Secure ARP and Secure DHCP Protocols to Mitigate Security Attacks

Biju Issac
Information and Security Research (iSECURES) Lab, Swinburne University of Technology (Sarawak Campus)
Kuching, Sarawak, Malaysia (Email: bissac@swinburne.edu.my)

(Received Apr. 24, 2007; revised Aug. 31, 2007; and accepted Jan. 30, 2008)

Abstract

For network computers to communicate to one another, they need to know one another's IP address and MAC address. Address Resolution Protocol (ARP) is developed to find the Ethernet address that map to a specific IP address. The source computer broadcasts the request for Ethernet address and eventually the target computer replies. The IP to Ethernet address mapping would later be stored in an ARP Cache for some time duration, after which the process is repeated. Since ARP is susceptible to ARP poisoning attacks, we propose to make it unicast, centralized and secure, along with a secure design of DHCP protocol to mitigate MAC spoofing. The secure protocol designs are explained in detail. Lastly we also discuss some performance issues to show how the proposed protocols work.

Keywords: Address Resolution Protocol, DHCP, MAC address and network security

1 Introduction

The data link layer hardware does not understand the IP addresses. It only understands the physical address or MAC address. A computer cannot use MAC address alone to communicate to others in a network. Usually the computers are attached to any network using a network interface card that has with a unique physical address called as MAC address (or 48-bit Ethernet address). No two cards would have the same address since such network card manufacturers get the card numbers from a central authority that would assign only unique MAC addresses. This can very well avoid MAC address conflict. These cards know nothing about the IP address of the computer where it is housed [10]. In the following sections we outline two new protocols – a new centralized protocol called Secure Unicast Address Resolution Protocol (S-UARP) to mitigate ARP poisoning attacks and a new secure DHCP protocol to mitigate MAC spoofing attacks. The organization of further sections are as follows: Section 2 describes on the current ARP and DHCP protocols, Section 3 briefly explains on the ARP poisoning and its implementation, Section 4 shows related work, Section 5 to 7 explains on the S-UARP protocol proposal and related issues, Section 8 is on MAC spoofing attack and implementation, Section 9 is on secure DHCP protocol, Section 10 is performance analysis and Section 11 is the conclusion.

2 The ARP and DHCP Protocols

2.1 The Address Resolution Protocol

Address Resolution Protocol (ARP) which is defined in RFC 826 [9] is used to map the IP addresses onto the data link layer MAC address. It is explained as follows. Consider the Figure 1 on interconnected networks.

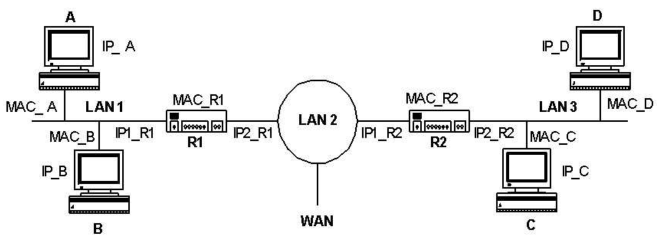


Figure 1: Interconnected networks (two bus networks with a ring network in middle). Each computer on LAN has been shown with an IP address and MAC address.

We note that two computers (A and B) on LAN1 have IP address, MAC address pair as [IP_A, MAC_A] and [IP_B, MAC_B] respectively. Similarly, two computers on LAN3 (C and D) have IP address, MAC address pair as [IP_C, MAC_C], [IP_D, MAC_D]. Also note that the two routers (R1 and R2) between the networks have two IP address corresponding to the link to bus and ring network. Each router possesses unique MAC address. For a user A on LAN1 to send packets to user on B within LAN1 the following happens: A query to DNS would return the IP address IP_B. It then frames a packet with IP_B in the destination field and passes it to IP layer to transmit. The IP layer sees that the address is on the same network. But it needs to find B's MAC address. To find that it broadcasts a packet asking, "Who own IP address IP_B?". This broadcast would reach on all computers in LAN1. Only computer B would respond with its MAC address MAC_B. Thus ARP works by this request and reply approach. The method is quite simple [10].

Some optimizations are possible with ARP. Once computer A gets the ARP reply from B, it stores that IP-to-MAC address mapping of B in a local cache. So if in a short period of time, if A wants to communicate with B, it refers to the local ARP cache, eliminating a second broadcast. Usually, A would include its IP-to-MAC address mapping in the ARP packet, thus informing B of its mapping. In fact all machines on LAN1 can enter this mapping information on A into their ARP cache. Another optimization is to have every computer broadcast its mapping when it boots, in the form of an ARP looking for its won address. To allow for changes in mapping, especially when network card breaks down, and is replaced with a new one, entries in ARP cache should time out after few minutes [6].

2.2 The Dynamic Host Configuration Protocol

DHCP stands for 'Dynamic Host Configuration Protocol' and is a way by which networked computers get their TCP/IP networking settings from a central server. Dynamic Host Control Protocol (DHCP) is defined in RFC 2131 [4] and 2132 [1]. It is an extension of BOOTP, the previous IP allocation specification. It allows manual and dynamic IP address assignment to computers that requests for that. DHCP server is not reachable by broadcasting from a different network. Hence a *DHCP relay agent* is needed to forward the DHCP DISCOVER broadcast packet from a newly booted machine. It is send as a unicast transmission to the DHCP server (which may be on another network) by the relay agent. The relay agent usually keeps the IP address of the DHCP server. Thus the relay agent is for relaying packets between servers and clients. This makes the DHCP server handle the sub-net that has no server available and thus there is no need to setup a server per sub-net. To keep track of the duration of IP address assignment, a DHCP server uses the concept of *leasing*. As mentioned before, the DHCP server assigns IP addresses automatically from a pool of IP addresses. If a compute leaves the network 'abruptly' and does not return the IP address that it was using, that IP address is lost for any further assignment. As a precaution to that, assignment of IP address is only for a fixed duration of time, called leasing. Just before the expiry of the lease, a computer should request the DHCP server for renewal. Otherwise, that IP address cannot be used further [10].

A DHCP client may receive offers from multiple DHCP servers and can accept any one of the offers; however, the client usually accepts the first offer it receives. Additionally, the offer from the DHCP server is not a guarantee that the IP address will be allocated to the client; however, the server usually reserves the address until the client has had a chance to formally request the address. The client returns a formal request for the offered IP address to the DHCP server in a DHCPREQUEST broadcast message. The DHCP server confirms that the IP address has been allocated to the client by returning a DHCPACK unicast message to the client as in Figure 2.

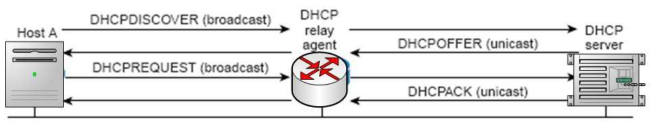


Figure 2: DHCP protocol operation

The formal request for the offered IP address (the DHCPREQUEST message) that is sent by the client is broadcast so that all other DHCP servers that received the DHCPDISCOVER broadcast message from the client can reclaim the IP addresses that they offered to the client.

If the configuration parameters sent to the client in the DHCPOFFER unicast message by the DHCP server are invalid (a misconfiguration error exists), the client returns a DHCPDECLINE broadcast message to the DHCP server.

The DHCP server will send to the client a DHCPNAK denial broadcast message, which means the offered configuration parameters have not been assigned, if an error has occurred during the negotiation of the parameters or the client has been slow in responding to the DHCPOFFER message (the DHCP server assigned the parameters to another client) of the DHCP server.

3 ARP Poisoning Security Attack and Implementation

In ARP Poisoning, forged ARP request and reply packets are used to update the target computer's ARP cache. The target computer is being fooled into believing that the attacker computer (which has a totally different MAC and IP address) as the computer that has the desired IP address with a specific MAC address. Thus, the attacker can monitor the packet sent by the target computer to the original destination since it is sent to the attacker's computer first before they are sent to the original destination [7].

In the ARP poisoning experiment, two desktop computers and one laptop was used as in Figure 3. The two desktop computers (Computer A and Computer B) acted as the victims while the laptop (Computer C) acted as the attacker as in Figure 4. A was the source while B was the destination. C was equipped with the Ethereal packet capturing software [11] and the ARP poisoning software known as Cain & Abel [12]. Computer A was used to send continuous ICMP packets to B by pinging B. When ARP poisoning was carried out using Cain and Abel software installed on C on Computer A's ARP cache,

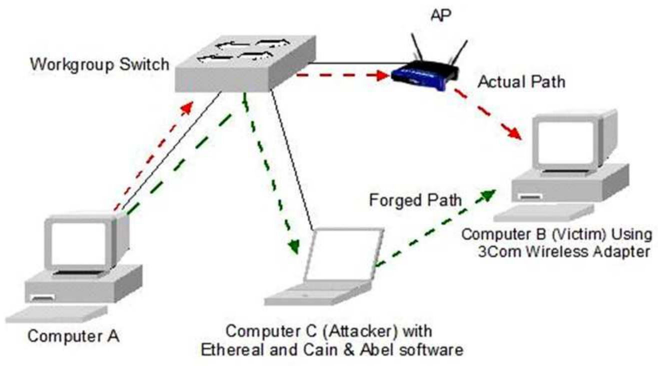
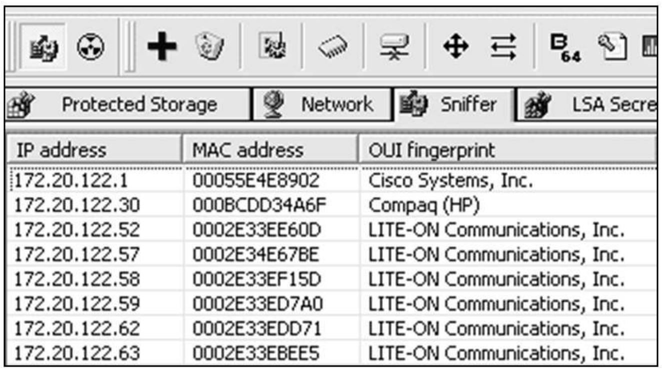

AP
Workgroup Switch
Actual Path
Forged Path
Computer B (Victim) Using 3Com Wireless Adapter
Computer A
Computer C (Attacker) with Ethereal and Cain & Abel software

Figure 3: ARP poisoning implementation in our lab



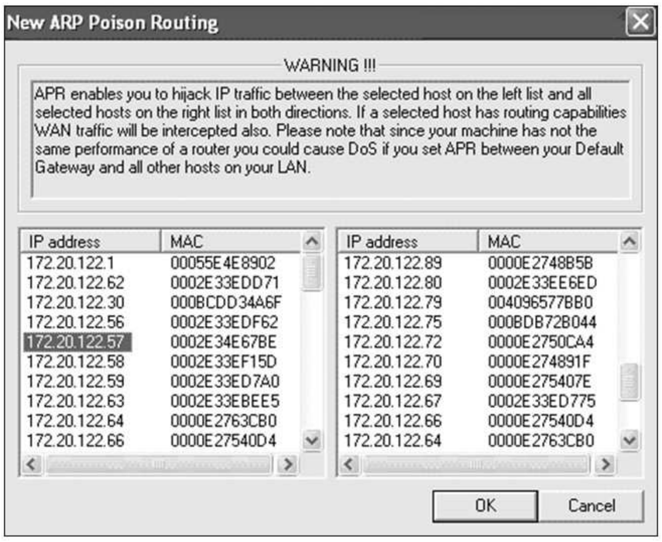
IP address	MAC address	OUI fingerprint
172.20.122.1	00055E4E8902	Cisco Systems, Inc.
172.20.122.30	000BCDD34A6F	Compaq (HP)
172.20.122.52	0002E33EE60D	LITE-ON Communications, Inc.
172.20.122.57	0002E34E67BE	LITE-ON Communications, Inc.
172.20.122.58	0002E33EF15D	LITE-ON Communications, Inc.
172.20.122.59	0002E33ED7A0	LITE-ON Communications, Inc.
172.20.122.62	0002E33EDD71	LITE-ON Communications, Inc.
172.20.122.63	0002E33EBEE5	LITE-ON Communications, Inc.

Figure 4: List of hosts in the network is shown using Cain and Abel software

it was observed in the Ethereal software on C that the ICMP packets were sent only between Computer A and Computer C (attacker), even though A sent it to B. In Cain and Abel software, it was observed that Computer C could monitor the ICMP packets sent between those two computers. It showed that Computer A has been fooled to send ICMP packets to Computer C, which has a different set of MAC and IP address from Computer B. Also, Computer C could then forward these packets to B, after keeping a copy to itself.

Thus ARP poisoning is a method of attacking a network by updating the target computer's ARP cache with forged ARP request and reply packets in an effort to change the Layer 2 Ethernet MAC address to one that the attacker can monitor. The target computer sends frames that were meant for the original destination to the attacker's computer, so that the frames can be read since the ARP replies have been forged. A successful ARP attempt is invisible to the user.

The actual configuration used for the attack in our lab is as follows. Computer A and Computer B were the victims and Computer C acted as the attacker.



WARNING !!!

APR enables you to hijack IP traffic between the selected host on the left list and all selected hosts on the right list in both directions. If a selected host has routing capabilities WAN traffic will be intercepted also. Please note that since your machine has not the same performance of a router you could cause DoS if you set APR between your Default Gateway and all other hosts on your LAN.

IP address	MAC
172.20.122.1	00055E4E8902
172.20.122.62	0002E33EDD71
172.20.122.30	000BCDD34A6F
172.20.122.56	0002E33EDF62
172.20.122.57	0002E34E67BE
172.20.122.58	0002E33EF15D
172.20.122.59	0002E33ED7A0
172.20.122.63	0002E33EBEE5
172.20.122.64	0000E2763CB0
172.20.122.66	0000E27540D4

IP address	MAC
172.20.122.89	0000E2748B5B
172.20.122.80	0002E33EE6ED
172.20.122.79	004096577BB0
172.20.122.75	000BDB72B044
172.20.122.72	0000E2750CA4
172.20.122.70	0000E274891F
172.20.122.69	0000E275407E
172.20.122.67	0002E33ED775
172.20.122.66	0000E27540D4
172.20.122.64	0000E2763CB0

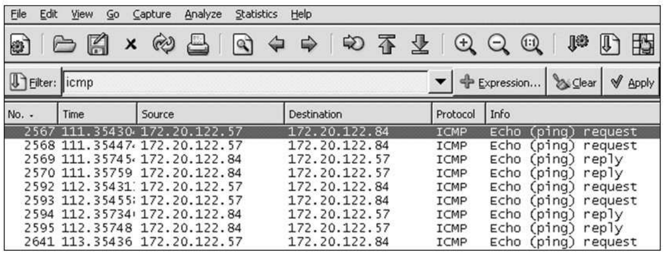
Figure 5: Selection of target computers to perform ARP poisoning in Cain and Abel software.

Computer A:	Desktop computer (IP address: 172.20.122.84) Ethernet adapter
Computer B:	Desktop computer (IP address: 172.20.122.57) 3 Com 11Mbps Office Wireless Adapter
Computer C:	Laptop (IP address 172.20.122.114) VT6105 Rhine III Fast Ethernet Adapter Software used – Cain and Abel, Ethereal

Initially at Computer B, the ping command was given continuously to Computer A by issuing "ping –t 172.20.122.84" at the command prompt. At Computer C, Ethereal was initiated and the appropriate Ethernet adapter was selected. Computer C could only capture packets that were to and from it and broadcast packets. Thus Cain and Abel (ARP poisoning) software was needed to redirect the traffic to Computer C, so that it could sniff the ICMP packets. Without Cain and Abel, no ICMP packets would be shown in the Ethereal software (on C) as Computer B was pinging Computer A.

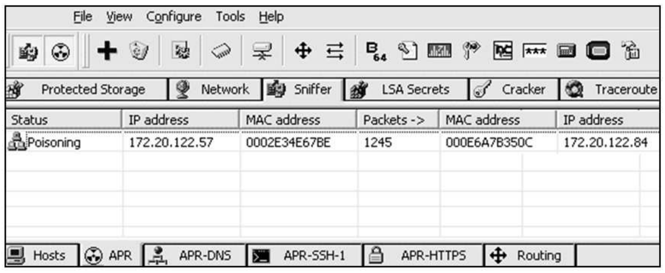
Cain and Abel software was started and the appropriate adapter based on IP address is selected. Next the sniffer was started and all the hosts were selected for MAC address scanning. Then the lists of hosts available in the network would be displayed on Cain and Abel software.

The ARP tab was clicked and "+ " tab is selected to add the victim computers to be poisoned. Computer B's IP was selected at the left column of the "New ARP Poison Routing" pop up window and Computer A's IP at the right column was also selected as in Figure 5. The start ARP tab was clicked to start the poisoning. The status "Idle" would change to "Poisoning" and packets transfer between Computer A and Computer B could be noticed on Computer C (attacker). Using Ethereal on Computer C, the ICMP packets transfer between Computer A and Computer B are shown in Figure 6. The communication



```
No. -  Time       Source          Destination     Protocol  Info
  2567 111.35430  172.20.122.57   172.20.122.84   ICMP      Echo (ping) request
  2568 111.35447  172.20.122.57   172.20.122.84   ICMP      Echo (ping) request
  2569 111.35745  172.20.122.84   172.20.122.57   ICMP      Echo (ping) reply
  2570 111.35759  172.20.122.84   172.20.122.57   ICMP      Echo (ping) reply
  2592 112.35431  172.20.122.57   172.20.122.84   ICMP      Echo (ping) request
  2593 112.35455  172.20.122.57   172.20.122.84   ICMP      Echo (ping) request
  2594 112.35734  172.20.122.84   172.20.122.57   ICMP      Echo (ping) reply
  2595 112.35748  172.20.122.84   172.20.122.57   ICMP      Echo (ping) reply
  2641 113.35436  172.20.122.57   172.20.122.84   ICMP      Echo (ping) request
```

Figure 6: ICMP packets transferred between the target computers shown in Ethereal.



Status	IP address	MAC address	Packets ->	MAC address	IP address
Poisoning	172.20.122.57	0002E34E67BE	1245	000E6A7B350C	172.20.122.84

Figure 7: ICMP packets transferred between A and B is monitored by Cain and Abel software.

between A and B can be monitored through Cain and Abel software as shown in Figure 7.

4 Related Work on Secure ARP

A research publication [3] on Secure ARP (S-ARP) has been done by D. Bruschi et al. which deals with ARP broadcast communication security. Here each host has a public/private key pair certified by a local trusted party on the LAN, which acts as a Certification Authority. Messages are digitally signed by the sender, thus preventing the injection of spurious and/or spoofed information. It has been implemented also in Linux [3]. Tripunitara et. al. had outlined a middleware approach to the prevention of ARP cache poisoning as given in [14].

5 The Secure Unicast ARP (S-UARP) Protocol

The S-UARP proposal we make is unicast in nature and have different options for security implementation. Many organizations would have implemented a DHCP server for dynamic IP address assignment to individual machines in a LAN. Hence the DHCP server can be configured to have the MAC-to-IP address mapping or vice-versa for all the computers/hosts under its domain. We propose to extend the DHCP protocol to handle Secure Unicast Address Resolution Protocol (S-UARP) packets. We denote such a server as DHCP+ server from now on. The DHCP relay agent also needs to be modified to forward the S-UARP request/response messages. When using dynamic IP addressing using DHCP, the DHCP+ server stores the mapping of IP to MAC address as it leases out the IP address to the requesting hosts. We are not dealing with static IP addressing option in this section. But some suitable modification to this protocol can make it suitable for static addressing as noted in the next section. The proposal itself has an inherent partial-security against eavesdropping compared to ARP broadcast in a wired network, since packets are unicast in nature and is not broadcasted. In a wireless network, a packet sniffer can capture these unicast packets too since the radio transmission has no defined boundaries of transmission. But we add security into our protocol proposal.

5.1 S-UARP Protocol

This is a centralized protocol unlike the decentralized approach in normal ARP. Consider the following notations and their meaning as shown below.

Notation	**M**eaning.
S-UARP_req:	S-UARP Request Packet.
S-UARP_res:	S-UARP Response Packet.
DHCP+:	DHCP+ Server.
ICP:	Integrity Check Pass (security flag).
ICF:	Integrity Check Fail (security flag).
A:	Host A.
B:	Host B.
IP_A:	IP address of A.
MAC_A:	MAC address of A.
IP_B:	IP address of B.
MAC_B:	MAC address of B.
S_K:	Session key.
K_{SA}:	Shared secret key between host A and the server.
MIC:	Message Integrity Code.
H:	Collision Free One-Way Hash Function.
t:	Time (independent variable) with one or more independent values.
t1:	Time period (duration) when receiver waits for S-UARP_req.
t2:	Time period when sender looks for a packet to be sent to the same host where ACK has to be sent.
t3:	Time period within ACK packet has to be sent. ($t3 > t2$)
t4:	Time period after which S-UARP cache needs refreshing.

The S-UARP protocol (for dynamic IP addressing) is described as follows in 3 steps:

1) A $\rightarrow$ DHCP+: S-UARP_req.

2) DHCP+ $\rightarrow$ A: S-UARP_res + MIC.

3) A $\rightarrow$ DHCP+: (ACK)K_{SA}.

A simple example and explanation to show how this can be implemented with DES algorithm is as follows:

1) When a host *A* wants to communicate to host *B*, it sends a S-UARP request packet (unicast packet) to the DHCP+ server (which answers the S-UARP packets), instead of sending a broadcast to all. We assume that the secret hashing key (K_{SA}) is distributed between the client and the server, using private-public key mechanism or any other secure mechanism.

2) The DHCP+ server encrypts the response message using DES with cipher block chaining (CBC). It cuts the message (S-UARP_res) into predetermined-sized of *i* blocks (where $i = 1, 2, \ldots, n$). Use the CBC residue (that is the last block output by CBC process) as a message integrity code (MIC). This MIC would act as a checksum [8]. The plaintext message plus the MIC would be transmitted to the host (receiver) or A. i.e. DHCP+ Server Host A: Transmit S-UARP response (plain text) + MIC. The transmitted response message will be as in Figure 8.

S-UARP response (plain text)	MIC (CBC residue)

Figure 8: The S-UARP response message and MIC transmitted from DHCP+ Server.

If the response message doesn't arrive within a time period *t1*, host A will retransmit another S-UARP request packet to server. This can continue until it gets a request packet.

3) Once the UARP response is received, host A checks for validity by using its secret key. The receiver (Host A) encrypts the plaintext S-UARP_res using DES that it received with the shared secret key and do the hashing process to produce similar MIC (say, MIC*). Finally it checks the CBC residue or MIC. If MIC = MIC*, the message is a non-tampered in transit. We then call it Integrity Check Pass (ICP) state. Otherwise it is Integrity Check Fail (ICF) state and is discarded. The S-UARP response contains time t_s when it was generated by the server. Host A also checks the freshness of the response by checking $t_r - t_s = \Delta t$ (similar to *t*3), where t_r is the time when A receives the response from the server and Δt is the accepted time interval for transmission delay. Finally, the host A sends an encrypted acknowledgment $(ACK)K_{SA}$ to the server. ACK contains the timestamp t_a generated by the host A to ensure that the message is fresh and is not a replay.

The entries in S-UARP cache remains valid for a time period, *t4* (say, in minutes) as in ARP protocol. Once that time period expires, a new S-UARP request need to be sent by a host to DHCP+ server to get the IP-to-MAC address mapping. This can deal with a situation of change in ethernet card for a machine.

5.2 Detailed Explanation

The protocol can be shown in detail as follows, with possible optimization (as explained under Section 5.3). When DHCP+ Server assigns a dynamic IP address to a host, the IP and MAC address of the DHCP+ server should be made known to the host.

```
Procedure S-UARP_Communication (A→B)
  BEGIN:
  Initialize the flag [pkt_send (from→to)] = failure;
  while (pkt_send (A→DHCP+) == failure)
  {
     Initialize t;
     S-UARP_req (IP_A, MAC_A, IP_B);
     A→DHCP+: Sends S-UARP_req; //no broadcast
     if ($t < t1$)
        pkt_send (A→DHCP+) = success;
     else
        pkt_send (A→DHCP+) = failure;
  } //while loop
  while  (pkt_send (A→DHCP+) == success || $t > t3$)
  {
     Initialize t;
     S-UARP_res (IP_A, MAC_A, IP_B, MAC_B, $t_s$)
     DHCP+→A: Sends UARP_res + MIC;
     if (pkt_send (DHCP+→A) == success && $t < t2$ &&
      ICP)
     {
        Host A→DHCP+: Piggyback (ACK)$K_{SA}$;
        if (pkt_send (A→DHCP+) == success)
           S-UARP Cache updated;
        else
           Go to start of enclosed while loop; flag = success;
        A→B: A communicates to B directly;
     }
     else if (pkt_send (DHCP+→A) == success &&
      $t2 < t < t3$ && ICP)
     {
        Host A DHCP+: Sends (ACK)$K_{SA}$ packet;
        if (pkt_send (A→DHCP+) == success)
           S-UARP Cache updated;
        else
           Go to start of enclosed while loop; flag = success;
        A→B: A communicates to B directly;
     }
     else if (pkt_send (DHCP+→A) == failure || $t > t3$)
     {
        Go to start of enclosed while loop;
     }
  } //while loop
  if  ($t > t4$ || ICF )
     S-UARP_Communication (A→B);
  END: //end of procedure
```

5.3 Possible Optimization

An optimization possible is that the ACK can be piggybacked on another packet to the DHCP+ server, if packet transmission from host A to server happens within time $t2$. This can eliminate the separate ACK packet sent and save ACK congestion in the network. If there is no scope for piggybacking, and the acknowledgement is not received within a reasonable time period $t3$ (where $t3 > t2$), the server sends the S-UARP response packet again. If the S-UARP response packet is received by the host and the ACK packet is lost on transit, the duplicate response packets send by the server (after timeout $t3$) would be rejected.

5.4 Flow Chart for S-UARP

The flow chart for the S-UARP protocol can be shown as in Figure 9. It depicts the scenario when Host A wants to communicate to Host B (or a general Host X) and how the protocol works with respect to different time durations.

Note in Figure 9, $t1$ is the maximum time period for S-UARP response arrival (if it fails, host A would send another request), $t2$ is the maximum wait time for sending piggybacked ACK, $t3$ is the maximum acknowledgement wait time for sending ACK packet, where $t3 > t2$ (if $t > t3$, the server would send response again) and $t4$ is the maximum wait time, until S-UARP cache is refreshed.

5.5 Alternate S-UARP Protocols (with more security)

One of the limitations of the above protocol is that the request and the response are both in clear, though this is not a serious threat considering the content of the packets. Moreover, the message integrity is only on the server's response side.

Alternate Version 1: A better approach needs to ensure the integrity of both S-UARP request and response as follows:

Packet size:

1) A → DHCP+ : S-UARP_req + MIC1.

2) DHCP+ → A : S-UARP_res + MIC2.

3) A → DHCP+ : (ACK, NRN) K_{SA}.

In this protocol, we assume that a random number RN is known to both host and the server and is kept secret (generated by A or DHCP+). In Step 1, A sends the request in clear and the MIC (i.e. MIC1). The MIC1 is generated using a collision-free one-way hash function like SHA1 that takes the secret key K_{SA},, the S-UARP_req and the random number RN as inputs. That means, MIC1 = H(K_{SA}, RN, S-UARP_req). In Step 2, the server uses the S-UARP_req (in plain text), the known random number RN and secret key, K_{SA} to create a similar MIC (say, MIC1*). If MIC1 = MIC1*, then the request is accepted else it will be rejected. After verifying the integrity of the message, the server sends the response and MIC2 to the host. The MIC2 is generated in the same way (i.e. MIC2 = H(K_{SA}, RN, S-UARP_res). Finally in Step 3, Host A will check the integrity of the response as in the above case (to see MIC2 = MIC2*). Host A then sends an acknowledgement and a new random number (NRN) encrypted by the secret key (K_{SA}). NRN can be used in the next request/response exchange. As in the first protocol, the acknowledgment contains the timestamp to check when the server sent the response to the host, thus protecting against replay attacks.

Alternate Version 2: Another more secure alternative is to use a session key S_K and an Exclusive-OR (XOR) operation as follows:

1) A → DHCP+ : S-UARP_req + MIC1.

2) DHCP+ → A : S- UARP_res + SK ⊕ MIC2 + MIC3.

3) A → DHCP+ : MIC4.

When the clients power up they advertise their encrypted IP address and MAC address to a central server (DHCP-), using the symmetric key. The DHCP- server keeps a record of the IP address and MAC address of all hosts in that network, much like a DHCP server, but doesn't issue IP addresses. It advertises its identity on a frequent basis and this takes precedence over normal DHCP addressing (if any) and clients would know whom to contact during ARP request. The ARP requests would then go unicast to DHCP- server from the clients as shown before.

6 Co-existence of DHCP and DHCP+ Servers

The DHCP+ server is an 'improved' implementation of the normal DHCP server where it allows all DHCP queries to be directed to itself with security options, than doing a DHCP request broadcast as it is done in normal networks. If there is a situation where DHCP+ server implementation and DHCP server exist in the same network, the DHCP server needs to be patched to allow priority to DHCP+ server, so that the IP address assignment would only be done by the DHCP+ server. The software patching can help resolve the conflict of operation between the two within a same network. So in a co-existence scenario, the normal DHCP server would resign to a 'passive' mode and DHCP+ server would be in an "active" mode.

7 Mac Spoofing Attack and Implementation

Mac spoofing is done where an attacker alters the manufacturer-assigned MAC address to any other value

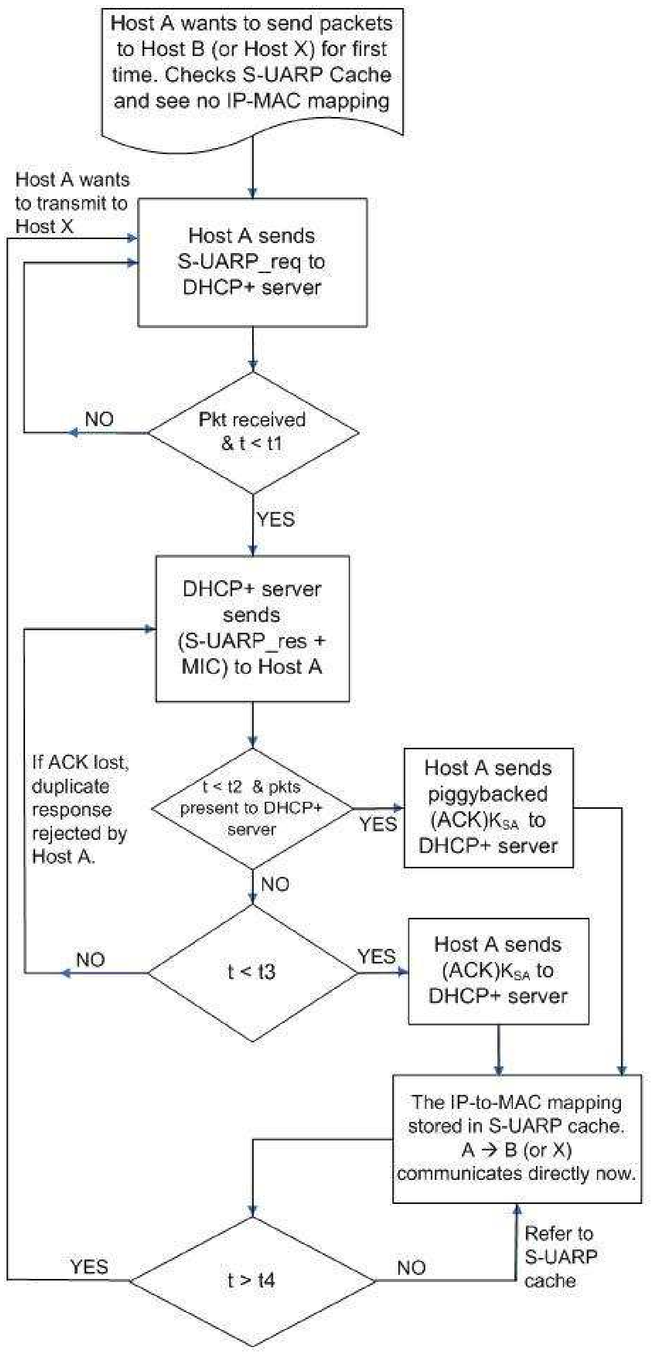
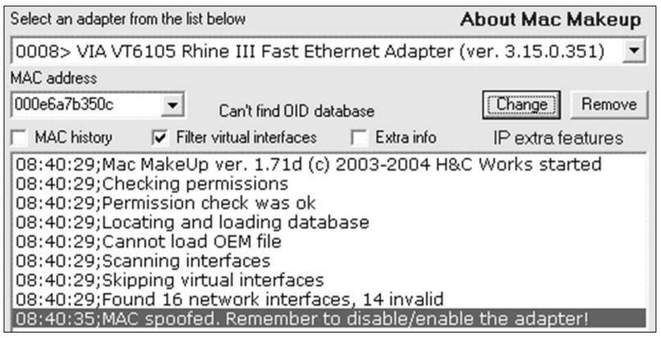

Host A wants to send packets to Host B (or Host X) for first time. Checks S-UARP Cache and see no IP-MAC mapping
Host A wants to transmit to Host X
Host A sends S-UARP_req to DHCP+ server
NO
Pkt received & t < t1
YES
DHCP+ server sends (S-UARP_res + MIC) to Host A
If ACK lost, duplicate response rejected by Host A.
t < t2 & pkts present to DHCP+ server
YES
Host A sends piggybacked (ACK)K_SA to DHCP+ server
NO
NO
t < t3
YES
Host A sends (ACK)K_SA to DHCP+ server
The IP-to-MAC mapping stored in S-UARP cache. A → B (or X) communicates directly now.
Refer to S-UARP cache
YES
t > t4
NO

Figure 9: The flowchart showing the procedure of S-UARP operation



0008> VIA VT6105 Rhine III Fast Ethernet Adapter (ver. 3.15.0.351)

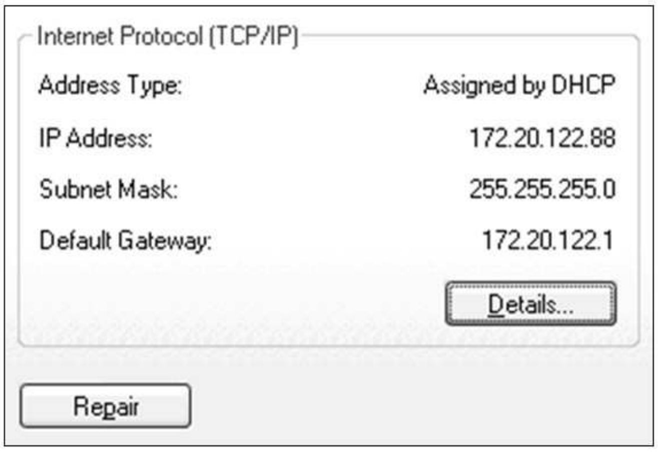
MAC address: 000e6a7b350c

Can't find OID database

☐ MAC history ☑ Filter virtual interfaces ☐ Extra info

```
08:40:29;Mac MakeUp ver. 1.71d (c) 2003-2004 H&C Works started
08:40:29;Checking permissions
08:40:29;Permission check was ok
08:40:29;Locating and loading database
08:40:29;Cannot load OEM file
08:40:29;Scanning interfaces
08:40:29;Skipping virtual interfaces
08:40:29;Found 16 network interfaces, 14 invalid
08:40:35;MAC spoofed. Remember to disable/enable the adapter!
```

Figure 10: Mac Makeup software used to perform MAC spoofing for Ethernet adapter.



Internet Protocol (TCP/IP)

Address Type:	Assigned by DHCP
IP Address:	172.20.122.88
Subnet Mask:	255.255.255.0
Default Gateway:	172.20.122.1

Figure 11: The IP address assigned to the attacker's ethernet card.

by using softwares like Mac Makeup [13, 15] shows details of such attack.

In a brief experiment on a wireless LAN, the MAC address of the Intel PRO/Wireless LAN 2100 3B Mini PCI Adapter that we were using was changed to the MAC address of a 3Com ll Mbps USB wireless adapter that was connected to the wireless network. Now it was found that the MAC address (0006a7b350c) and the IP address (172.20.122.88) assigned to the Intel Wireless Adapter is identical to the 3Com adapter.

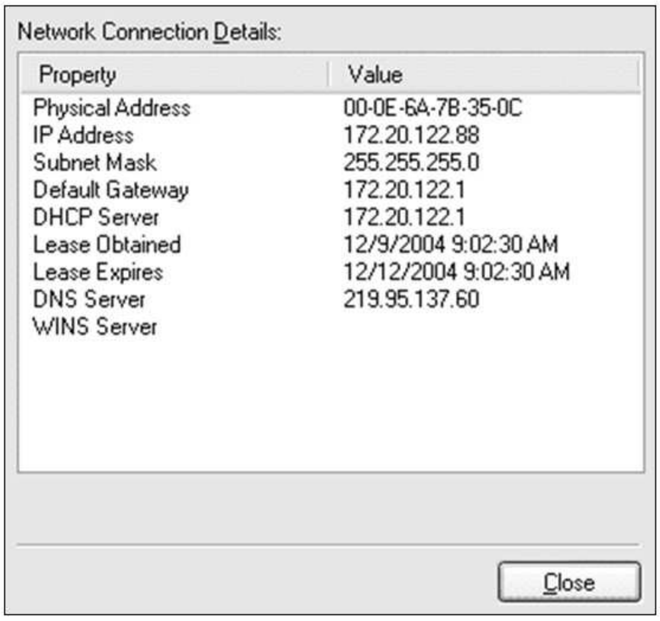
After identifying a MAC address to be spoofed, well-published DoS attack against the target was launched to cause the target's terminal to crash. In a real life attack, the attacker shall then immediately change the MAC address, IP address and default gateway to the value the target was using. With the target's computer rebooting, the attacker can access network resources bypassing the WLAN security appliance.

After MAC spoofing using Mac Makeup software as shown in Figure 10, the attacker's Ethernet card showed details as in Figure 11 and the overall network connection details as in Figure 12.

In our case, the presence of firewall in Cisco AP was bypassed when the Ethernet card was used to spoof the MAC address of the wireless adapter and the Internet was browsed on a computer that used spoofed address. The access point's association table showed that the attacker's computer using spoofed MAC address was connected to wireless network as shown in Figure 13.

8 Secure DHCP Protocol

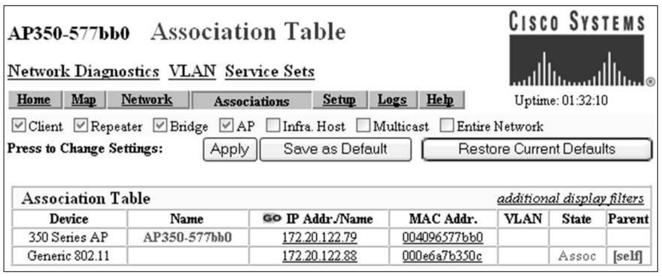
The dynamic address assignment that is done by DHCP server also needs to be secured, against hacking. MAC spoofing can easily be done through software to alter



Network Connection Details:

Property	Value
Physical Address	00-0E-6A-7B-35-0C
IP Address	172.20.122.88
Subnet Mask	255.255.255.0
Default Gateway	172.20.122.1
DHCP Server	172.20.122.1
Lease Obtained	12/9/2004 9:02:30 AM
Lease Expires	12/12/2004 9:02:30 AM
DNS Server	219.95.137.60
WINS Server	

Figure 12: The spoofed IP and MAC address shown at attacker's computer.



AP350-577bb0 Association Table

Uptime: 01:32:10

☑ Client ☑ Repeater ☑ Bridge ☑ AP ☐ Infra. Host ☐ Multicast ☐ Entire Network

Press to Change Settings:

Association Table

Device	Name	IP Addr./Name	MAC Addr.	VLAN	State	Parent
350 Series AP	AP350-577bb0	172.20.122.79	004096577bb0			
Generic 802.11		172.20.122.88	000e6a7b350c		Assoc	[self]

Figure 13: The spoofed IP and MAC address shown on access point's association table.

MAC addresses. We propose a secure DHCP (S-DHCP) protocol to make it less prone to MAC spoofing attacks. Related work was done by Komori and Saito in [5] and another in [2]. Even if the MAC address is spoofed, the secure DHCP server will not assign the IP address, without proper credentials, as shown below. They don't take any additional steps than usual DHCP, except for the MIC overhead. The S-DHCP protocol is described as follows in 4 steps:

1) A $\rightarrow$ ALL: Broadcast S-DHCP_DISCOVER.
2) DHCP+ $\rightarrow$ A: S-DHCP_OFFER + (MIC1)K$_{SA}$ (unicast).
3) A $\rightarrow$ DHCP+: S-DHCP_REQUEST + MIC1 + MIC2, to all DHCP servers that responded.
4) DHCP+ $\rightarrow$ A: S-DHCP_ACK (unicast).

Explanation of the secure version is generally similar to that for secure ARP. Host A broadcast a normal S-DHCP_DISCOVER message packet. The DHCP+ server responds with a unicast secure S-DHCP_OFFER message (that contains the IP address) appended with an encrypted Message Integrity Code (MIC1) using K$_{SA}$. K$_{SA}$ is the shared secret key between host A and DHCP+ server. MIC1 can be the CBC residue that is derived using DES CBC encryption method or the like, as outlined before under secure ARP. Host A would verify this message, by doing the same operation on the message and checks the result with MIC1 to see if it is same. Host A then responds by sending a secure S-DHCP_REQUEST message appended with MIC1 and MIC2. MIC2 can be the CBC residue from encrypting S-DHCP_REQUEST. DHCP+ server verifies this and sends a unicast acknowledgment S-DHCP_ACK to host A. ACK contains the timestamp t_s generated by the server to ensure that the message is fresh and is not a replay. S-DHCP_ACK can be optionally encrypted with K$_{SA}$. Only when DHCP+ server issues the ACK (Step 4) that the IP address to client would be confirmed.

Alternate Version 1: Another secure version of the protocol is shown as follows:

1) A $\rightarrow$ALL: Broadcast S-DHCP_DISCOVER + MIC1.
2) DHCP+ $\rightarrow$A: S-DHCP_OFFER + MIC2 (unicast).
3) A $\rightarrow$DHCP+: S-DHCP_REQUEST + MIC3, to all DHCP servers that responded.
4) DHCP+ $\rightarrow$A: (S-DHCP_ACK, NRN) K$_{SA}$ (unicast).

Where, MIC1= H(K$_{SA}$, RN, S-DHCP_ DISCOVER), MIC2 = H(K$_{SA}$, RN, S-DHCP_OFFER) and MIC3 = H(K$_{SA}$, RN, S-DHCP_REQUEST). Explanation of the above protocol is similar to that has been done for secure ARP. In this protocol, we assume that a random number RN is known to both host and the server and is kept secret (generated by A or DHCP+). In Step 1, A broadcasts the request in clear and the MIC (i.e. MIC1). The MIC1 is generated using a collision-free one-way hash function like SHA1 that takes the secret key K$_{SA,}$, the S-DHCP_ DISCOVER and the random number RN as inputs, as listed above. In Step 2, the server uses the S-DHCP_DISCOVER (in plain text), the known random number RN and secret key, K$_{SA}$ to create a similar MIC (say, MIC1*). If MIC1 = MIC1*, then the request is accepted or else it will be rejected. After verifying the integrity of the message, the server sends the response and MIC2 to the host. The MIC2 is generated in the same way and is shown above. Finally in Step 3, Host A will check the integrity of the response as in the above case (to see MIC2 = MIC2*). Host A then sends S-DHCP_REQUEST (in plain) along with MIC3, as in previous Steps. Finally, the server would check the integrity of the message from A and sends an acknowledgement and a new random number (NRN) encrypted by the secret key (K$_{SA}$). NRN can be used in the next request/response exchange.

Table 1: Details of ARP packets in captured files

Session No.	No. of hosts in n/w	No. of total pkts	No. of ARP pkts	Avg. ARP packet size (bytes)	% of ARP pkts
1	48	28366	1326	51.67	4.67
2	45	15539	656	59.15	4.22
3	45	10331	557	59.02	5.39
4	46	15298	650	59.15	4.25
5	48	12511	668	59.24	5.34
6	45	17614	677	59.19	3.84
7	50	11103	646	59.16	5.82
8	48	16909	675	59.22	3.99
9	45	11666	583	59.09	5.00
10	42	11479	562	58.93	4.90

Alternate Version 2: Another more secure version of the protocol is given below.

1) A → ALL: Broadcast S-DHCP_DISCOVER + MIC1.

2) DHCP+ →A: S-DHCP_OFFER + $S_K \oplus$ MIC2 + MIC3.

3) A → DHCP+: S-DHCP_REQUEST+ $S_K \oplus$ MIC4 + MIC5, to all DHCP servers that responded.

4) DHCP+ →A: MIC6.

Where, MIC1 = H(K_{SA}, RN, S-DHCP_DISCOVER), MIC2 = H(K_{SA}, S-DHCP_DISCOVER, S-DHCP_OFFER), MIC3 = H(S_K, NRN), MIC4 = H(K_{SA}, S-DHCP_OFFER, S-DHCP_REQUEST), MIC5 = H(S_K, NRN) and MIC6 = H(S_K, S-DHCP_ACK, NRN). In this protocol, the RN is generated by the server and is also known to host as a secret. In Step 1, A broadcasts S-DHCP_DISCOVER and the MIC1. In Step 2, the server checks the integrity of the message (as shown in the previous protocols), and sends S-DHCP_OFFER, $S_K \oplus$MIC2 and MIC3 to A. MIC2 and MIC3 are generated using the secret key and the session key respectively. MIC2 is XORed with session key, S_K. In Step 3, host A checks the integrity of the message (as shown in the previous protocols), and sends S-DHCP_REQUEST, $S_K \oplus$MIC4 and MIC5 to server. MIC4 and MIC5 are generated using the secret key and the session key respectively. MIC4 is XORed with session key, S_K.In Step 4, the server checks the integrity of the message received and then computes the acknowledgment as shown in MIC6. This acknowledgement calculation involves the timestamp as in previous cases. The NRN (generated by A or DHCP+) is used by the server in MIC3 (which is also contained in MIC5 and MIC6) is kept secret by both parties for the next request/response exchange. It is clear here that even when an attacker knows K_{SA}, he will not be able to send the acknowledgment or MIC4 as he does not know the S_K,used. As in the previous protocol, the attacker cannot also reply an old message (replay attack) since the ACK contains the timestamp when host A generated the message in Step 3.

9 Performance Analysis of ARP and DHCP Protocols

We captured live packet traffic from a wired office network using Ethereal software [13] and filtered all the ARP packets out to do an analysis of ARP packets as in Figure 14. The traffic analysis below shows the percentage of ARP packets found in packet samples collected for around 30 minutes each during 10 sessions. This is shown in Table 1. On the average the network contained around 40 to 50 hosts (clients and servers), including print servers. There were five HP Jet-Direct Print Servers, three Canon Network Print Servers and one D-Link Print Server. The operating system platform was mostly Windows XP on clients, along with Linux server and Netware Server. The percentage of ARP packets was found to be around 4% to 5% of the total traffic.

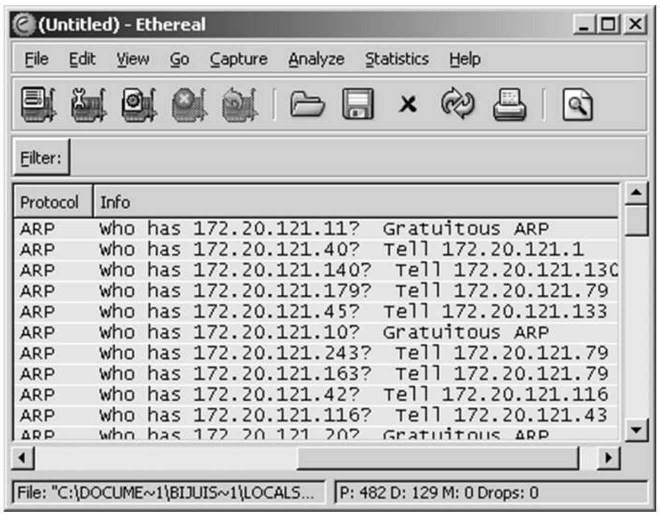
When we calculate S-UARP packet details, we assume that the channel is free of errors and there are no retransmissions required. Also we assume that the ACK is piggybacked every time. The Table 2 shows the Broadcast Packet Reduction because of S-UARP implementation. The S-UARP packet count is done by finding the number of ARP replies and multiplying that by 2.

This is because S-UARP is unicast and hence there would only be 2 packets exchanged (request and reply) between host and server, excluding the ACK packets. So the calculation can be as follows: No. of S-UARP packets (no ACK) = 2 x No. of ARP reply packets. To get the ARP reply packets, we need to use the Ethereal software. Ethereal filter can be enabled with the expression, arp.opcode == 0x2, which is the opcode for ARP reply packet, to get all the ARP reply packets as in Figure 15.

It's quite clear when the number of computers in the network increases the ARP broadcast can still be higher.

Table 2: ARP, S-UARP (no ACK) and SARP comparison

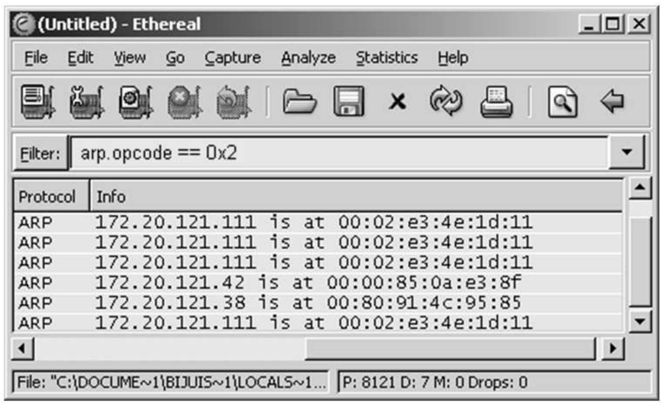
No.	No. of total pkts	No. of ARP pkts	No. of S-UARP pkts [No ACK]	% of ARP pkts	% of S-UARP pkts	% of SARP pkts
1	28366	1326	228	4.67	0.80	7.09
2	15539	656	60	4.22	0.39	5.38
3	10331	557	62	5.39	0.60	7.19
4	15298	650	64	4.25	0.42	5.50
5	12511	668	58	5.34	0.46	6.73
6	17614	677	62	3.84	0.35	4.90
7	11103	646	62	5.82	0.56	7.49
8	16909	675	60	3.99	0.35	5.06
9	11666	583	60	5.00	0.51	6.54
10	11479	562	72	4.90	0.63	6.78
Average Broadcast Packet Reduction in S-UARP (w.r.to ARP) = 09.77 times						



```
Protocol  Info
ARP       Who has 172.20.121.11?  Gratuitous ARP
ARP       Who has 172.20.121.40?  Tell 172.20.121.1
ARP       Who has 172.20.121.140?  Tell 172.20.121.13
ARP       Who has 172.20.121.179?  Tell 172.20.121.79
ARP       Who has 172.20.121.45?  Tell 172.20.121.133
ARP       Who has 172.20.121.10?  Gratuitous ARP
ARP       Who has 172.20.121.243?  Tell 172.20.121.79
ARP       Who has 172.20.121.163?  Tell 172.20.121.79
ARP       Who has 172.20.121.42?  Tell 172.20.121.116
ARP       Who has 172.20.121.116?  Tell 172.20.121.43
ARP       Who has 172.20.121.20?  Gratuitous ARP
```

```
File: "C:\DOCUME~1\BIJUIS~1\LOCALS...   P: 482 D: 129 M: 0 Drops: 0
```

Figure 14: Sample of ARP packet capture using Ethereal.



Filter: arp.opcode == 0x2

```
Protocol  Info
ARP       172.20.121.111 is at 00:02:e3:4e:1d:11
ARP       172.20.121.111 is at 00:02:e3:4e:1d:11
ARP       172.20.121.111 is at 00:02:e3:4e:1d:11
ARP       172.20.121.42 is at 00:00:85:0a:e3:8f
ARP       172.20.121.38 is at 00:80:91:4c:95:85
ARP       172.20.121.111 is at 00:02:e3:4e:1d:11
```

```
File: "C:\DOCUME~1\BIJUIS~1\LOCALS~1...   P: 8121 D: 7 M: 0 Drops: 0
```

Figure 15: Sample of ARP reply packets captured using Ethereal.

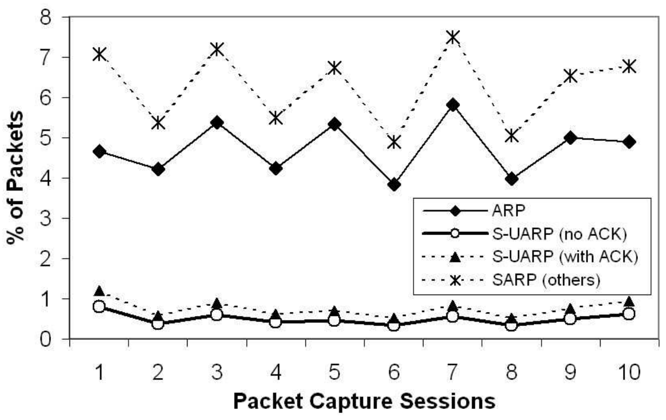


Figure 16: Host channel link utilization (ARP, S-UARP without ACK, S-UARP with ACK and SARP)

The results in Table 2 show that there is a reduction in unwanted broadcast packets by 9.77 times (excluding ACK packets, which is piggybacked). The value is an average of 10 samples. SARP is any other secure ARP scheme that uses PKI infrastructure that needs 4 steps to complete an ARP request cycle. The comparison graph can be as in Figure 16.

The S-UARP channel link utilization with ACK packets is shown in Table 3. Here we assume the worst case of no piggybacking ACK. Thus ACK is sent as a separate packet. Again, we don't consider any retransmission cases here and assume that the channel is free of such errors. The no. of S-UARP packets (with ACK) = 3 x No. of ARP reply packets; As 3 packets are needed to be exchanged here for one cycle – i.e. S-UARP request, S-UARP response and ACK.

There can be a 6.50 times reduction in congestion through S-UARP packets (with ACK) as seen in Table 3. The comparison line graph can be as shown in Figure 16. Assuming that, piggybacking can happen with ACK transmission (host to DHCP+ server) for about 50% time, the Broadcast Packet Reduction can be around 8.13 times (average of previous two cases) than in a normal ARP scenario. This is quite a good result.

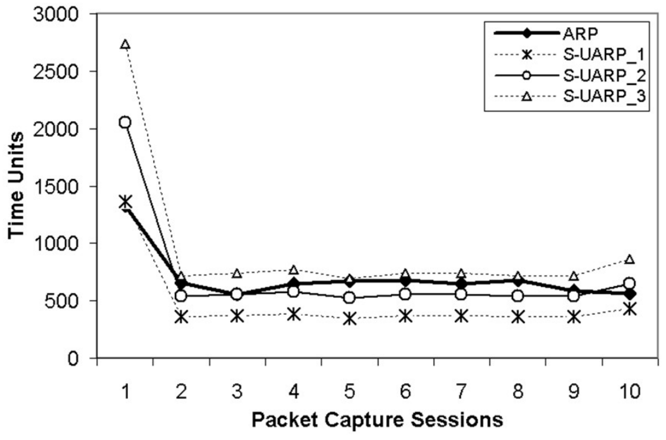


Figure 17: Overall time consumed per session for each of the ARP schemes (ARP vs. three S-UARP versions).

Considering that the encryption operation to carry a factor of 2, compared to a normal operation, the time consumption for the different ARP schemes would be as shown in Figure 17. Here any step that uses encryption (considering the encryption and decryption process) is given double the weight of a normal step without encryption. S-UARP_1, S-UARP_2 and S-UARP_3 are the three proposed schemes with increasing security. It shows SUARP scheme with light encryption is better in time than normal ARP. Figure 18 shows the graph, ignoring encryption performed on ACK.

Since secure DHCP(S-DHCP) uses the same number of steps as its original version, the only overhead encountered would be that of calculating MICs, appending them etc. Like before, a factor of 2 is assigned to all encryption steps. The Figure 19 shows the comparisons, for each of the 10 sessions for a random sample of packets. It is right to infer from the graph that the basic version with minimum security (S-DHCP version 1) has lesser computation overhead and lesser delay.

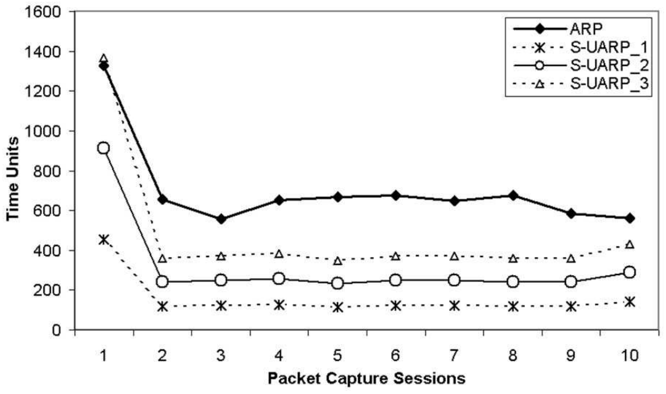


Figure 18: Overall time consumed per session for each of the ARP schemes (ARP vs. three S-UARP versions), ignoring the encryption performed for ACK sent.

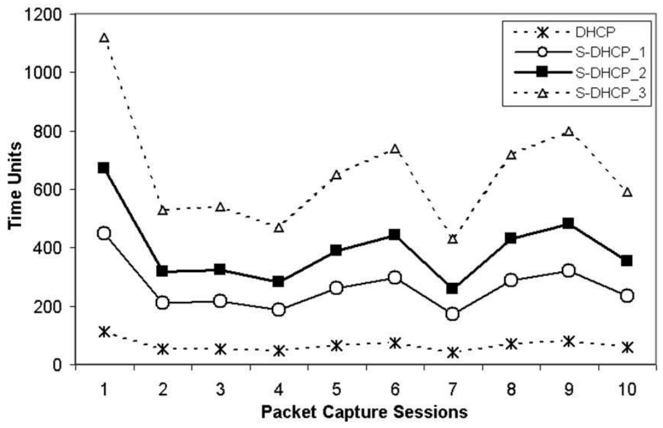


Figure 19: Overall time consumed per session for each of the DHCP schemes (DHCP vs. three S-DHCP versions).

10 Conclusion

Though some initiatives had been there to mitigate ARP poisoning, the new S-UARP protocol (along with secure DHCP) is more efficient in terms of performance and security. It reduces broadcast congestion in network, since the S-UARP request is unicast and directed to only the secure DHCP server. It is quite difficult for an attacker to do ARP poisoning attack, especially on the more secure versions of S-UARP. It is thus protected against message integrity attacks (when ARP packet content can be modified by attacker) and masquerading attacks (when new ARP bogus packet injection can be done by attacker). Also since the DHCP protocol is made secure, the MAC spoofing attacks are also eliminated. The performance analysis of both the protocols are also discussed. This proposal is mostly relevant to IPv4 networks, since ARP is implemented only in IPv4 networks. IPv6 networks use a different mechanism (called Neighbor Discovery Protocol). Nevertheless it is quite relevant until a whole conversion to IPv6 from IPv4 fully happens.

Acknowledgments

This paper is a major extension of the paper titled – "Secure Unicast Address Resolution Protocol (S-UARP) by Extending DHCP" that was presented during ICON 2005, Malaysia.

References

[1] S. Alexander, *DHCP Options and BOOTP Vendor Extensions*, RFC 2132, 1997. (http://rfc.net/rfc2132.html)

Table 3: ARP, S-UARP (with ACK) and SARP comparison

No.	No. of total pkts	No. of ARP pkts	No. of S-UARP pkts [No ACK]	% of ARP pkts	% of S-UARP pkts	% of SARP pkts
1	28366	1326	342	4.67	1.21	7.09
2	15539	656	90	4.22	0.58	5.38
3	10331	557	93	5.39	0.90	7.19
4	15298	650	96	4.25	0.63	5.50
5	12511	668	87	5.34	0.70	6.73
6	17614	677	93	3.84	0.53	4.90
7	11103	646	93	5.82	0.84	7.49
8	16909	675	90	3.99	0.53	5.06
9	11666	583	90	5.00	0.77	6.54
10	11479	562	108	4.90	0.94	6.78
Average Broadcast Packet Reduction in S-UARP (w.r.to ARP) = 6.50 times						

[2] H. Altunbasak, S. Krasser, H. Owen, J. Sokol, and J. Grimminger, "Addressing the weak link between layer 2 and layer 3 in the Internet architecture," *Proceedings of 29th Annual IEEE International Conference on Local Computer Networks (LCN)*, pp. 417-418, Florida, USA, 2004.

[3] D. Bruschi, A. Ornaghi, and E. Rosti, "S-ARP: A secure address resolution protocol," *19th Annual Computer Security Applications Conference*, pp. 66-74, Nevada, USA, 2003.

[4] R. Droms, *Dynamic Host Configuration Protocol*, RFC 2131, 1997. (http://www.faqs.org/rfcs /rfc2131.html)

[5] T. Komori, and T. Saito, "The secure DHCP system with user authentication," *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN)*, pp. 123-131, Florida, USA, 2002.

[6] C. M. Kozierok, *The TCP/IP Guide Website*, 2005. (http://www.tcpipguide.com/free/ t_ARPMessageFormat.htm)

[7] C. Nachreiner, *Anatomy of an ARP Poisoning Attack*, Washington, USA, 2003. (http://www. watchguard.com/infocenter/editorial/135324.asp)

[8] L. L. Peterson, and B. S. Davie, *Computer Networks – A systems approach*, 3rd Edition, pp. 583-601, Morgan Kaufmann (Elsevier), 2003.

[9] D. C. Plummer, *ARP Protocol*, RFC 826, 1982. (http://www.faqs.org/rfcs/rfc826.html)

[10] A. S. Tanenbaum, *Computer Networks*, 4th Edition, Prentice Hall PTR, pp.450-454, 2003.

[11] *The Ethereal Software*, Version 0.99.0. (http://www. ethereal.com)

[12] *The Cain & Abel Software*, Version 2.5. (http:// www.nwcet.org/downloads/cainAbel.pdf)

[13] *The Mac Makeup Software*, Version 1.71d. (http://www.gorlani.com/publicprj/MacMakeUp/ macmakeup.asp)

[14] M. V. Tripunitara, and P. Dutta. "A middleware approach to asynchronous and backward compatible detection and prevention of ARP cache poisoning," *Proceedings of the 15th Annual Computer Security Application Conference (ACSAC)*, pp. 303-309, 1999.

[15] J. Wright, *Detecting Wireless LAN MAC Address Spoofing*, Johnson and Wales University, GCIH, CCNA, 2003. (http://home.jwu.edu/jwright/ papers/wlan-mac-spoof.pdf)

Biju Issac is a lecturer in Information Technology at Swinburne University of Technology (Sarawak Campus), Malaysia. He is holding a BEng (Electronics and Communication Engineering) degree along with an MCA (Master of Computer Application) with Honours from Calicut University, India. He is an IEEE, IEEE Communication Society and IEEE Education Society member. His research interests are mainly in computer networks and education. Specifically, his research interest is in mobility management, wireless and network security, education and e-learning. He is heading the network security research in Information and Security Research (iSECURES) Lab in Swinburne. He has a number of refereed publications that includes many conference papers, journal papers and book chapters.