ARP Cache Rectification for Defending Spoofing and Poisoning Attacks

Alok Pandey
Senior Systems Manager,
Birla Institute Of Technology, Mesra
Jaipur Campus, Rajasthan, India
Email Id: alokpandey1965@yahoo.co.in

Jatinderkumar R. Saini
Professor & I/C Director,
Narmada College of Computer Application
Bharuch, Gujarat, India
Email Id: saini_expert@yahoo.com

***Abstract* – Securing LAN plays an important role in overall network security as it actually involves guarding it from both the external and internal attacks. The internal users can also launch several types of attacks by exploiting the weaknesses of the TCP/IP protocol suite. Address Resolution Protocol (ARP) is an essential component in LAN communications as it helps in resolving the IP Address into MAC Address. By exploiting some of the in-built vulnerabilities of ARP, an adversary can launch different types of DoS and MITM attacks for accessing confidential data or launching other attacks. We present a comprehensive mechanism that checks and rectifies ARP cache of compromised system on the LAN and also guards against some of the commonly used methods for launching of DoS and MITM based attacks in LAN.**

Keywords – ARP Spoofing, Denial-of-Service (DoS), LAN, Man-In-The-Middle (MITM), Security

I. INTRODUCTION & RELATED WORKS

Corporate and large scale organizations utilize ICT for building organisation wide networks for sharing and dissemination of information, integration of enterprise wide business applications, supply chain management etc. Such networks are mostly TCP/IP based and often use Internet, intranet and extranet technologies for running client / server based business applications which run in LAN and WAN environments. These internal company networks are separated from other networks by firewalls, for preventing unauthorized access to company's internal data and leakage of sensitive information. However, for exchanging mails and other related information with other organizations, Internet connection may be required.

These applications normally rely upon variety of other protocols and services such as DHCP, DNS, HTTP, SMTP, FTP etc. as may be required in Local Area Network. LANs often work on IPv4 using 32 bit addressing and are Ethernet based which uses 48 bit addressing. Thus some mechanism for resolving the IP addresses into MAC address is required. ARP is widely used protocol for providing such mappings. It is also used by interconnecting devices to find out the MAC address of connected host whose IP Address matches with the IP address of an incoming packet [1] for a device connected to its LAN.
With the increased usage of network based applications, the numbers of unauthorised access and data breaches have also increased with motives like financial gains or publicity or to damage the victim's reputation.

In a typical LAN environment, internal user can launch different types of network attacks based upon sniffing, spoofing techniques and capture sensitive information like user name, passwords, IP addresses, port numbers, and other proprietary data [2] and use it for penetrating further into network for thefts and damages to data. This underlines the urgent need for reliable techniques for detection of sniffing and spoofing based attacks caused by internal users in LANs.
Capturing and analyzing a TCP/IP packet on a network [3] for stealing network based information is called Sniffing [4]. Another well known technique for launching attacks in network environments is spoofing [5].

Most of the programming languages support raw socket programming feature which permits crafting & injecting packets in the network. Attackers craft bogus packets for gaining access to networks and try to capture real time data. The process of creating and injecting fake TCP/IP packets with some one-else's identifications on networks is called as Spoofing [6]. Another version of this type of attack is called as MITM attack, where entire session is hijacked to steal data.

The concept of spoofing is used to launch different types of attacks on the TCP /IP based applications and services. Protocols like IP and ARP are exploited for launching attacks like Port Scanning, ARP Cache Poisoning, Changing of Default gateway, ICMP redirect, DHCP poisoning, DNS poisoning etc. Based upon IP spoofing, which involves forging of IP addresses of the source device, different types of attacks can be launched [7] whereas attacks like DoS and MITM can be achieved using ARP spoofing.

Address Resolution Protocol (ARP) is used for finding out the MAC address of the destination device on a LAN [8]. ARP stores such mappings of IP addresses to MAC addresses in temporary storage called cache for future usage [9]. This cache is updated from time to time. Whenever the system has to transmit a frame it first checks its ARP cache for locating the corresponding MAC address of the receiver [10]. It uses two types of messages namely ARP request and ARP reply which are encapsulated inside an Ethernet frame. It contains

978-9-3805-4421-2/16/$31.00 ©2016 IEEE

MAC addresses of sending and receiving devices along with a value of 0x0806 in Ethernet type [11]. The frame also contains the IP and MAC addresses of the sender and receiver along with an operation code as part of the ARP message.

The entries to the ARP cache can be added either statically or dynamically [12]. For supporting the DHCP enabled hosts, these entries are removed periodically form the cache. The devices update their ARP cache whenever they receive an ARP Reply even if they had not sent out the corresponding ARP request earlier as ARP is stateless protocol [13,14]. Thus, despite its crucial importance ARP provides ground for launching ARP spoofing & ARP cache poisoning attacks [12].

For genuine communication both Ethernet and ARP headers should match. But since there is no mechanism to check consistency of these headers, attackers intentionally craft packets having different or forged values of IP-MAC addresses [15,16,17,18]. This is called ARP Cache Poisoning.

Thus aattacker modifies the entry for gateway or any other genuine host with mapping of their IP Addresses and its MAC address in ARP Cache of victim system. After this a variety of attacks can be launched [14, 19] namely Denial of Service (DoS) attacks, Man in the Middle (MITM) attacks etc. The attackers craft different types of packets based upon IP, ICMP, TCP, UDP etc protocols and try to disrupt various functionalities of the network.

Although solutions based upon Static ARP Cache entries to prevent ARP spoofing attacks exist yet they have some major issues like effort required for manual configuration of static entries, limited scalability and workability in static and DHCP based networks [14].

Some of the typical works done in this category include the DAPS (Dynamic ARP spoof Protection System) technique suggested in [20] which is a solution to ARP spoofing that snoops DHCP packets. Katkar et al. [21] have proposed a light weight approach for prevention & detection of ARP Spoofing. A server based solution has been proposed by Ortega et .al. [22]. Another mechanism to prevent ARP spoofing based upon the use of static ARP entries was suggested by Ai-Zeng Qian [23]. A combination of using static ARP entries and SNORT-IDS is suggested in [24] for resolving the ARP spoofing problem.

II. METHODOLOGY

Our aim is to design a cost effective, easy to implement solution that is compatible with existing ARP protocol and does not require major changes to the network and uses minimum of cryptographic techniques [16] and other costly hardware. It is capable of detecting and blocking spurious ARP communications and repairs the infected hosts, in a switched LAN environment. The proposed solution not only detects, corrects and guards against ARP based problems but also guards against a variety of DoS and MITM based attacks The solution also aims to provide mobility and a consistent working environment to the user as he roams on the different sub networks of the corporate network which might be located at different geographical locations.

The proposed solution is a combination of different sub-routines. The first flowchart shown in Fig. 1 registers and validates the user initially and provides proper username, password, authentication code and location code etc. which are used for logging on the network and getting the IP from the respective DHCP Server for that location [25].

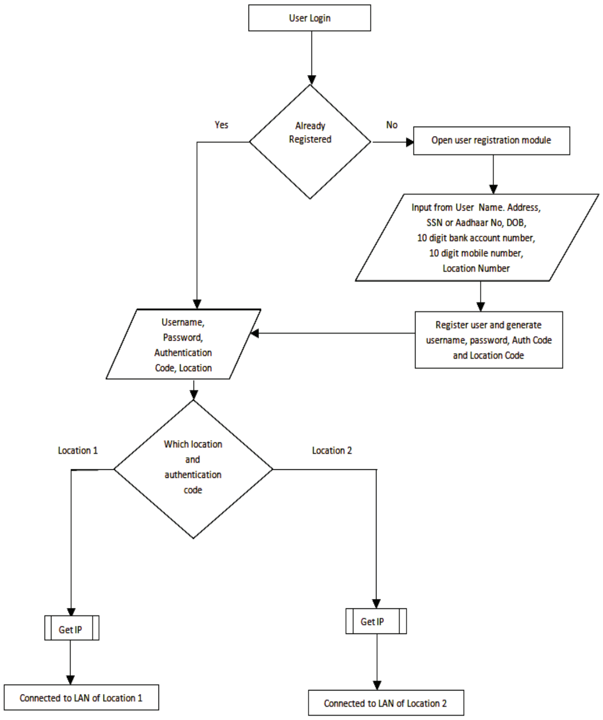
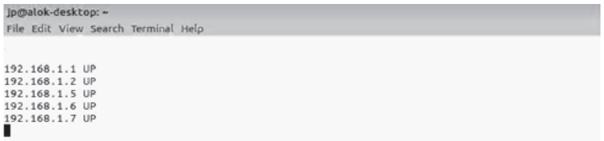

User Login
Already Registered
Yes
No
Open user registration module
Input from User Name. Address, SSN or Aadhaar No, DOB, 10 digit bank account number, 10 digit mobile number, Location Number
Register user and generate username, password, Auth Code and Location Code
Username, Password, Authentication Code, Location
Which location and authentication code
Location 1
Location 2
Get IP
Get IP
Connected to LAN of Location 1
Connected to LAN of Location 2

Fig. 1 Client Side Flow Chart



```
192.168.1.1 UP
192.168.1.2 UP
192.168.1.5 UP
192.168.1.6 UP
192.168.1.7 UP
```

Fig. 2 Client Side Scan for Active Hosts on the LAN

Second flowchart shown in Fig. 3 runs and scans for open ports of the system. It then selectively closes undesired open ports based upon user confirmation. It also tries to discover the neighbors as in Fig. 2 and reports them to the server side of the process. It also flushes and updates the ARP Cache of the client system based upon the authenticated updates as received from the Server side from time to time.

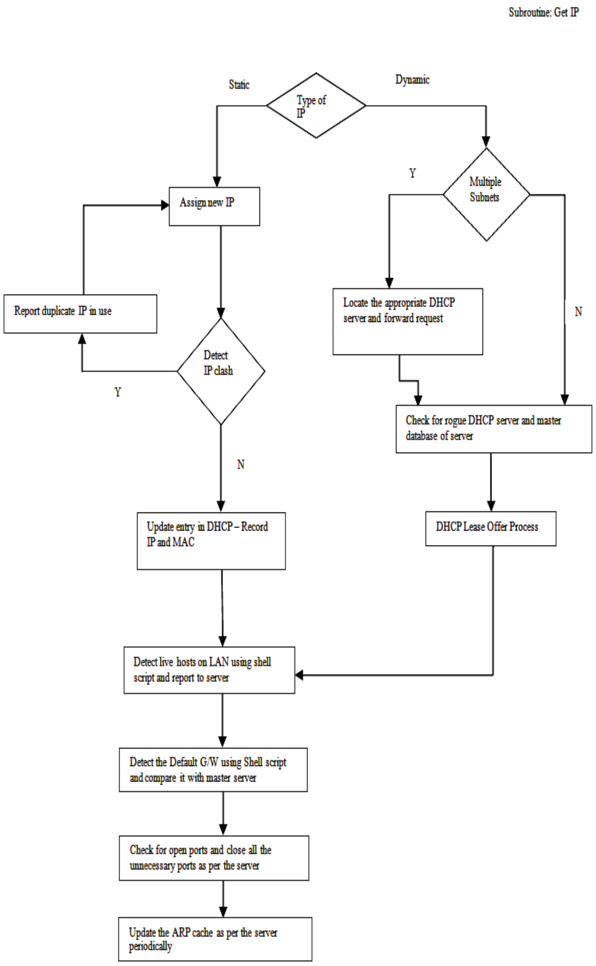

Subroutine: Get IP
Static
Type of IP
Dynamic
Assign new IP
Y
Multiple Subnets
Report duplicate IP in use
Locate the appropriate DHCP server and forward request
N
Detect IP clash
Y
Check for rogue DHCP server and master database of server
N
Update entry in DHCP – Record IP and MAC
DHCP Lease Offer Process
Detect live hosts on LAN using shell script and report to server
Detect the Default G/W using Shell script and compare it with master server
Check for open ports and close all the unnecessary ports as per the server
Update the ARP cache as per the server periodically

Fig. 3 Client Side Flow Chart

The communication between the client and server portions is encrypted using the encryption key which may be generated using government / authenticated individual identifications like driving license, passport number or other related information provided by the user at the time of initial registration.

The flowchart shown in the Fig. 4 runs on the server side and performs the functions of Genuine Host Detection, Cross Layer Verification, Final Node Detection, Updating of Client ARP Cache and Provide Secure Data Exchange.

For detecting genuine MAC and IP Addresses of clients on the network, data regarding the active hosts on the LAN is collected from clients, lease table of the DHCP server and sniffed packets are compared.

Entries found common in all the three are recorded in the genuine host table while the rest of entries are recorded in the suspicious host table.

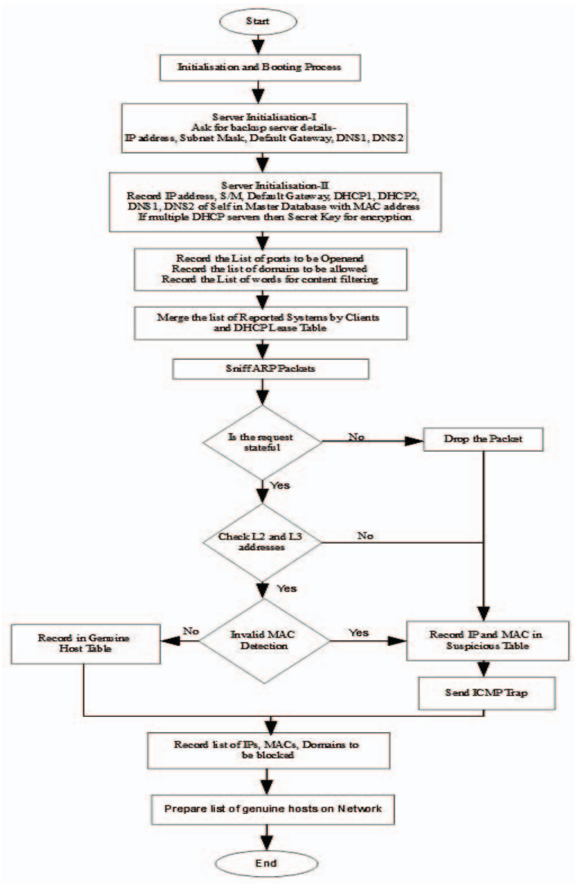
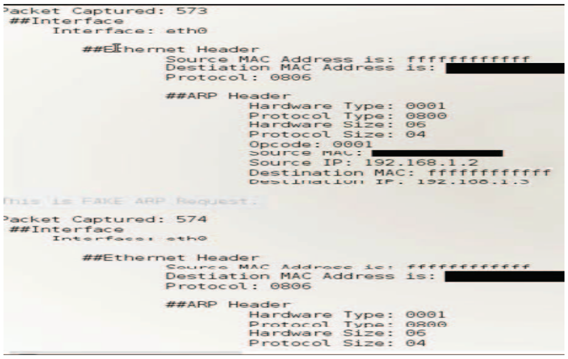

Start
Initialisation and Booting Process
Server Initialisation-I
Ask for backup server details-
IP address, Subnet Mask, Default Gateway, DNS1, DNS2
Server Initialisation-II
Record IP address, S/M, Default Gateway, DHCP1, DHCP2, DNS1, DNS2 of Self in Master Database with MAC address
If multiple DHCP servers then Secret Key for encryption
Record the List of ports to be Openend
Record the list of domains to be allowed
Record the List of words for content filtering
Merge the list of Reported Systems by Clients and DHCP Lease Table
Sniff ARP Packets
Is the request stateful
No
Drop the Packet
Yes
Check L2 and L3 addresses
No
Yes
Record in Genuine Host Table
No
Invalid MAC Detection
Yes
Record IP and MAC in Suspicious Table
Send ICMP Trap
Record list of IPs, MACs, Domains to be blocked
Prepare list of genuine hosts on Network
End

Fig. 4 Server Side Flow Chart

The program also performs cross layer examination and detects invalid MAC Address IP Address combinations by comparing the Ethernet and ARP headers [1,5,19]. If both source and/or destination MAC addresses are not identical as seen in Fig. 5 indicates that ARP spoofing is occurring on the network and such packets should be dropped. Valid MAC address packets are recorded for further processing and physical node detection process.



```
Packet Captured: 573
 ##Interface
     Interface: eth0

        ##Ethernet Header
                   Source MAC Address is: ffffffffffff
                   Destiation MAC Address is: [redacted]
                   Protocol: 0806

                   ##ARP Header
                           Hardware Type: 0001
                           Protocol Type: 0800
                           Hardware Size: 06
                           Protocol Size: 04
                           Opcode: 0001
                           Source MAC: [redacted]
                           Source IP: 192.168.1.2
                           Destination MAC: ffffffffffff
                           Destination IP: 192.168.1.3
This is FAKE ARP Request:
Packet Captured: 574
 ##Interface
     Interface: eth0

        ##Ethernet Header
                   Source MAC Address is: ffffffffffff
                   Destiation MAC Address is: [redacted]
                   Protocol: 0806

                   ##ARP Header
                           Hardware Type: 0001
                           Protocol Type: 0800
                           Hardware Size: 06
                           Protocol Size: 04
```

Fig. 5 Cross Layer Examination

After this it cross verifies the existence of host physically on the network by sending ICMP ping packets to the combination of IP-MAC address as recorded earlier. For the ones where no reply is received TCP SYN packets are sent using the details of the detected IP and MAC address combination. If the host is there on the LAN it will respond back with SYN / ACK or RESET packet [26]. Such entries are recorded in the genuine host table and passed to the clients for updating their ARP Cache [27]. If there is no response then the entry is passed for taking appropriate defence mechanism or raising the alarm on the network.

The proposed solutions also checks and guards against common DoS and MITM attacks that are launched by specifically crafting and injecting abnormal packets based upon TCP, UDP, IP, ICMP etc. Such packets are intentionally injected into the networks by the attackers for locating and further attacking the poorly configured weak points of network like Servers, Workstations, Computers, Firewalls, IDS and other interconnecting devices such as Switches, Routers, Gateways etc. in such a way that the supporting software either malfunction or crashes and further attacks can be launched[28]. Hence it is essential for the system administrators to know about such abnormally / maliciously crafted packets and drop them.

TCP uses a combination of six flags to indicate specific functionality of the current packet and its contents. Each flag is one bit long and has a specific functionality associated with it like some TCP segments carry data while others are used for acknowledgements of the received data. Out of these flags the most popular flags are the "SYN", "ACK" and "FIN", which are used for establishing connections, acknowledging and terminate connections.

A SYN packet is used for initiating a TCP connection whereas an ACK indicates that contents have been received and the device is ready to accept further packets. The 3-way handshake mechanism is based upon these packets and is used to ensure that both the sender and the receiver are ready to communicate before the actual transmission of data is done from either side.

Packets with other flag combination should be treated as suspicious. Attackers use such illegal combination to identify the operating system at the victims system and then exploit some of its known vulnerabilities to further penetrate into the system. Sometimes such illegal combinations may go undetected through firewalls and intrusion detection systems or may crash the victim’s target device.

Some of the commonly seen invalid TCP combinations may include packets with all flags set or no flags set at-all or zero value set in source and/or destination port numbers or in source and destination MAC addresses or setting of both SYN and FIN Flags in the same packet or setting invalid combinations of SYN, FIN, RST, PSH flags in the same packet. [5, 12, 17]

UDP is another Protocol that is available at the transport layer. It is a connectionless protocol with little services. UDP also uses source and destination ports for identifying the sending and receiving processes. Many protocols like DHCP, SNMP, DNS [31] and TFTP use UDP as a transport mechanism [17,29,30]

Attackers use UDP Packets with zero values set in the source and destination port numbers or flood the victim devices by sending multiple UDP packets with same IP address or same port numbers. Similarly abnormal IP packets are also crafted which may have unknown values for protocol type or the packet may be illegally fragmented. [5,12,17]

Likewise, it is possible to generate few types of abnormal ICMP packets which may be used to attack a system.

An ICMP Packet which are fragmented or larger than 65535 bytes or redirects everything to the victim machine should be dropped. If multiple ICMP packets with same Destination IP Address are seen then it is a case of ICMP flooding and such packets should be blocked. [5, 12, 17, 29, 30]

Similarly, based upon the size and the permissible MTU size on the underlying network an IP Packet may be broken down or fragmented into two or more smaller pieces called fragments. [5, 12, 17, 29, 30]

The process of Fragmentation occurs for most of the protocols including TCP, UDP etc. Hence an attacker may craft illegal fragments of the packets which might be illegal or overlapping as a result of which the victim machine either hangs or crashes. [5,12,17,29,30,31]

III. EXPERIMENTAL SETUP & RESULTS

Created a test network as shown in Fig. 6 for realizing different types of attacks. The set up consists of a network of three computers. For demonstrating the concept, a multi-homed system with two LAN cards was used to simulate and provide the functionality of a Router, DHCP Server and testing of the conditions.

On the attacker machine packet crafting software was used for generating the bogus packets and injecting in the network. For verifying the generation and successful injection of the bogus packets Wireshark, a packet capturing and displaying software was used.

The system with two LAN Cards was used for simulating the functionality of a DHCP Server and also a router. Wireshark was used to check and verify the receipt of bogus packets as generated by the attacker system. Different filter conditions were implemented one by one at this system. Screenshots were taken before and after implementing the filter conditions.

At the victim machine Wireshark was run to capture and display the packets that were crafted and sent to the victims system by the attacker.

Different types of ARP, IP, TCP, UDP and ICMP packets were crafted and injected in the network by the attacker system. Several packets with invalid source MAC Address, Destination MAC Address, Source and Destination Port Numbers, illegal flag combinations, were generated and injected in the network. Proper filter conditions to filter out such bogus packets were implemented at the Routing system.

One of the test conditions implemented was that the source MAC address should not be 00:00:00:00:00:00. Fig. 7 shows the generation of crafted packet by attacker. Fig. 8 shows the multi-homed computer (router) where the filter rules were applied. Fig. 9 shows the victim's computer after application of filters. It can be observed that the crafted packets with illegal or invalid parameters by the attacker system are being filtered out and thus the victim system is protected against different types of DoS and MITM attacks.

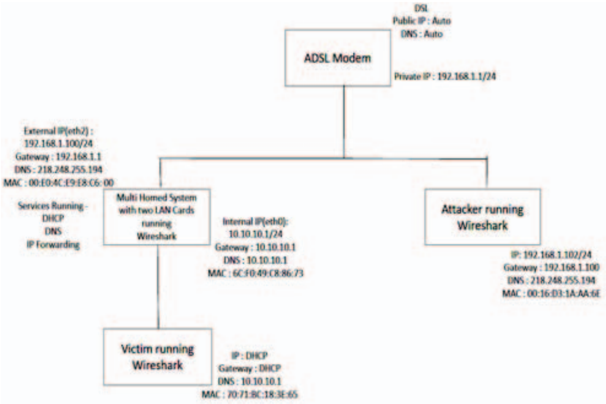
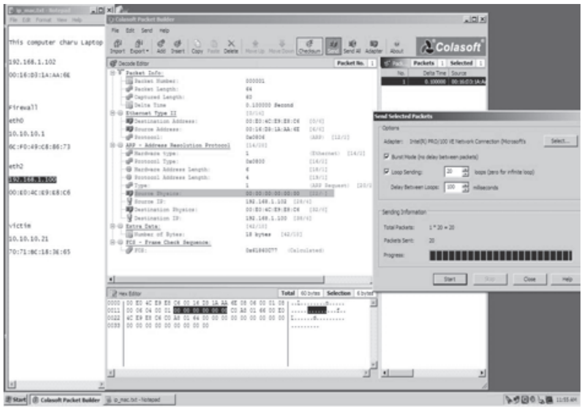


Fig. 6 Experimental Setup


ip_mac.txt - Notepad
File Edit Format View Help
Colasoft Packet Builder
File Edit Send Help
Import Export Add Insert Copy Paste Delete Move Up Move Down Checksum Send Send All Adapter About
Colasoft
Decode Editor
Packet No. 1
Packets 1 Selected 1
No. Delta Time Source
Send Selected Packets
Select...
Start Stop Close Help
Hex Editor
Total 60 bytes Selection 6 bytes
Start Colasoft Packet Builder ip_mac.txt - Notepad
11:55 AM

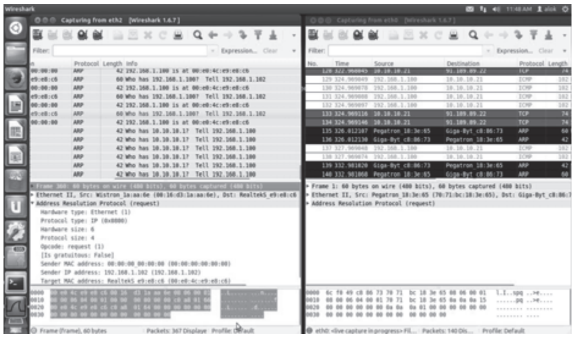
```
This computer charu Laptop
192.168.1.102
00:16:D3:1A:AA:6E

Firewall
eth0
10.10.10.1
6C:F0:49:C8:86:73

eth2
192.168.1.100
00:E0:4C:E9:E8:C6

victim
10.10.10.21
70:71:BC:18:3E:65
```

```
Packet Info:
    Packet Number:                 000001
    Packet Length:                 64
    Captured Length:               60
    Delta Time                     0.100000 Second
Ethernet Type II                   [0/14]
    Destination Address:           00:E0:4C:E9:E8:C6    [0/6]
    Source Address:                00:16:D3:1A:AA:6E    [6/6]
    Protocol:                      0x0806               (ARP)   [12/2]
ARP - Address Resolution Protocol  [14/28]
    Hardware type:                 1                    (Ethernet)  [14/2]
    Protocol Type:                 0x0800               [16/2]
    Hardware Address Length:       6                    [18/1]
    Protocol Address Length:       4                    [19/1]
    Type:                          1                    (ARP Request)  [20/2]
    Source Physics:                00:00:00:00:00:00    [22/6]
    Source IP:                     192.168.1.102        [28/4]
    Destination Physics:           00:E0:4C:E9:E8:C6    [32/6]
    Destination IP:                192.168.1.100        [38/4]
Extra Data:                        [42/18]
    Number of Bytes:               18 bytes    [42/18]
FCS - Frame Check Sequence:
    FCS:                           0x61840077    (Calculated)
```

```
0000  00 E0 4C E9 E8 C6 00 16 D3 1A AA 6E 08 06 00 01 08
0011  00 06 04 00 01 00 00 00 00 00 00 C0 A8 01 66 00 E0
0022  4C E9 E8 C6 C0 A8 01 64 00 00 00 00 00 00 00 00 00
0033  00 00 00 00 00 00 00 00 00
```

Send Selected Packets
Options
Adapter: Intel(R) PRO/100 VE Network Connection (Microsoft's
☑ Burst Mode (no delay between packets)
☑ Loop Sending: 20 loops (zero for infinite loop)
Delay Between Loops: 100 milliseconds
Sending Information
Total Packets: 1 * 20 = 20
Packets Sent: 20
Progress:

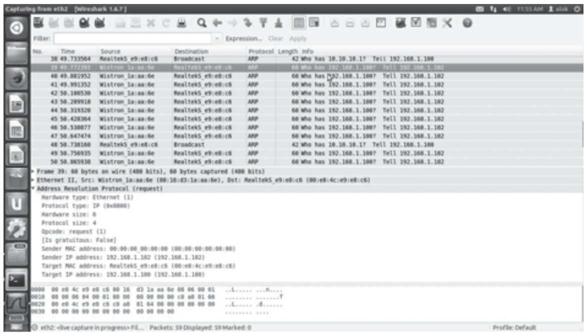
Fig. 7 Attacker crafts and sends ARP Request Packet with Source MAC 00:00:00:00:00:00


Wireshark
11:48 AM alok
Capturing from eth2 [Wireshark 1.6.7]
Capturing from eth0 [Wireshark 1.6.7]
Filter:
Expression... Clear

```
   Protocol Length Info
00:00:00   ARP   42 192.168.1.100 is at 00:e0:4c:e9:e8:c6
e9:e8:c6   ARP   60 Who has 192.168.1.100?  Tell 192.168.1.102
00:00:00   ARP   42 192.168.1.100 is at 00:e0:4c:e9:e8:c6
e9:e8:c6   ARP   60 Who has 192.168.1.100?  Tell 192.168.1.102
00:00:00   ARP   42 192.168.1.100 is at 00:e0:4c:e9:e8:c6
e9:e8:c6   ARP   60 Who has 192.168.1.100?  Tell 192.168.1.102
00:00:00   ARP   42 192.168.1.100 is at 00:e0:4c:e9:e8:c6
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
           ARP   42 Who has 10.10.10.1?  Tell 192.168.1.100
▸ Frame 360: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
▸ Ethernet II, Src: Wistron_1a:aa:6e (00:16:d3:1a:aa:6e), Dst: RealtekS_e9:e8:c6
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    [Is gratuitous: False]
    Sender MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Sender IP address: 192.168.1.102 (192.168.1.102)
    Target MAC address: RealtekS_e9:e8:c6 (00:e0:4c:e9:e8:c6)
Frame (frame), 60 bytes    Packets: 367 Displaye    Profile: Default
```

```
No.  Time         Source             Destination        Protocol Length
128  322.960045   10.10.10.21        91.189.89.22       TCP      74
129  324.969049   192.168.1.100      10.10.10.21        ICMP     102
130  324.969078   192.168.1.100      10.10.10.21        ICMP     102
131  324.969088   192.168.1.100      10.10.10.21        ICMP     102
132  324.969097   192.168.1.100      10.10.10.21        ICMP     102
133  324.969116   10.10.10.21        91.189.89.22       TCP      74
134  324.969146   10.10.10.21        91.189.89.22       TCP      74
135  326.012107   Pegatron_18:3e:65  Giga-Byt_c8:86:73  ARP      60
136  326.012130   Giga-Byt_c8:86:73  Pegatron_18:3e:65  ARP      42
137  327.969048   192.168.1.100      10.10.10.21        ICMP     102
138  327.969074   192.168.1.100      10.10.10.21        ICMP     102
139  332.981020   Giga-Byt_c8:86:73  Pegatron_18:3e:65  ARP      42
140  332.981060   Pegatron_18:3e:65  Giga-Byt_c8:86:73  ARP      60
▸ Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
▸ Ethernet II, Src: Pegatron_18:3e:65 (70:71:bc:18:3e:65), Dst: Giga-Byt_c8:86:7
▸ Address Resolution Protocol (request)
0000  6c f0 49 c8 86 73 70 71  bc 18 3e 65 08 06 00 01   l.I..spq ..>e....
0010  08 00 06 04 00 01 70 71  bc 18 3e 65 0a 0a 0a 15   ......pq ..>e....
0020  00 00 00 00 00 00 0a 0a  0a 01 00 00 00 00 00 00   ........ ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
eth0: <live capture in progress> Fil...   Packets: 140 Dis...   Profile: Default
```

Fig. 8 ARP Request Packet with Source MAC 00:00:00:00:00:00 being sent before filtration


Capturing from eth2 [Wireshark 1.6.7]
11:55 AM alok
Filter:
Expression... Clear Apply

```
No.  Time        Source              Destination         Protocol Length Info
 38  49.733564   RealtekS_e9:e8:c6   Broadcast           ARP      42 Who has 10.10.10.1?  Tell 192.168.1.100
 39  49.772393   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 40  49.881952   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 41  49.991352   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 42  50.100530   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 43  50.209918   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 44  50.319320   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 45  50.428364   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 46  50.538077   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 47  50.647474   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 48  50.738160   RealtekS_e9:e8:c6   Broadcast           ARP      42 Who has 10.10.10.1?  Tell 192.168.1.100
 49  50.756935   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
 50  50.865938   Wistron_1a:aa:6e    RealtekS_e9:e8:c6   ARP      60 Who has 192.168.1.100?  Tell 192.168.1.102
▸ Frame 39: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
▸ Ethernet II, Src: Wistron_1a:aa:6e (00:16:d3:1a:aa:6e), Dst: RealtekS_e9:e8:c6 (00:e0:4c:e9:e8:c6)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    [Is gratuitous: False]
    Sender MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Sender IP address: 192.168.1.102 (192.168.1.102)
    Target MAC address: RealtekS_e9:e8:c6 (00:e0:4c:e9:e8:c6)
    Target IP address: 192.168.1.100 (192.168.1.100)
0000  00 e0 4c e9 e8 c6 00 16  d3 1a aa 6e 08 06 00 01   ..L..... ...n....
0010  08 00 06 04 00 01 00 00  00 00 00 00 c0 a8 01 66   ........ .......f
0020  00 e0 4c e9 e8 c6 c0 a8  01 64 00 00 00 00 00 00   ..L..... .d......
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....
eth2: <live capture in progress> Fil...   Packets: 59 Displayed: 59 Marked: 0   Profile: Default
```

Fig. 9 ARP Request Packet with Source MAC 00:00:00:00:00:00 not being sent after filtration

IV. CONCLUSION

In this paper we have highlighted the security based issues of Local Area Network and shown how some of the known vulnerabilities of the basic protocols of TCP / IP protocol suite can be exploited to launch different types of attacks.

The proposed solution incorporates cross layer inspection, identifies invalid combinations of source and destination IP addresses and MAC addresses, port scanning, restoring of default gateways and helps the victim machine to recover from Spoofing and Poisoning based attacks in the Local Area Networks.

The proposed solution does not require any additional hardware and is fully backward compatible with existing versions of ARP as no modifications are required to the existing LAN protocols.

The aim of this paper is purely academic research and to spread awareness amongst the network administrators and other related persons who manage, maintain and guard the networks against such attacks in LAN and WAN environments Though highlighted, we do not intend to promote attack mechanisms nor defame any proprietary or open-source network defense tools already existing in the market.

REFERENCES

[1] Hansche, "Elecommunications, Network, and Internet Security", (ISC) 2 Press, 2003.

[2] Pandey A., Saini J. R. "*Study of Emerging Trends of Cyber Attacks in Indian Cyber space and their Countermeasures"* International Journal of Computer Science & Communication Networks 2249-5789

[3] "Cyber Attacks Explained Network Sniffing.", LINUX For You, Jan 10 2012 Issue

[4] www.linuxforu.com

[5] El-Hajj, Zouheir Trabelsi and Wassim*, "On investigating ARP Spoofing Security Solutions"*, International. Journal of Internet Protocol Technology, Inrscience Enterprises Ltd., 2010, Vol. 5.

[6] S. G. Bhirud. "Light weight approach for IP-ARP spoofing detection and prevention", 2011 Second Asian Himalayas International Conference on Internet (AH-ICI), 11/2011

[7] Mateti, Prabhakar. [Online] http://cecs.wright.edu/~pmateti/Courses/4420/Probing/index.html [Accessed: November 2012]

[8] Mitchell, Bradley, http://compnetworking.about.com/od/networkprotocols/g/ bldef_arp.htm. [Online] [Accessed: December 2012]

[9] Khaled Shuaib. "NIS04-4: Man in the Middle Intrusion Detection", IEEE Globecom 2006, 11/2006

[10] F. A. Barbhuiya. "An Active Host-Based Detection Mechanism for ARP-Related Attacks", Communications in Computer and Information Science, 2011

[11] Kumar, Sumit, and Shashikala Tapaswi. "A centralized detection and prevention technique against ARP poisoning", Proceedings Title 2012 International Conference on Cyber Security Cyber Warfare and Digital Forensic (CyberSec), 2012.

[12] Mohamed Al-Hemairy, Saad Amin, and Zouheir Trabelsi."*Towards More Sophisticated ARP Spoofing Detection/ Prevention Systems in LAN Networks"* : CTIT, December 2009.

[13] Barbhuiya, Ferdous A., Santosh Biswas, Neminath Hubballi, and Sukumar Nandi. "A host based DES approach for detecting ARP spoofing", 2011 IEEE Symposium CICS, 2011.

[14] M., Ahmed, Wail S. Elkilani, and Khalid M. Amin. "An Automated approach for Preventing ARP Spoofing Attack using Static ARP Entries", International Journal of Advanced Computer Science and Applications, 2014.

[15] J.C.Gondim, Marco Antonio Carnut & Joao., "*Arp Spoofing Detection on Swtched Ethernet Networks:A Feasibility Study"*: Symposium on Security in Information Practices, Nov. 2003.

[16] Mohamed Al-Hemairy. "Towards more sophisticated ARP Spoofing detection/prevention systems in LAN networks", 2009 CTIT, 12/2009

[17] Trabelsi, Zouheir. "Hands-on lab exercises implementation of DoS and MiM attacks using ARP cache poisoning", Proceedings of the 2011 Information Security Curriculum Development Conference on - InfoSecCD 11

[18] Pandey A., Saini J. R. *" Counter Measures to Combat Misuses of MAC address Spoofing Techniques"* IJANA Vol. 03, Issue 05, 0975-0282

[19] I.Bonilla, Christina L.Abad and Rafael"*An Analysis on the schemes for Detecting and Preventing ARP cache Poisoning Attacks",* 27th International Conference on distributed Computing system Workshops, June 2007. ICDCSW'07.

[20] Masuai, Soumnuk Puangpronpitag & Narongit."*An Efficient and Feasible Solution to ARP Spoof Problem",* 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology , 2009. (ECTI-CON 2009)

[21] Katkar, Dr. S. G. Bhirud and Vijay."*Light Weight Approach for IP-ARP Spoofing Detection and Prevention"*: Second Asian Himalayas International Conference on Internet , November 2011. (AH-ICI)

[22] Andre P. Ortega, Xavier E. Marcos, Luis D. Chiang and Cristina L. Abad."*Preventing ARP Cache Pisoning Attacks: A proof of concept using OpenWrt"*.: Latin American Network Operations and Management Symposium, October 2009. (LANOMS)

[23] Qian, Ai-Zeng. *The Automatic Prevention and Control Research of ARP Deception and Implementation"*. : WRI World Congress on Computer Science and Information Engineering, April 2009.

[24] Boughrara, A. and Mammar, S."*Implementation of a SNORT's output Plug-In in reaction to ARP Spoofing's attack"*. : 6th International Conference on Sciences of Electronics Technologies of Information and Telecommunications, March 2012. (SETIT)

[25] Pandey A., Saini J. R. *"Centralised Web based allocation and management approach towards IP addressing for providing Mobility and Security"* International Journal Of Emerging Trends & Technology in Computer Science, Vol-3, Issue-3, 2278-6856

[26] Sanguankotchakorn, Teerapat, and Thanatorn Dechasawatwong. "*Automatic attack detection and correction system development",* 2011 13th Asia-Pacific Network Operations and Management Symposium, 2011.

[27] Pandey A., Saini J. R. *" A Simplified Defense Mechanism Against Man in the Middle Attack"* IJEIR ,Jan 2014 Vol 1, Issue 5, 2277-5668

[28] Pandey A., Saini J. R.*" Attacks and Defense Mechanisms for TCP/IP based Protocols"* IJEIR ,Jan 2014 Vol3, Issue 1, 2277-5668

[29] JUNIPER. 2006. *DDOS Secure* [Online]. Available: http://www.juniper.net/techpubs/software/management/ddos/ddos5.13.1/ ddos-secure-1200-quick-start-guide.pdf.

[30] http://www.symantec.com/security_response/definitions.jsp

[31] Pandey A., Saini J. R. *"Simplified TCP based approach towards Domain Name Service for Improving Security"* International Journal Of Computer Science and Communication Networks, Dec 2013, Vol 2 , 2249-5789