ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
**Indian Journal of Science and Technology,** *Vol 11(22), DOI: 10.17485/ijst/2018/v11i22/92337, June 2018*

# ARP Poisoning Detection and Prevention Mechanism using Voting and ICMP Packets

**Sweta Singh* and Dayashankar Singh**

Computer Science and Engineering Department, Madan Mohan Malaviya University of Technology, Gorakhpur – 273016, Uttar Pradesh, India; swetasss22691@gmail.com, dss_mec@yahoo.co.in

## Abstract

To tap into the communication, modify traffic, and stop the network traffic is always the intension of an attacker. ARP poisoning is one of the simplest ways to accomplish these malicious intensions of the attacker. **Objective:** For detection and prevention of such attempt, a concept of voting and ICMP echo requests has been introduced to verify the binding and defend these malicious intensions of the attacker. **Methods:** A voting is done to validate the binding from the other hosts of the LAN such that if attacker pretends to be a new host, it can easily be detected. In case of mismatch of IP or MAC, ICMP echo packets have been used. **Findings:** The validation performed by each host has made the scheme free from being centralized and even do not demand any incompatibility or modification in the existing protocol model. Implementation is conducted on Ubuntu using raw socket coding in python and scapy. ICMP packets are created and spoofing is conducted. Fake ARP packet is sent using packEth and the incoming and outgoing of packets between the hosts is analyzed using Wireshark. **Improvements/ Application:** New host entering the network is also validated in this scheme. The scheme can effectively mitigate LAN attacks.

**Keywords:** Address Resolution Protocol (ARP), ARP Poisoning, ICMP (Internet Control Message Protocol), LAN attacks, Voting

## 1. Introduction

Address Resolution Protocol (ARP) identified under RFC 826[1–3] is a data link layer protocol to bind the IP or network address of the host to its corresponding data link or MAC address. In the OSI model it operates as an interface protocol between the Network (layer 3) and Data Link layer, where it accepts the IP of a host from network layer and resolves its MAC, which is understood by the data link layer. For communication over a LAN, both IP and MAC address are equally important. The host can only send the packet to the destination only if it knows the destination host's MAC. MAC address is responsible in dropping the packet at the correct site or host. If the host intends to communicate to another host but do not know the MAC of destination host, broadcasts an ARP request with IP of the destination host to all the hosts in the LAN. On receiving the Request only the intended host whose IP matches to the IP in destination field of ARP Request sends an ARP Reply, providing its MAC. Thus the <IP, MAC> binding resolution requires both an ARP Request and Reply.

This binding is maintained in the ARP cache at the host's machine to prevent sending of ARP messages each time to attain the binding and minimize the network traffic. This binding is held in ARP cache of the host for a particular time period (approx 20 minutes) and on timeout is expired and removed[2,3].

ARP operates well under normal environment but suffers from two vulnerabilities; one is its stateless nature and other is its un-authenticated nature. The unauthenticated nature of ARP causes the host to accept any ARP reply it receives with no concern either it is coming from a genuine entity or not. Whereas the ARP's stateless nature enables the host to accept any unsolicited reply, whether the request for it was made or not. These two drawbacks provide the attacker an easy way to send a fake ARP reply, spoofing its identity i.e. attaching its MAC with a legitimate host IP and send it over LAN pretending to be that host. On receiving this forget spurious reply the host will update its cache with the false binding resulting in diversion of network traffic to the attacker instead of the intended host[4,5]. With poisoning of ARP cache

**Author for correspondence*

the attacker can successfully carry out other attacks such as MITM attack, Dos attack, Host Impersonation, Http Sniffing, Port Scanning, etc.[3,5].

The organization of remaining part of the paper is such as: Section 2 gives a background study of ARP protocol, ARP cache poisoning and the possible attacks due to ARP poisoning. Section 3, discusses related works and the scheme so far to detect and prevent ARP poisoning attack. The proposed scheme is discussed in Section 4 with the implementation and result are provided in Section 5. Section 6 finally concludes the paper with some glimpse of future work that can be attempted.

## 2. Research Background

### 2.1 Address Resolution Protocol (ARP)

Address Resolution Protocol is responsible to map the IP (network address) into its MAC (hardware address). To send an IP datagram to another node over a LAN; the source should possess both IP of destination to route the packet and the MAC. Only holding IP of the host is not enough for data communication as it can only route the packet to correct network but also needs Data Link address (MAC). Thus to learn the MAC of the destination ARP is used Figure 1 represents the resolution of IP into its corresponding MAC in the network layer of OSI model.

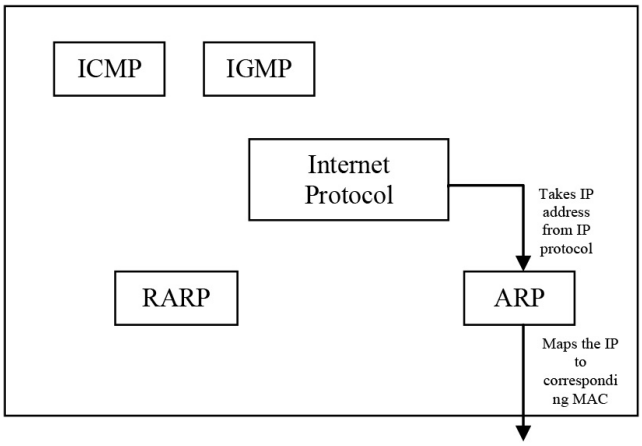


**Figure 1. Resolution of IP into its corresponding MAC.**

To learn the MAC; the host first checks its cache and if found then starts the communication else broadcasts an ARP Request over the Local Area Network (LAN). This request consists of IP and MAC of the source with destination IP address. On receiving the Request the intended node whose IP matches to the IP in the destination field of ARP Request sends a unicast Reply to the sending host while the other hosts drops the packet. This ARP Reply carries the MAC of the destination host. On receipt of ARP Reply the sending host caches the reply into its local primary ARP table to avoid this sending and receiving of ARP request for resolution each time. The entry containing IP and MAC of the host persists in memory for 20 minutes approx. and then is automatically removed. The ARP cache entry can be both static and dynamic[6,7]. Figure 2 represents the mechanism of ARP Request and Reply.

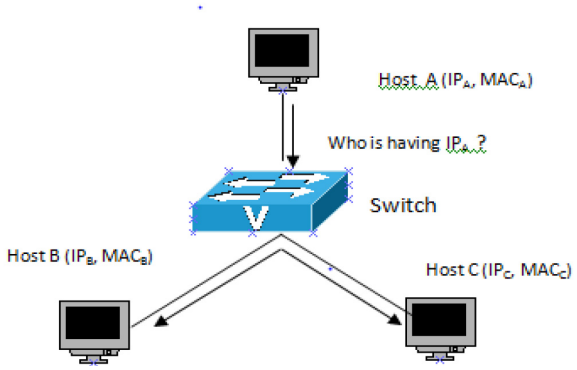


**Figure 2. ARP request and reply.**

### 2.2. ARP Cache Poisoning

The unauthenticated and stateless nature of ARP has given an opportunity to the attackers to introduce a spurious <IP, MAC> binding; attaching legitimate host IP with its MAC in another hosts cache. By performing this malicious act, the hosts cache is poisoned with a false <IP, MAC> association; resulting in diversion of network traffic to attackers site[5,8]. Even the host can accept the ARP Reply from a host irrespective of whether it made an ARP Request for it or not. This attempt of poisoning can me made either by sending unsolicited Reply, or an ARP Request to gather host's information to misuse it later, sending a fake Reply to a genuine Request, or sending a spurious Request and Reply both [9,10]. The attempt of poisoning is represented in Figure 3.

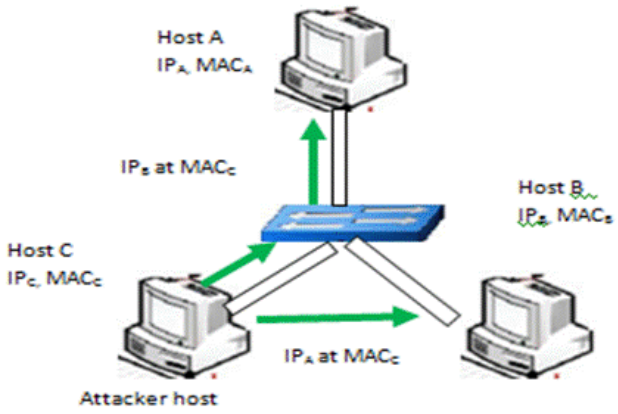


**Figure 3.** **ARP poisoning causing MITM attack.**

## 2.3 Possible Attacks due to ARP Poisoning

The attacker once poisoned the cache of host by attacking its MAC to a legitimate entity IP, can effectively perform several threats; such as posing to be a genuine host and sitting in the network between two hosts playing role of MITM, controlling over a communication session, DoS attack, HTTP Sniffing, Bombing packet attack, etc . Some are discussed as under[6]:

- Man-in-the-Middle attack (MITM): Suppose there are two hosts in the network targeted by the attacker Host A with <$IP_A$, $MAC_A$>and Host B<$IP_B$, $MAC_B$>. The attacker with $MAC_{ATCK}$ sends host A pretending to host B saying "$IP_B$ is at $MAC_{ATCK}$" and similarly sends to host B saying "$IP_A$ is at $MAC_{ATCK}$". Now both host A and Host B cache are poisoned with fake binding and every packet A sends to B or B sends to A, reaches the attacker who is sitting between the two. To hide its existence, the attacker can enable packet forwarding.
- Denial of Service attack (DoS): Once attained the MITM position; the attacker can easily block the services to the host or can block the communication. The attackers can even cause unavailability of resources to its intended victims.
- Host Impersonation: The attacker apart from sniffing the network can also send packets on behalf of the victim to the other victim or other.
- Packet Bombing: Sending of a large number of packets onto a victim's machine with an intention to overload its resources or crash the system is termed as packet bombing. It leads to buffer overflow, overloading and failure of the system.

## 2.4 Related Work

A number of schemes have been discussed so far by the researchers to provide an effective solution to ARP Poisoning. They proposed an idea of making static <IP, MAC> [2]entries into the cache such that only the packets received from those hosts whose binding is maintained in cache is accepted. But this solution was only feasible in a small network raising an overhead upon the operating system to maintain these bindings.

In[3] have made use of shell scripts performing continuous monitoring of binding. To hold the MAC address of gateway preventing it from being poisoned Arping request is sent at regular interval. The commands used were "ip neighbour show" and "arp –a". But this solution required the gateway <IP, MAC> to be known in advance, with only working on Linux platform.

Cryptography-based scheme included Secure ARP (SARP)[11] and Ticket ARP (TARP)[12]. In SARP concept of private/public key with digital signature was implemented with Secure-DHCP. Each host was authenticated with a private/public key certificate issued by AKD (Authoritative Key Distributor). TARP on the other hand uses a ticket which served as digital attestation issued by LTA (Local Ticket Agent) to authenticate the host. Both these schemes required a large number of packets with single-site failure issue.

Antidote Scheme aimed at checking the aliveness of the host i.e. if a binding is found with a mismatch then the previous MAC is checked for aliveness and if found then new binding is not updated in cache. In[13] have made use of a centralized technique in which a central server validates the binding. It also made use of probing scheme to check the aliveness of host. In[14] proposed a scheme in which two queue were maintained a Request_Q and Response_Q. When a Request is sent; the target host's IP is held in Request_Q. On receiving the response, it is checked whether its request is outstanding. If yes it is then transferred to Response_Q. If there is no entry of the reply in Request_Q, Response_Q is checked for that IP. If found and the binding is found to be same the packet is dropped else for case of mismatch or binding not found in Response_Q; the packet is dropped.

Passive detection tool[2] such as AntiARP, ARPGuard, ARPWatch, Antidote[15] etc. but these schemes were dependent on the arrival time of attacker and were only able to detect attack and do not prevent it. The other issue with this approach was that it raises an alarm and by the time administrator could take action, the attacker can carry out mischief. This software was successful in detecting the attack but by the time it warned the administrator for defence action the attacker could succeed in its attempt.

In[7] made use of ICMP protocol to validate the binding and check the aliveness of the host by sending ICMP ping packets to previous binding. But being centralized as it maintained a database at detection host thus faced an issue of single site failure. It introduced the usage of two ICMP probe packets depending upon the type of attacker. Based on the response the mapping was validated. Here the probe packets were sent to the new binding. In[1] proposed a concept by usage of a secondary cache and ICMP ping packet to check the aliveness of previous host. They used an entering and existing algorithm to validate the binding. The scheme provided an effective solution against IP exhaustion problem.

In[16] coined a mechanism in which a central server was selected using voting and were responsible of sniffing each reply to a host and whenever a mismatch was found; validates it using ICMP packet.

## 3. The Proposed Solution

The proposed scheme makes an attempt to provide solution for ARP poisoning by overcoming the drawbacks of ARP. The solution incurs no change in the existing protocol infrastructure with distributed nature. No additional software is required or there exists any complexity issue as in Cryptographic scheme. The scheme is implemented in a comparatively smaller space to propose its mechanism.

Here, in the scheme a concept of voting has been introduced where if the attacker uses the IP of a host unavailable in the network, is being enquired by the neighbouring hosts, i.e. if the hold the binding received in the ARP reply, if 50% of the hosts polls to hold the same mapping, the binding is accepted else goes for ICMP validation. ICMP probe packets are also adopted for any mismatch found in the secondary table pre-maintained by the hosts.

The proposed scheme was derived from the schemes of[1,4,16] use of a secondary table to hold the <IP, MAC> binding the host. This entry can be stored in form of a text file so that it persists for longer time period. A voting mechanism is adopted with an assumption, if the entry is not found in primary as well as secondary cache; there might be a possibility that the entry is maintained at some other host in the network. The host broadcasts a voting request sending the IP received in ARP reply to attain the MAC and waits for time interval of 0-100 msec. The reply when received is responded by immediately without any delay. If the reply received has more than 50% (0.5N; N= no. of nodes) for that binding received; the binding is accepted. In case no host has the binding or, the reply received is <50%; an ICMP echo request is sent to validate the binding. If reply is received the binding is accepted else is discarded. After validation updation is made accordingly in primary and secondary table.

For any case of mismatch in secondary table, whether be IP or MAC is found in secondary table; an ICMP probe packet is sent to both new as well old binding. On the basis of reply received, the entry is accepted or discarded.

The scheme works as follows: There could be two possible scenarios i.e. either the host wants to send an ARP Request/Reply or it has received an ARP Request/ Reply. If it wants to make an ARP Request simply broadcast it to all the hosts in the network. Else if it wishes to send an ARP Reply places it's MAC and routes it in unicast mode to the host which made the Request for it.

In case of receipt of packet, if the received ARP packet is an ARP Request, verifies if the packet destination matches to its IP. If the Request packet is intended to it, send a Reply providing its MAC address. Else if not the packet is dropped. For the received packet an ARP Reply can give rise to certain cases represented in Figure 4. It checks the primary cache for the <IP, MAC> association; if found the entry is updated. For case the entry is not present or there is any mismatch of IP or MAC; the entry is searches in Secondary table. If the binding is obtained in secondary table and is same as received in ARP Reply, update the primary cache with the binding. But for case that the entry is not found in secondary table or there is case of mismatch it enters the Validation phase.

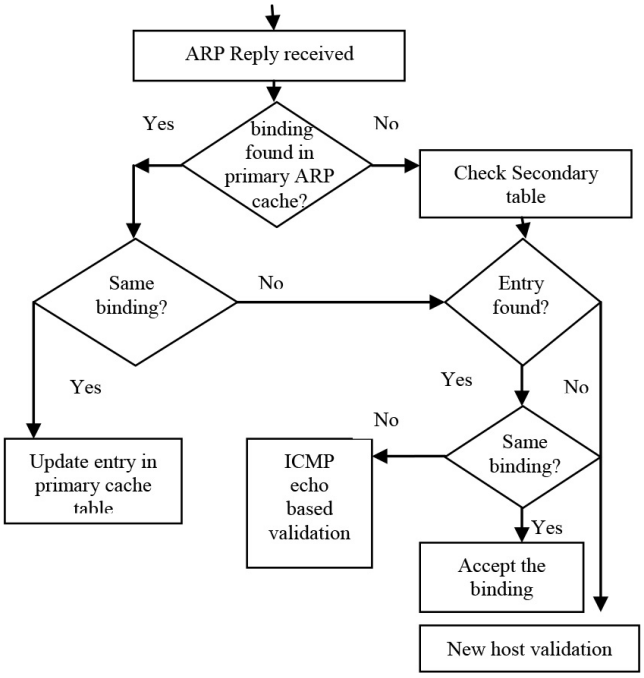


**Figure 4.** **Certain cases in case of received ARP reply packet.**

If the binding is found to be different from the previous one, an ICMP echo packet is sent for validation to both the previous and the new binding. If reply is attained from both host means there is a poisoning attempt hence the new entry is blocked and packet is dropped and its entry is removed from primary cache. If only the previous host sends a response mean the new binding is fake and hence is dropped and removed from primary cache. For case that new binding replies to echo request and not the old host, means the IP is provided to some other host (in case of MAC mismatch) or host has been provided a new IP (in case of IP mismatch). Accordingly, the secondary table is updated with the new received binding. There could be a possibility that none of the host response back, for such case mean that the previous host has got offline and new host is trying to log on ARP poisoning thus the packet is dropped and removed the entry from primary table.

If the binding is neither found in primary table i.e. expired nor available in secondary table, shows the chance that the host is new entry entering the network. There is no guarantee that the new host is a genuine host or a malicious host using the IP of a legitimate entity. So, to validate the new binding voting request is initiated by host to all its neighbour hosts in the network, with an assumption that some other host may hold the binding in its cache. The response is wait for particular time period and evaluated. If more than 50% hosts poll to have the binding, the new host is accepted and its binding is updated in primary and stored in secondary table as well. Else if the voting polls (attained by calculating 0.5*n, where n is the number of neighbour hosts in the network) is less than 50%, the binding may be spurious or genuine such that the host is new for all the others in the network. Thus an ICMP ping packets are used to verify the genuine of the binding. An ICMP echo request is sent to the host. If reply is received, then the binding is accepted to be true else if no reply received then means is a falsified packet. In case of no reply, the packet is dropped and entry is removed from primary table. Thus this scheme involves the other hosts in the LAN in the validation of the Reply, defending poisoning attempt of the malicious hosts.

The procedure is discussed as under:

If a frame has been received
- If it is a request:
  - If the packet is intended to it: accept the packet and generate the reply
  - If the packet is designated to some other host: drop the packet
- If it is a reply:
  - If the binding in Primary cache: update cache and drop the reply packet
  - Else if binding not found in primary cache or there is any case of mismatch:
  - Check the secondary table:
    - If the binding is found: update the primary cache with the entry and drop the packet
    - Else if binding not found in secondary cache:
      - Broadcast voting request to all the hosts to find if some other host holds that binding in the network:
      - If reply>0.5*no. Of nodes in the network:
      - Accept the binding
    - Else send ICMP probe packet to the binding: reply received accept the binding and update both primary and secondary cache
    - If ICMP reply not received drop reply and remove entry from primary cache.
    - Else if there is mismatch: Send ICMP ping packet to both new as well as old binding
    - If reply received from only old binding: drop the reply packet and remove its entry from primary cache

Else If the reply received from both older and new binding: drop the packet and remove its entry from primary cache

Else if no reply received from both binding: drop reply and remove entry from primary as well secondary table

Else If reply received from only new binding mean the previous host has been gone offline. Accept the new binding update the primary and secondary table with new entry.

Else a frame is to be sent:

If it is a request:

Broadcast the ARP request to all

Else if it is a reply:

Send a unicast reply to the sending host providing its MAC

In Figure 5, represents the flow chart of the proposed scheme to provide a clear view of the scheme.

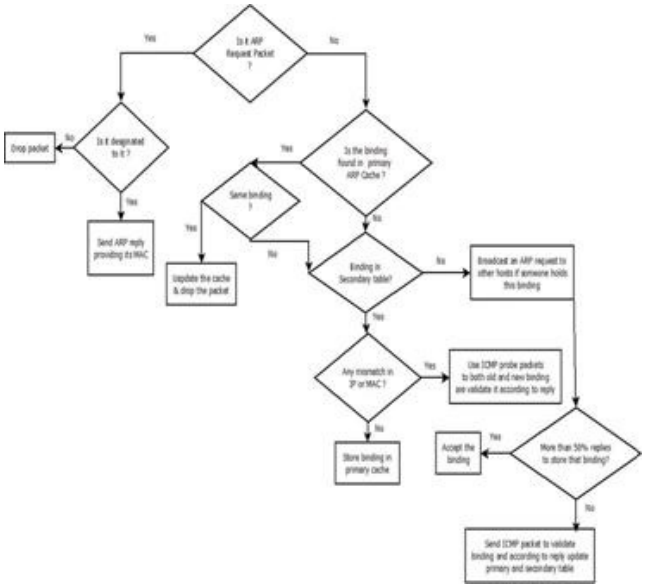

**Figure 5.** **Flowchart of the proposed mechanism.**

## 4. Implementation

The implementation is carried out on Ubantu operating system with use of certain tools which include Wireshark, packEth and Ettercap. Wireshark is to analyse the network traffic, packEth to generate ARP request/reply packet, and scapy i.e. a packet manipulation tool in python. Coding is done to implement the scheme. It involved use of 3 hosts with IP and MAC as in Figure 6. To generate false ARP packet PackEth tool is used and sent over the network, which is captured by "sniff" function of scapy. The coding is done in python using scapy. Packets are sent over the network using "sendp". For alarm generation festival tool is used where, when the attack is detected or ICMP echo request is not answered a file is created and the message is printed which is read and converted to speech using festival tool package of scapy. The unanswered and unanswered packets are calculated in order to attain the voting response in case the entry is new. The secondary table is in form of text file stored and read each time. "sniff" function only reads the ARP packet using the "filter" parameter.

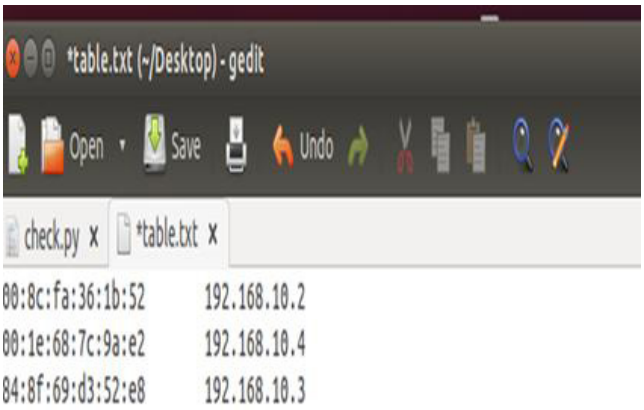


**Figure 6.** **Secondary table holding <IP, MAC> of the host.**

The operation is such that when the host send a fake ARP reply packet using packet tool, the binding is checked in secondary table. If the entry is found and there is no mismatch is updated but in case a mismatch of IP or MAC, ping request is send using "srploop" command one designated to new host and other to previous host. In case of IP mismatch reads and prints both the IP's one in ARP reply receives and other held in secondary table and generates an alarm.

Thus, the scheme successfully prevents and detects poisoning attempt and thus defending the host from other possible attacks too. The voting parameter is used to validate the genuinity of new host and not in choosing the central detection server used previously by certain researchers.

Here the attacker possess the IP 192.168.10.2 and pretends to be holding IP 192.168.10.4 (victim) by attaching its MAC to the victim as shown in Figure 7.

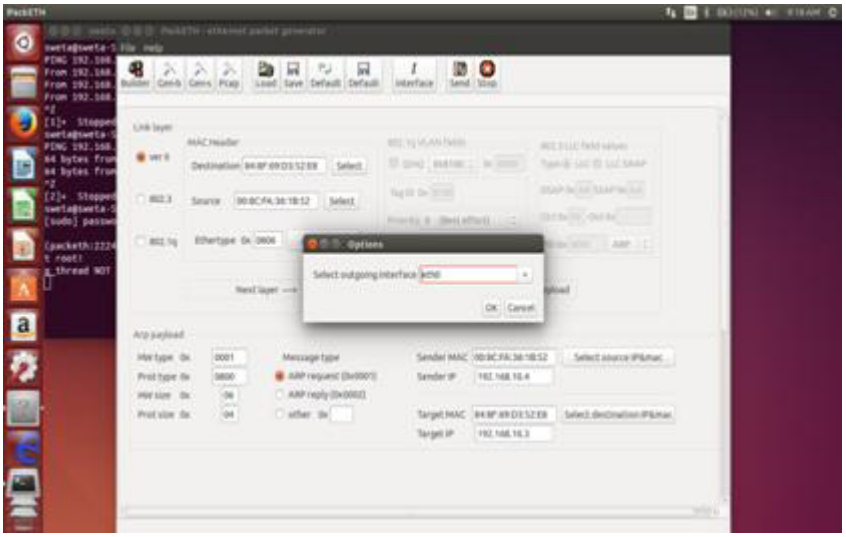

**Figure 7.** **Attacker binding its MAC with IP 192.168.10.4 and sending in ARP reply to 192.168.10.3.**

While implementation ARP table was checked, Wireshark was analysed periodically to identify the packets. "sendp" command was used in scapy to send ARP Request packet "who-has" for IP address "192.168.10.4" and at the same time fake binding was created using packEth tool and sent over the network. Thus the fake binding can be seen in Wireshark. Then Request was sent for the original IP "192.168.10.2", since already found in network produce ARP reply providing its MAC. The captured packet is shown in Figure 8.

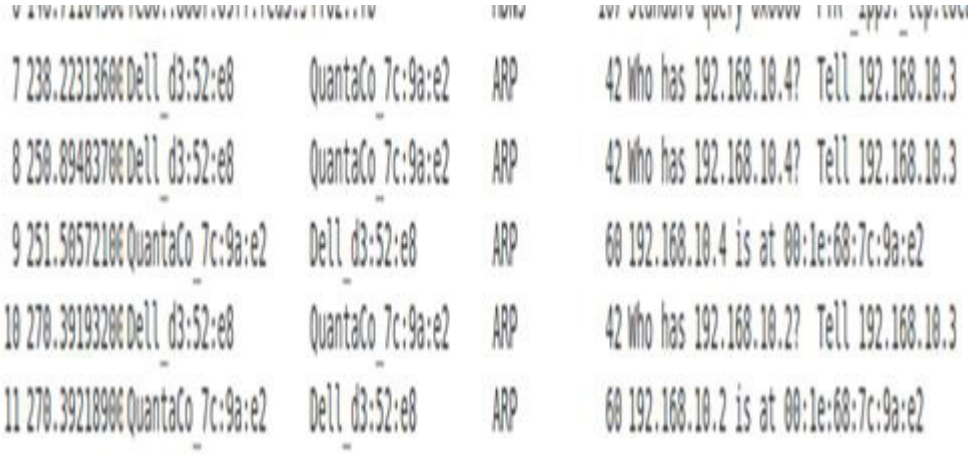


**Figure 8.** **Wireshark captured packet showing the poisoning attack.**

Since the experiment is conducted on a small scale, the attacker's IP is stored previously in secondary table, which later uses packEth tool to send fake binding. But the host has its previous binding, thus when code is run; its IP is found in secondary table with mismatch IP; hence an ICMP probe packet is sent to its previous binding. Since the attacker possess the binding and is in network replies to it. Thus, is detected and alarm is raised as shown in Figure 9.


**Figure 9.** **Detection using ICMP probe packet.**

A comparative discussion has been presented in Table 1, to define how our scheme operates, in comparison to previous schemes using voting mechanism. Remark is provided to define the areas focussed and issues encountered. The remark could help future researchers to cover the area and produce an effective proposal for its solution.

**Table 1.** Comparative analysis of voting procedure

| Scheme Proposed | Voting adopted | Centralized or Distributed | Remark |
|---|---|---|---|
| In[16] | For choosing a centralized server which validates the ARP Reply received | Centralized | Incurred single site failure where central server validated the binding using probing mechanism |
| In[4] | To validate new binding without use of ICMP probe packet | Distributed | Used voting for new host validation but accepted the binding for both cases whether received 50% positive polls or not. |
| In[8] | Voting used for validation of new binding together with ICMP probe packets for mismatch case.<br>If 50% positive polls achieved for new binding accepted and updated in table else sent ICMP packets for validation | Distributed | There is possibility that attacker can poison a number of hosts cache, thus causing the neighbour host too to hold fake binding. Thus they can poll for fake binding. |

## 5. Conclusion and Future Scope

Here with this mechanism, the attackerscattempt to poison the cache is failed by checking the entry in secondary table and for the case of mismatch in IP-MAC binding, ICMP echo packet is used. If an attacker tries to mask itself by making use of an IP of a host who is not present in the network, its attempt is failed too, as the binding information is sent to all other hosts in the network to validate the binding. This validation is made by checking if some other host in the network holds this binding information or not. The voting mechanism involves the other hosts in the network to find if they possess the binding. If the binding is obtained, then neighbour hosts polls for it. The number of response is calculated to find the votes. The experimental analysis is carried out on a smaller scale and is found to run well. Considering the implementation on a wider scale, will require large number of hosts and experts to present the complete picture. The manipulation will be conducted accordingly to the present the desired workflow. The concentration is on further expanding the work by bringing about some modifications at few sites such as introducing certain parameters such as response time while receiving the response. It could help in understanding the behavior of legitimate and malicious host. Other improvements could call upon introducing a hybrid model or using 2 ICMP packets to provide defense to all types of attacker discussed by author of[5]. The prime focus would be in finding all the areas which could be attacked and then keeping them in mind while defining a new solution or improving the existing solutions.

## 6. References


1. Tripathi N, Mehtre BM. An ICMP based secondary cache approach for the detection and prevention of ARP poisoning. Proceedings of International Conference on Computational Intelligence and Computing Research (ICCIC), India; 2013 Dec 26. p. 1–6. Crossref.
2. Tripathi N, Mehtre BM. Analysis of various ARP poisoning mitigation techniques: A comparison. Proceedings of International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT); 2014 Jul 10. p. 125–32. Crossref.
3. Nayak GN, Samaddar SG. Different flavours of man-in-the-middle attack, consequences and feasible solutions. Proceedings of International Conference on Computer Science and Information Technology (ICCSIT). 2010 Jul 9; 5:491–5. Crossref.
4. Nam SY, Kim D, Kim J. Enhanced ARP: preventing ARP poisoning-based man-in-the-middle attacks. IEEE Communications Letters. 2010 Feb 1; 2:187–9. Crossref.
5. Pandey P. Prevention of ARP spoofing: A probe packet based technique. Proceedings of International Advance Computing Conference (IACC); 2013 Feb 22. p. 147–53. Crossref.
6. Abad CL, Bonilla RI. An analysis on the schemes for detecting and preventing ARP cache poisoning attacks. Distributed Computing Systems Workshops, 2007. ICDCSW'07; 2007 Jun 22. p. 60–60.
7. Salim H, Li Z, Tu H, Guo Z. Preventing ARP spoofing attacks through gratuitous decision packet. Proceedings of International Symposium on Distributed Computing and Applications to Business, Engineering & Science (DCABES); 2012 Oct 19. p. 295–300. Crossref.
8. Jinhua G, Kejian X. ARP spoofing detection algorithm using ICMP protocol. Proceedings of International Conference on Computer Communication and Informatics (ICCCI); 2013 Jan 4. p. 1–6. Crossref.
9. Arote P, Arya KV. Detection and prevention against ARP poisoning attack using modified ICMP and voting. Proceedings of International Conference on Computational Intelligence and Networks (CINE); 2015 Jan 12. p. 136–41. Crossref.
10. Goyal V, Tripathy R. An efficient solution to the ARP cache poisoning problem. Proceedings of Australasian Conference on Information Security and Privacy, Springer Berlin Heidelberg; 2005 Jul 4. p. 40–51. Crossref.
11. Lootah W, Enck W, McDaniel P. TARP: Ticket-based address resolution protocol. Computer Networks. 2007 Oct 24; 15:4322–37. Crossref.
12. Kumar S, Tapaswi S. A centralized detection and prevention technique against ARP poisoning. Proceedings of International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec); 2012 Jun 26. p. 259–64. Crossref.
13. Wang Z, Zhou Y. Monitoring ARP attack using responding time and state ARP cache. Proceedings of The Sixth International Symposium on Neural Networks (ISNN), Springer Berlin Heidelberg; 2009. p. 701–9. Crossref.
14. Tripunitara MV, Dutta P. A middleware approach to asynchronous and backward compatible detection and prevention of ARP cache poisoning. Proceedings of Computer Security Applications Conference(ACSAC'99); 1999. p. 303–9. Crossref.
15. Antidote [Internet]. [cited 2015 Sep 17]. Available from: http://online.securityfocus.com/archive/1/299929.
16. Bruschi D, Ornaghi A, Rosti E. S-ARP: a secure address resolution protocol. Proceedings of Computer Security Applications Conference; 2003 Dec 8. p. 66–74. PMCid:PMC1298960