Enhanced ARP: Preventing ARP Poisoning-based Man-in-the-Middle Attacks

Seung Yeob Nam, *Member*, *IEEE*, Dongwon Kim, and Jeongeun Kim

***Abstract*—In this letter, an enhanced version of Address Resolution Protocol (ARP) is proposed to prevent ARP poisoning-based Man-in-the-Middle (MITM) attacks. The proposed mechanism is based on the following concept. When a node knows the correct MAC address for a given IP address, if it retains the mapping while that machine is alive, then MITM attack is impossible for that IP address. In order to prevent MITM attacks even for a new IP address, a voting-based resolution mechanism is proposed. The proposed scheme is backward compatible with existing ARP and incrementally deployable.**

***Index Terms*—ARP cache poisoning, Man-in-the-Middle attack, ARP poisoning prevention, voting.**

I. Introduction

The Address Resolution Protocol (ARP) resolves IP addresses into hardware or MAC (Medium Access Control) addresses. The ARP poisoning attack targets to modify the IP/MAC address mapping in the ARP cache of a remote machine maliciously. This ARP poisoning is usually used to mount other types of attacks such as DoS or MITM attacks.

Several attempts have been made to resolve the ARP cache poisoning problem. Dynamic ARP Inspection (DAI) performed on Ethernet switches [1] might prevent ARP poisoning, but this requires manual configuration by network managers and the network portion covered by the Ethernet switches incapable of DAI cannot be protected. The approaches that do not require support from Ethernet switches can be classified into two categories based on the use of cryptography. Antidote [2] is a non-cryptographic approach, which uses a similar idea to ours, especially querying the previous MAC address in case of MAC conflict. However, Antidote cannot prevent poisoning for a new IP address if a malicious ARP reply arrives first [3].

S-ARP [3] and Ticket-based ARP (TARP) [4] are two well-known cryptography-based approaches. S-ARP may have a high computational cost [4] and the central servers, such as Authoritative Key Distributor (AKD) for S-ARP and Local Ticket Agent (LTA) for TARP, might be subject to a single point of failure problem. In addition, they usually require the upgrade of the DHCP server and incremental deployment is not easy. For example, TARP-enabled or S-ARP-enabled machines may not accept ARP replies from non-TARP/S-ARP nodes.

We investigate a new mechanism to prevent ARP poisoning-based MITM attacks while overcoming the limitations of existing approaches. Since we do not use cryptographic mechanisms and central servers, there are no complexity issues and a single point of failure problem. We incorporate two new concepts, long-term memory and voting, with the existing ARP to resolve the problem, while satisfying the following requirements: backward compatibility with existing ARP, minimal infrastructure upgrade cost (e.g. no upgrade of Ethernet switches or modification of DHCP), and incremental deployability. The proposed mechanism is evaluated through experiments.

The authors are with the Department of Information and Communication Engineering, Yeungnam University, Gyeongsan-si, Gyeongbuk, 712-749 Korea (e-mail: synam@ynu.ac.kr).

II. MITM-Resistant Address Resolution Protocol

The proposed MITM-Resistant Address Resolution Protocol, which is called *MR-ARP*, is based on the following concept. When Node A knows the correct IP/MAC address mapping for Node B, if Node A retains the mapping while Node B is alive, then ARP poisoning and the MITM attack between A and B are impossible.

MR-ARP employs a long-term IP/MAC mapping table, as well as the ARP cache used in existing ARP to retain IP/MAC mapping for alive machines over longer periods. Three fields, IP, MAC, and Timer T_L, are allocated to each IP address registered in the long-term table. The default value of the timer in the long-term table is 60 minutes. In order to avoid losing the mapping of ($\mathrm{IP_a}$, $\mathrm{MAC_a}$) for an alive host after 60 minutes, we send new ARP request messages for $\mathrm{IP_a}$ only to $\mathrm{MAC_a}$ through unicasting to check if the $\mathrm{MAC_a}$ is alive. In this case, 50 ARP request messages are sent at random intervals with an average of 10 msec. If at least one ARP reply is returned, then the mapping is registered in the short-term ARP cache and the corresponding long-term table timer is set to 60 minutes again. If no ARP reply returns, then the mapping of ($\mathrm{IP_a}$, $\mathrm{MAC_a}$) is considered invalid and the corresponding entry is deleted from the long-term table. Thus, the IP/MAC mapping can be retained in the long-term table until the binding is released.

MR-ARP attempts to manage the IP/MAC mappings for all alive machines in the same LAN through the long-term table. This goal can be easily achieved if we fill the long-term table based on the received ARP request messages, especially the source IP and MAC address portions, since alive machines tend to send ARP requests to find the MAC address of the gateway router repeatedly because of the timer expiry in the ARP cache. However, IP/MAC mapping of every ARP request cannot be directly reflected because of the possibility of ARP cache poisoning attempts. Thus, the short-term cache and the long-term table need to be updated carefully when ARP request packets arrive. Fig. 1 shows the detailed management policy. Although Fig. 1 shows how to resolve MAC conflicts induced by ARP requests, the same mechanisms apply when the conflicts arise from ARP reply messages. By the rule for case (A) in Fig. 1, each IP/MAC mapping registered in the short-term cache is

```
/* (IP_a, MAC_a): the source IP and MAC addresses of the received
                  ARP request packet */

if(IP_a is registered in the short-term cache){ — (A)
  no action; /* in case of conflict, preserve existing mapping.*/
}
else if((IP_a, MAC_a) is registered in the long-term table){
  register (IP_a, MAC_a) in the short-term cache;
  set the long-term table timer to 60 minutes;
}
else if(IP_a is in the long-term table, but the registered MAC
      is not MAC_a){ — (B)
  /* conflict on IP and MAC mapping */
  send 50 ARP requests to existing MAC through unicasting
    at random intervals with an average of 10 msec;
  if(at least one ARP reply arrives)
    retain the existing (IP, MAC) mapping and drop the new one;
  else
    accept the new mapping;
  The accepted mapping is registered in the short-term cache, too.
}
else{ /*i.e. IP_a is not in short-term/long-term tables */ — (C)
  send voting requests for IP_a;
  if(no response)
    the mapping (IP_a, MAC_a) is registered in both tables;
  else if(there exists a MAC that polls over 50% of votes for IP_a)
    that mapping is registered in both tables;
}
```

Fig. 1. Short-term cache and long-term table update policy applied on the arrival of ARP request packets

frozen until it expires, e.g. for 2 minutes for Windows XP, to prevent too frequent cache updates by ARP request sniffing.

In case (B) of Fig. 1, MAC conflict occurs because the newly received MAC address $\mathrm{MAC_a}$ for $\mathrm{IP_a}$ is different from $\mathrm{MAC'_a}$ that is already associated with $\mathrm{IP_a}$. The conflict is resolved by giving a priority to $\mathrm{MAC'_a}$ only if it is alive. As shown in Fig. 1, the activity of a host is examined by sending 50 ARP request packets and counting the ARP replies. Multiple ARP requests are sent to cope with unexpected packet losses including the case of DoS attack on $\mathrm{MAC'_a}$. Even though the packet loss probability is as high as 90% by DoS attack, at least one ARP reply will be returned with a probability of $99.5\%(=1-0.9^{50})$.

A. Voting-based Conflict Resolution

Thus far, we investigated how to prevent MITM attacks for the nodes whose IP/MAC mapping is known already. However, if Node A receives an ARP request from a new IP address, then Node A cannot easily judge the correctness of the source IP and MAC address mapping contained in the ARP request. For example, when a new machine is added in a LAN with no IP/MAC mapping information and the machine sends an ARP request for the gateway router, if an adversary's ARP reply arrives first, the ARP cache can be poisoned. In order to solve this poisoning problem, we propose a voting-based resolution mechanism which corresponds to case (C) in Fig. 1.

The basic concept of the voting-based resolution mechanism is as follows. When Node A is turned on with empty ARP cache and long-term table, if multiple neighbor hosts inform Node A of the true MAC address of the gateway router that they know, then gateway MAC poisoning can be prevented.

We investigate the details of the voting-based resolution mechanism. Two more ARP packet types are defined for MR-ARP: voting request and voting reply packets. However, they reuse the packet format of ARP request/reply packets. The *operation* field is set to 20 and 21 for voting request and reply packets, respectively. If Node A observes an ARP request from a new IP address $\mathrm{IP_B}$ with the MAC address of $\mathrm{MAC_B}$, then Node A broadcasts a voting request with $\mathrm{IP_B}$ in the *target protocol address* field to collect IP/MAC mapping for that IP from other hosts after waiting a random time of between 0 and 100 msec. The random waiting time is employed to prevent a simultaneous ARP voting request/reply storm. If a voting-cognizant host receives an ARP voting request for $\mathrm{IP_B}$, then it sends back 50 ARP voting replies with IP/MAC mapping for $\mathrm{IP_B}$ at the maximum rate without delay when it knows the mapping. Then, Node A calculates the polling score for each received MAC address based on early N replies. If a MAC exists that received over $0.5N$ votes, then that MAC address is accepted for $\mathrm{IP_B}$. In order to avoid the bias by the machines with small RTT, we start counting N after waiting at least RTT of the machine with the largest RTT in the LAN. Currently, the waiting time before counting N is set to 0.3 msec.

When a new MR-ARP-enabled machine is deployed in some LAN, if there are no other MR-ARP-enabled machines, then this new machine cannot benefit from the voting mechanism. However, we can show that the new MR-ARP-enabled machine can be additionally protected by voting, if at least two MR-ARP-enabled machines exist when there is one attacker. Let us consider a case where k MR-ARP-enabled machines $\mathrm{MAC_1}, \ldots, \mathrm{MAC_k}$ and one adversary $\mathrm{MAC_v}$ are interconnected by the same Ethernet switch. When a new MR-ARP-enabled machine $\mathrm{MAC_e}$ is attached to the same switch, if $\mathrm{MAC_e}$ receives an ARP request with a false IP/MAC mapping from the adversary, then $\mathrm{MAC_e}$ will broadcast the ARP voting request. Let r_i denote the ARP voting reply traffic rate of $\mathrm{MAC_i}$ for $i = 1, 2, \ldots, k$, and r_v denotes the ARP voting reply rate of the adversary. If $\mathrm{MAC_e}$ observes the voting replies during an interval of length I, then the average ratio of voting replies from the adversary becomes $r_v/(r_v + \sum_{i=1,2,\ldots,k} r_i)$ under the assumption that the Ethernet switch serves input buffers fairly. If the voting reply rate is the same for every machine, then the ratio becomes $1/(k+1)$. Since recent machines can send traffic up to near the link rate, the effect of r_v on the ratio of the votes for the adversary is bound to be limited.

We now investigate how large N should be to prevent ARP poisoning, when there are two extant MR-ARP-enabled machines and one adversary node. In this case, the reply ratio from the adversary will be close to 1/3 by the previous reasoning. Let us assume that each packet arrival is independent of other arrivals, and the probability that each packet arrival is from the adversary node is p. X is a random variable that represents the total number of packets from the adversary among N voting reply packets, then X has a binomial distribution, i.e. $X \sim Binomial(N, p)$. We can obtain the following inequality using Chernoff bounds [5, Corollary 3.1.2]:

$$Pr(X/N > \eta) \leq \exp\{-2N(\eta - p)^2\}. \quad (1)$$

Since X/N is the ratio of the adversary's replies, if we set η to the decision threshold 0.5, then (1) gives an upper bound on the false negative probability for the adversary. When $p = 1/3$, if

we set N to 120, then the upper bound becomes 0.0013. Thus, the value of 120 for N gives a low false negative probability.

Let us investigate the traffic overhead of the voting-based resolution mechanism. L and M represent the total number of alive machines and the total number of alive MR-ARP-enabled machines in the LAN. To simplify the analysis, we assume that IP addresses are allocated through DHCP for all nodes, and each IP lease time is exponentially distributed with the same average T_D. MR-ARP-enabled Node A performs voting-based resolution for an alive machine with an IP address $\mathrm{IP_B}$ only once during its own IP lease time. This occurs when Node A comes up with a newly allocated IP address. When $\mathrm{IP_B}$ is released and reallocated, the new mapping is resolved, by the rule for case (B) in Fig.1, without voting. Thus, the average voting reply rate for IP address $\mathrm{IP_B}$ to Node A can be calculated as $(M-1) \times 50 \times 28 \times 8/T_D = 11.2(M-1)/T_D$ Kbps. Thus, the aggregate voting reply rate to Node A is $11.2L(M-1)/T_D$ Kbps. If $M = L$, the rate becomes $11.2L(L-1)/T_D$ Kbps. If we assume that T_D is one day and $L = 255$, then the average voting reply rate to Node A is about 8.4 Kbps. Thus, the voting traffic overhead is not significant for a small subnet.

III. Performance Analysis

We first investigate whether a host under DoS attack can respond to ARP requests to check the feasibility of the mechanism corresponding to case (B) in Fig. 1. We measure the response probability of a victim host under DoS attack for various numbers of ARP requests in a 100 Mbps LAN, and even a single packet response for multiple ARP requests is considered as a successful response of the victim. We used several DoS attack patterns: SYN flooding, UDP flooding, ICMP flooding, and ICMP smurf attack, and the lowest response probability was obtained from smurf attack because more machines have been involved in the attack than for other DoS attack types. The number of involved attack nodes is 25 for smurf attack. Table I shows the result corresponding to smurf attack. We find that the response probability of 99% is achieved if at least 20 ARP packets are sent. Thus, the algorithm for case (B) in Fig. 1 can work reliably for the selected DoS attack patterns. If a victim node is disabled by another type of DoS attack in the worst case, then the MITM attack cannot be valid by the definition of MITM. However, when the victim recovers, it might be subject to the MITM attack. In this case, if the victim sends a voting request for its own IP address under the assumption that there are a sufficient number of MR-ARP-enabled nodes, then the victim can easily know whether its own IP address is used by another machine based on the voting results and avoid MITM attack by stopping the use of the intercepted IP address.

We next evaluate the voting-based resolution mechanism in a test-bed where six MR-ARP-enabled machines, which are 2.66 GHz Dual-core PCs, and several colluding adversary machines, which are 2.66 GHz Quad-core PCs, are interconnected by a Gigabit Ethernet switch. We implemented the proposed MR-ARP mechanism on Fedora 9 Linux (kernel 2.6.25) by modifying the ARP module code. False decision is made when the aggregate number of votes from the adversaries exceeds $N/2$. Fig. 2 shows the theoretical values and the measured values of the false decision probability. Each measurement value is obtained from 200 experiments. The theoretical values are obtained by approximating the binomial distribution of X in (1) by Gaussian distribution $\mathrm{N}(Np, Np(1-p))$. We observe that the measured false decision ratios are always zero differently from the theoretical results. When we derive the binomial distribution for X, we assume that the outcome of each voting is independent from other outcomes. However, we found that the votes from different nodes are arriving in an approximately round-robin manner. Because of a rather deterministic pattern of voting outcomes, the false decision ratios are measured to be nearly zero when the MR-ARP-enabled nodes outnumber the adversary nodes.

TABLE I

ARP response probability for various numbers of ARP requests under ICMP smurf attack

# of ARP requests	1	5	10	15	20	25
response prob. (%)	24.7	68.4	92.2	97.6	99.6	100

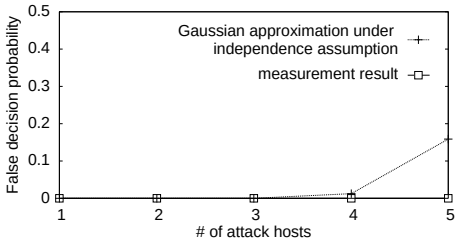

Fig. 2. False decision probability for various numbers of attack hosts when the number of MR-ARP-enabled hosts is 6

MR-ARP-enabled Node A responds to an ARP request destined to itself by sending an ARP reply message as current ARP. Even though Node A is the only MR-ARP-enabled node in the LAN, Node A can accept the received IP/MAC mapping for a new IP address using the rule for (C) in Fig. 1. Thus, MR-ARP is backward compatible with existing ARP.

IV. Conclusions

A new mechanism to prevent ARP poisoning-based MITM attacks is proposed based on two key concepts: long-term IP/MAC mapping table and voting. Even though the proposed scheme is installed on a small number of hosts, they can be well protected through voting-based collaboration. Since the proposed scheme does not use cryptography and central servers, it does not have complexity and single point of failure problems while achieving backward compatibility with existing ARP.

References

[1] Y. Bhaiji, Network Security Technologies and Solutions, Cisco Press, 2008.
[2] I. Teterin, Antidote, http://online.securityfocus.com/archive/1/299929.
[3] D. Bruschi, A. Ornaghi, and E. Rosti, "S-ARP: a Secure Address Resolution Protocol," *in Proc. of Annual Computer Security Applications Conference (ACSAC)*, 2003.
[4] W. Lootah, W. Enck, and P. McDaniel, "TARP: Ticket-based address resolution protocol," *Computer Networks*, vol. 51, pp. 4322-4337, Oct. 2007.
[5] S. M. Ross, Probability Models for Computer Science, Harcourt/Academic Press, 2002.