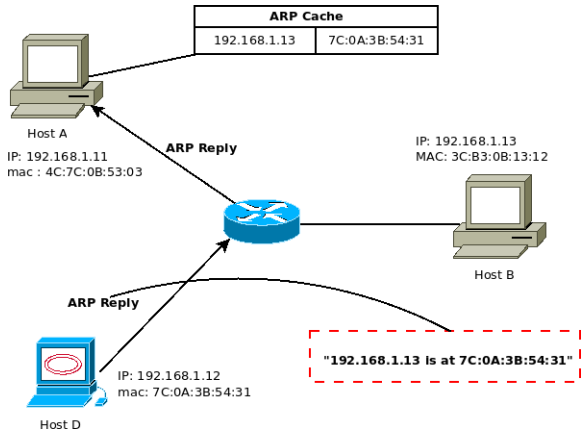


Fig. 3. ARP poisoning using ARP reply packet

those packets to Host B making it appear as normal network behavior.

The same process is done using ARP poisoning using ARP reply packet, where the ARP reply packet will be forged as "192.168.1.13 is at 7C:0A:3B:54:31." as shown in the Fig.3. After receiving the packet by Host A, Host A simply refreshes by updating its ARP cache table with this information. Now whenever the Host A sends the data to Host B will just be sent to Host D because the MAC associated with Host B IPis Host D's MAC.

B. Normal ARP Packet and Forged ARP Packet

Table Inormal Arp Reply

Format	Host A to Host B
Sender IP Address	Host A IP
Sender MAC Address	Host A MAC
Target IP Address	Host B IP
Target MAC Address	Host B MAC

An attacker uses the ARP poisoning attack to manipulate ARP packets and send them to a victim. ARP reply packet (as shown in Table 1) containing Host A's MAC and associated IP

Table Iiforged Arp Reply

Format	Host D to Host A
Sender IP Address	Host B IP
Sender MAC Address	Host D MAC
Target IP Address	Host A IP
Target MAC Address	Host A MAC

address is typically sent in response to an ARP request from Host B. But in an MITM attack, the malicious actor (Host D) has the ability to spoof an ARP reply packet and send it to any host on the network. According to Table 2, this forgery has Host D as the Sender MAC address but Host B as the Sender IP address. The attack can establish the MITM between the hosts A and B using this forged ARP packet.

II. Related Works

There have been various proposed solutions to the lack of authentication in ARP, including S-ARP [2], Enhanced ARP [4], and Ticket-based ARP [5]. S-ARP is an extension that en- ables authentication of ARP packets using cryptography while being backward compatible. Each computer in the LAN would use its own public/private key certificates to authenticate, and the receiver would use the sender's public key for authentica- tion. However, S-ARP requires replacing ARP on every host, which is not scalable. Ticket-based ARP, on the other hand, appends a ticket to ARP packets and distributes it to hosts through a Local Ticket Agent. The packets are authenticated using public key encryption, providing authentication without restructuring of ARP. Enhanced ARP suggests the utilization of a secondary ARP cache table with short and long-term update policies. The incoming ARP packet is cross-checked with the long-term cache table to confirm its authenticity.

Hou et al., [6] paper suggests a modification to the existing network tool called Snort. Where adding certain rules based on the activity of ARP poisoning which the Snort will detect for the behaviour mentioned in the rules, with that ARP poisoning is detected.

Al Abri et al., [7] The proposed method called Payload Matching , involves sampling the traffic and processing the payload matching, which is a intensive task for computingand can create a single point of failure. Abdulla et al., [8] proposed a neural network-based algorithm for detection of ARP Spoofing attacks in IoT networks, which overcomes the limitations of existing "Auto regressive Integrated Moving Average" (ARIMA) methods. To detect ARP Spoofing attacks, a technique was developed that feeds UDP, TCP, and ARP packets

into multiple neural networks. By evaluating the MeanSquared Error (MSE), a suitable neural network architecture was selected. This architecture consisted of a hidden layer anda single output layer, ultimately achieving a detection accuracy rate exceeding 90%. Sebbar et al., [9] employed Machine Software-defined network (SDN) infrastructures which could detect malicious nodes automatically without compromising system performance.

Kumar et al., [10] Another approach is the use of an ARP central server (ACS) , which compares the ARP requests and reply from the clients and sends the proper IP and MACaddress in case if it differs, thus preventing ARP poison- ing. Majumdar et al., [11] have proposed an algorithm for detection of the ARP Poisoning attacks by examining the differences between the MAC in the ARP Packet and the response MAC. Additionally, they have provided a prevention algorithm that uses static entries in the ARP table. Prevention algorithms were implemented using scapy library in Python language. Setpanov et al., [12] conducted an examination of the prerequisites needed to carry out a MITM attack on ARP- based networks. They also explored methods for identifying and defending against such attacks. Additionally, they show- cased a Python implementation of an ARP spoofing attack. Agrawal et al., [13], the function and operation of ARP were explained, followed by a discussion of ARP Spoofing and its numerous attack methods. Various detection and mitigation techniques were compared to defend against ARP attacks, with an analysis of their advantages and disadvantages. Xia et al., [14], researchers investigated the ARP spoofing attackin networks and found that it can still pose a significant threat even in an SDN environment. To mitigate this issue, the authors proposed a new protection mechanism that relies on the OpenFlow environment. This approach was subjected to theoretical analysis and practically integrating it as a module of the Python-based OpenFlow eXperimentation platform (POX) controller, leading to a significant decrease in security threats associated with ARP spoofing on OpenFlow and other SDN environment.

Based on the description in above all paper, few Learning methods to identify and prevent MITM attacks. A Context-Based Node Acceptance-Random Forest (CBNA- RF) model developed by the researchers to enable efficient security policy creation and automated detection operations on

of thoseare trying to change the ARP protocol itself which will be a tedious task to implement in every client host if the network have more number of host. In the same way, there are few which have client side implementation which monitors the ARP cache table will not be scalable and some of it shows more processing necessary when they want to analzy the behavior of the network just for ARP poisoning. So to the author's knowledge, the proposed solution tries to reduce all the processing by focusing only on ARP packets in the network and no client side implementation. Further these modules will be explained more in Section IV and SectionV.

III. Proposed Algorithm

In the proposed algorithm, a Rule-Based system to defend against the ARP poisoning in the network. This algorithms demands no client-side implementation, and only one host is designated as the Admin host, responsible for monitoringthe entire network for any unusual traffic. The algorithm is designed to focus only on monitoring ARP packets, thereby making it more efficient and robust in detecting attacks.

A. Rule-based System

Based on scenarios derived from analyzing how ARP poi- soning attacks are executed in the network the Rule-Based system is designed. So the following rules mentioned below will be used to detect the ARP poisoning.

1. In a typical network, the ARP request is sent to entire network, and they should either receive the ARP reply or not receive it at all. Therefore, it's important to ensure that the ARP reply packets is not more then ARP request packets in the network, as this could be a sign of an attempted ARP poisoning attack.
2. The algorithm proceeds to verify each new combina- tion of IP/MAC that appears in the network by cross- referencing it with a database

3. containing IP and MAC information. If a new
4. address is already present in the database, but with a different MAC address, this indicates that the attacker is attempting to replace the MAC of a specific IP with their own.
5. Since the admin maintains record of all IP and MAC in the network, the algorithm can detect potential ARP poi- soning attempts by identifying instances where a single MAC address is associated with multiple IP addresses. This information can then be used to take appropriate actions to defend the attack.

B. Detection modules

In order to prevent any malicious activity in the network, it is important to first detect any attempts of such activity. The algorithm for detecting ARP packets are analyzed to keep track of the number of ARP reply and ARP request packets. When there are more ARP reply packets than ARP request packets in the network, it could indicate a possible ARP poisoning attack. Subsequently,the algorithm presents the most recent ARP packets to the Administrator, offering a summary of the network's current status. In order to verify the IP and MAC addresses within the ARP packets, a function call is made in line number 18 of Algorithm 2 to compare them with the host information table. Also this detection module runs continuously by updating the host info table here and then if in case of new host connects to the network and before the validation function call.

Algorithm 1 Network Scan

Require: Gateway address of the network
Ensure: List of clients with their IP and MAC addresses
1: Default = "Broadcast MAC"
2: arp request = Default/arp request
3: reply list = srp(arp request)[0]
4: clients = []
5: **for** elements in reply list **do**
6: host info = "ip": elements[1].psrc, "mac": ele-ments[1].hwsrc
7: clients.append(host info)
8: **end for**
9: **return** clients list

Algorithm 2 Monitor and Detect

Require: Captured ARP packets
Ensure: If there is attempt of ARP poisoning in a network
1: CALL **NetworkScan**(*gateway*)
2: arp request count = 0
3: arp reply count = 0
4: **if** ARP in packets **then**
5: **if** packet == arp request **then**
6: arp request count= arp request count + 1

combination is found and the IP poisoning attempts is based on rules mentioned in the Rule-based System subsection A, and this being implemented in the Admin Host with the necessary privileges to monitor all ARP packets in the network.

In Algorithm 1, referred to as Network Scan, the Admin- istrator broadcasts an ARP request packet across the entire network with the aim of retrieving the MAC addresses of all the hosts. After receiving the ARP reply, a table containing IP-MAC mappings is created. This host info table is updated every time an attack is detected to ensure to add any new host in the network which can be suspected as malicious actor. Coming to Algorithm 2 Monitor and Detect, captured ARP

```
7:      else
8:          arp reply count = arp reply count + 1
9:          if len(recent arp reply packets) lesser then 5 then
10:             recent arp packets.append(packet)
11:         else
12:             recent arp packets.pop(0)
13:             recent arp packets.append(packet)
14:         end if
15:         if arp reply count > arp request count then
16:             for p in recent arp packets do
17:                 print(p.summary())
18:                 is valid arp(packet, hosts list)
19:             end for
20:         end if
21:     end if
22: end if
```

In the Algorithm 3 Validation, the Validation function is called whenever an attempt to attack is detected. This function first checks if the combination of IP/MAC in ARP packet is present in the host info table. In case if MAC address within the ARP packet differs from the one associated with the IP address in the host information table, it indicates that the attacker is trying to modify the MAC of that IP address and then the algorithm attempts to identify the attacker's IP. Once the process is complete, the admin is notified of the details, this includes the IP and MAC of the poisoned host and as well as the IP address of the attacker.

C. Prevention Module

Before the prevention its important to confirm there is a malicious activity detected. Once an attack is notified, theAdmin must be aware of the continuous sending of forged ARP packets by the attacker to the victim's host. The algorithm

Algorithm 3 Validation

Require: current packet and host list

Ensure: Displays if attacker IP address found or it checks forfalse alarm

```
1: flag = 0
2: if packet == arp reply then
3:     sender ip = packet[ARP].psrc
4:     sender mac = packet[ARP].hwsrc
5:     receiver ip =packet[ARP].pdst
6:     receiver mac=packet[ARP].hwdst
7:     for host in host list do
8:         if sender ip == host[”ip”] then
9:             flag = 1
10:            if sender mac != host[”mac”] then
11:                print(”Victims IP”)
12:                print(”Attackers MAC”)
13:            end if
14:        end if
15:    end for
```

```
16:     if flag == 1 then
17:       for host in host list do
18:         if sender mac == host["mac"] then
19:           print("Attackers IP")
20:         end if
21:       end for
22:     end if
23: end if
```

Algorithm 4 Repair

Require: Details from the algorithm 3

Ensure: Proper ARP packet sent to recover from the attack

```
dst IP = victimsIP
dst MAC = victimsMAC
proper IP = repair IP
proper MAC = repair MAC
packet= ARP(op =2 , pdst= dst IP , psrc= proper IP ,hwdst= dst MAC , hwsrc=proper MAC)
send(packet)
```

continuously displays the recent ARP packets to the Admin with which the admin will be able to know if the attacker is sending continuous traffic then attacker's host can be blocked from the network with Admin privileges. Once the attacker's host is blocked, the Admin can recover the victim's host ARP cache table by using the Algorithm 4 Repair. This involves sending an ARP reply packet from the Admin host to the victim's host, In this scenario, the forged ARP packet sent by the attacker and the ARP packet used by the Admin share the same ARP protocol. However, the distinction lies in the content of the packets. The forged ARP packet, sent by the attacker, contains modified IP and MAC addresses that results in MITM, thus manipulating the victim's ARP cache. On the other hand, the ARP packet used by the Admin is a legitimate packet with accurate IP and MAC addresses sent to the victim's host. So the ARP packet received by the victim's host refreshes its cache table, effectively preventing the attack and recovering from it.

A. Hosts in the network

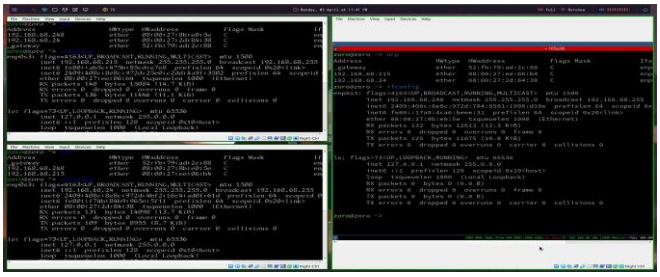
In Fig.4, the illustration presents the table containing the ARP cache, along with the output obtained from executing the "ifconfig" command for each host. The host with the IP address 192.168.68.248 shown in Fig.4, is considered as the at- tacker host, while the host with the IP address 192.168.68.220 serves as the monitoring host.

The Table 3 serves the purpose of presenting comprehensive information about the hosts within the LAN, including their re- spective MAC to the corresponding IP addresses. Additionally, the table indicates which hosts are

IV. IMPLEMENTATION

The proposed algorithm was implemented using three vir- tual machines in VirtualBox [15] with a bridged network adapter setup. This configuration allowed for the creation of a LAN that includes the main host as a monitoring node, result- ing in a total of four hosts in the network. The virtual machines were running Arch Linux [16] as the operating system, while the main host was running Debian-based Pop OS.

Fig. 4. Host in the network



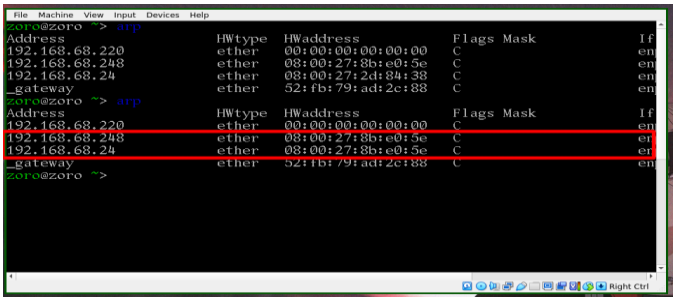
designated as the Attacker Host and Admin Host, serving as a reference point for further analysis and understanding of the output.

Table Iii Host Details

	IP	MAC
Host 1	192.168.68.215	08:00:27:ee:06:b4
Host 2	192.168.68.24	08:00:27:2d:84:38
Attacker Host	192.168.68.248	08:00:27:8b:e0:5e
Admin Host	192.168.68.220	34:e1:2d:14:dc:aa

B. Detection and Prevention

In the implementation, a Graphical User Interface (GUI)has been developed using the PyQt5 Python libraries [17].The GUI provides a flexible and user-friendly interface forthe admin to monitor and track the network. This also allows the admin to understand the network activities. In Fig.5, the



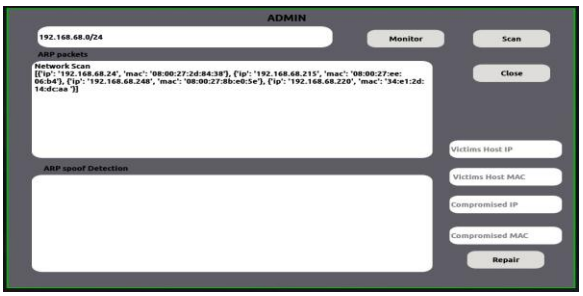


Fig. 5. Network Scan

gateway address (in this case 192.168.68.0/24) should be provided and the Scan button clicked to retrieve the details of all the hosts in the network. This gives an overall view of the network to the administrator and also updates the table used in the detection modules. The Monitor button capturesall the ARP packets in the network, which are displayed in theARP packet section. Any malicious activity and the attacker's IP are shown in the ARP spoof Detection section. The Repair module, located on the right-hand side, provides the proper IP and MAC address automatically which can be rechecked with output by admin and send it to the host which is poisoned. By this Admins can easily monitor all the background processes and get a clear understanding of the network's activity.

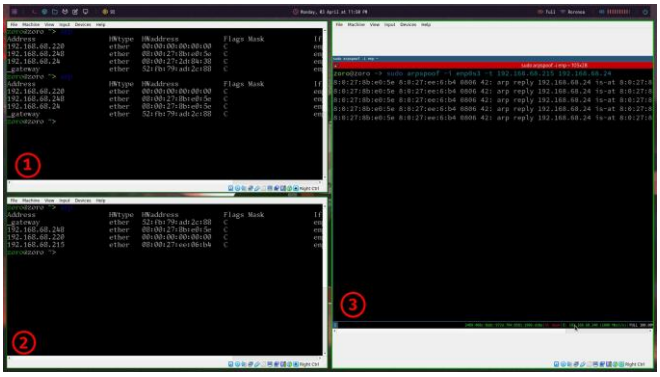


Fig. 6. ARP Poisoning Attack

In Fig.6, utilized an open source command line tool called arpspoof to conduct the ARP poisoning attack. The Host 3, with an IP of 192.168.68.248, was designated as the attacker and attempted to alter Host 2's address in the cache table of Host 1, as demonstrated in Fig.6. Upon initiation of the attack by Host 3, a modification occurred in the MAC address storedwithin the ARP cache, as illustrated in Figure 7. Where the MAC address of Host 2's IP matched the attacker's IP. As a

Fig. 7. ARP Poisoning Attack

result, any data transmitted from Host 1 to Host 2 would besent to the attacker.

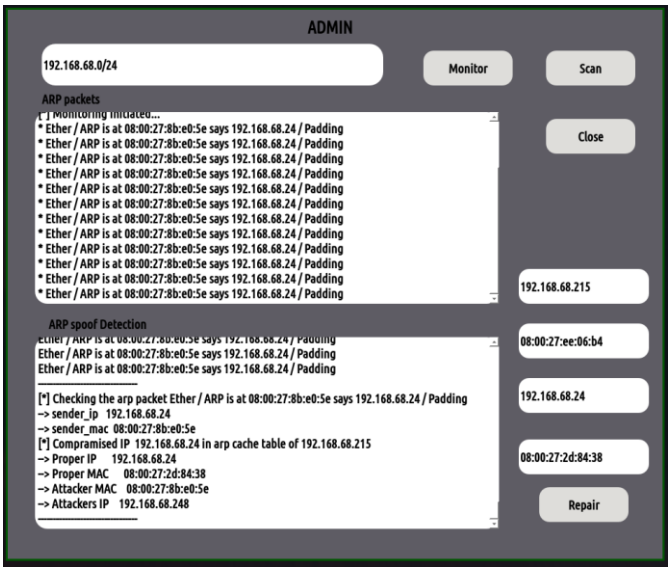


Fig. 8. ARP poisoning detection

As shown in Fig.8, The Admin host captures the network traffic. The "ARP packet" section in the GUI, displays allthe ARP packets. In the "ARP spoof detection" section, ARP poisoning is detected, along with relevant details. Recent suspected ARP packets are shown in Fig.8, followed by details of the poisoned host and attacker host. The administrator can block the attacker host by filling in the details on the rightside of the GUI. By clicking the "Repair" button resolves the issue by sending a packet to the poisoned host.

So as shown in Fig.9, the repair details which consists of poisoned host (Host 1) and the modified details in the poisoned host which is Host 2's (192.168.68.24) info. Once the packetis sent Host 1 we can see the repair done in the Fig.10 where all the MAC address is set proper in the Host 1's ARP cache table.

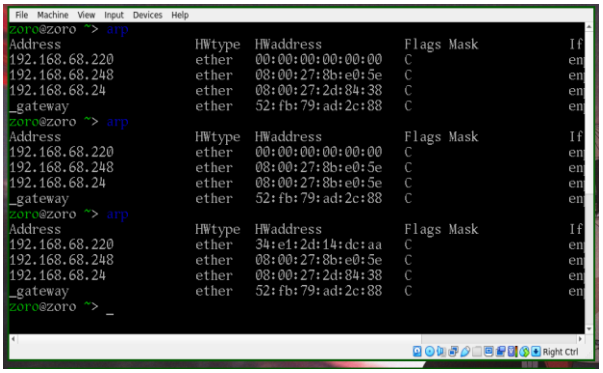


Fig. 9. Recover after the attack

V. Result

In Table 4, provides further insights into the results of the study. It demonstrates that when the simulated attack originated from the Attacker host, the Admin host effectively detected the attack and initiated the recovery process. This successful detection and recovery scenario demonstrates the effectiveness of the proposed method. Moreover, to com- prehensively evaluate the detection and prevention measures, attacks were simulated between Host 1 and Host 2 in both directions. This comprehensive approach allows for a thorough assessment of the system's ability to detect and prevent ARP attacks.

Table Iv Detection And Recovery Results For Simulated Arp Attacks

Attacking Host	Victim host	Detection	Prevention
Attacker Host	Host 2	Yes	Recovered
Attacker Host	Host 1	Yes	Recovered
Host 1	Host 2	Yes	Recovered
Host 2	Host 1	Yes	Recovered
Attacker Host	Admin Host	Yes	Recovered

It is worth noting that the Admin host requires manual input of its own details into the host information table. This step is crucial because during the creation of the table, the Admin host scans the entire network except for itself. By following this procedure, the proposed method ensures that ARP attacks can be accurately detected and effectively recovered from.

The findings presented in Table 4 align with the analysis in Figure 10, confirming the effectiveness of the detection and prevention measures against ARP poisoning. The data reveals consistent high detection rates, ranging from 96% to 98%, for various attack scenarios involving different hosts. Notably, in all cases, prevention efforts achieved a remarkable 100% success rate in recovering from the attacks. Moreover, the data emphasizes the importance of proper configuration and manual input of host details into the system, as demonstrated by the successful detection and prevention of attacks involving the Admin Host. This meticulous approach, despite requiring additional steps, guarantees comprehensive network scanning and protection, reinforcing the reliability of the proposed method.

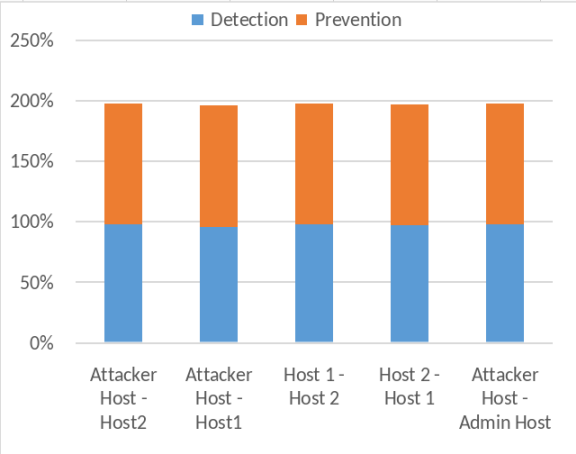


Fig. 10. Analysis of Detection and Prevention

Overall, the combination of the graphical analysis in Figure 10 and the data presented in Table 4 showcases the consistent and effective performance of the detection and prevention measures in safeguarding against ARP poisoning attacks.

VI.Conclusion

The ARP poisoning attack remains vulnerable in LAN or any network, allowing attackers to easily execute the MITM. To mitigate this issue, an algorithm is proposed that does not necessitate client-side implementation. Instead, a single host serves as a Admin node in the network, detecting abnormal network activities. The algorithm output provides

Admin withcomprehensive details on network traffic, including which host is being poisoned and how to recover from the attack. While understanding of network attacks is required, the algorithm provides a flexible and user-friendly method to prevent such attacks with the click of a button. Finally, results of the experiment demonstrate that the proposed algorithm safeguardagainst an ARP poisoning. Due to its built-in network monitor- ing scheme further this algorithm can be enhanced to include features that can detect other malicious attacks like Distributed Denial-of-service (DDOS), DNS spoofing , as well as MAC spoofing in the network.

References

[1] TechTarget, "Https," https://www.techtarget.com/searchsoftwarequality/ definition/HTTPS, accessed on 2023-05-29.

[2] D. Bruschi, A. Ornaghi, and E. Rosti, "S-arp: a secure address resolution protocol," in *19th Annual Computer Security Applications Conference, 2003. Proceedings.* IEEE, 2003, pp. 66–74.

[3] V. Goyal and R. Tripathy, "An efficient solution to the arp cache poisoning problem," in *Information Security and Privacy: 10th Australasian Conference, ACISP 2005, Brisbane, Australia, July 4-6, 2005. Proceedings 10*. Springer, 2005, pp. 40–51.

[4] S. Y. Nam, D. Kim, and J. Kim, "Enhanced arp: preventing arp poisoning-based man-in-the-middle attacks," *IEEE communications let- ters*, vol. 14, no. 2, pp. 187–189, 2010.

[5] W. Lootah, W. Enck, and P. McDaniel, "Tarp: Ticket-based address resolution protocol," *Computer networks*, vol. 51, no. 15, pp. 4322–4337, 2007.

[6] X. Hou, Z. Jiang, and X. Tian, "The detection and prevention for arp spoofing based on snort," in *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 5. IEEE, 2010, pp. V5–137.

[7] D. Al Abri, "Detection of mitm attack in lan environment using payload matching," in *2015 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2015, pp. 1857–1862.

[8] H. Abdulla, H. Al-Raweshidy, and W. S Awad, "Arp spoofing detection for iot networks using neural networks," in *Proceedings of the Industrial Revolution & Business Management: 11th Annual PwR Doctoral Symposium (PWRDS)*, 2020.

[9] A. Sebbar, K. Zkik, Y. Baddi, M. Boulmalf, and M. D. E.-C. E. Kettani, "Mitm detection and defense mechanism cbna-rf based on machine learning for large-scale sdn context," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 5875–5894, 2020.

[10] S. Kumar and S. Tapaswi, "A centralized detection and prevention tech- nique against arp poisoning," in *Proceedings Title: 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*. IEEE, 2012, pp. 259–264.

[11] A. Majumdar, S. Raj, and T. Subbulakshmi, "Arp poisoning detection and prevention using scapy," in *Journal of Physics: Conference Series*, vol. 1911, no. 1. IOP Publishing, 2021, p. 012022.

[12] P. Stepanov, G. Nikonova, T. Pavlychenko, and A. Gil, "The problem of security address resolution protocol," in *Journal of Physics: Conference Series*, vol. 1791, no. 1. IOP Publishing, 2021, p. 012061.

[13] G. Agrawal and V. Chennai, "Detection and prevention of arp-spoofing attacks," *International Journal of Engineering Research & Technology (IJERT)*, vol. 8, no. 10, 2019.

[14] J. Xia, Z. Cai, G. Hu, and M. Xu, "An active defense solution for arp spoofing in openflow network," *Chinese Journal of Electronics*, vol. 28, no. 1, pp. 172–178, 2019.

[15] VirtualBox, "VirtualBox documentation," https://www.virtualbox.org/ wiki/Documentation, 2023, [Accessed on 28th May 2023].

[16] "Arch linux installation guide," https://wiki.archlinux.org/title/ installation guide, accessed: February 8, 2023.

[17] M. Fitzpatrick. (2023) Pyqt5 tutorial. Accessed: March 15, 2023. [Online]. Available: https://www.pythonguis.com/pyqt5-tutorial/

[18] G. Jinhua and X. Kejian, "Arp spoofing detection algorithm using icmp protocol," in *2013 International Conference on Computer Communica- tion and Informatics*. IEEE, 2013, pp. 1–6.

[19] P. Arote and K. V. Arya, "Detection and prevention against arp poi- soning attack using modified icmp and voting," in *2015 International Conference on Computational Intelligence and Networks*. IEEE, 2015, pp. 136–141.

[20] M. Data, "The defense against arp spoofing attack using semi-static arp cache table," in *2018 International conference on sustainable information engineering and technology (SIET)*. IEEE, 2018, pp. 206– 210.

[21] R. Abhijith and B. S. Kumar, "First level security system for intrusion detection and prevention in lan," in *2021 2nd International Conference for Emerging Technology (INCET)*. IEEE, 2021, pp. 1–5.

[22] P. Mohan and C. Rishika, "Routing path discovery in wsn based on energy consumption and average delay-centralized & distributed approach," in *2022 IEEE North Karnataka Subsection Flagship Inter- national Conference (NKCon)*. IEEE, 2022, pp. 1–7.

[23] N. Kumar, D. Acharya, and D. Lohani, "An iot enabled vehicular decision fusion framework for accident detection and classification." *International Journal of Next-Generation Computing*, vol. 11, no. 2, 2020.

[24] A. Mukhopadhyay, A. Anoop, S. Manishankar, and S. Harshitha, "Network performance testing: a multi scenario contemplate," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. IEEE, 2020, pp. 1–7.

[25] A. A. Acharya, K. Arpitha, and B. S. Kumar, "An intrusion detection system against udp flood attack and ping of death attack (ddos) inmanet," *International Journal of Engineering and Technology (IJET)*, vol. 8, no. 2, 2016.

[26] D. S. Nair and S. K. BJ, "Identifying rank attacks and alert application in wsn," in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*. IEEE, 2021, pp. 798–802.