An ARP-disabled network system for neutralizing ARP-based attack

Davaadorj Battulga*, Rhong-Ho Jang*, Dae-Hun Nyang*
*Dept. of Computer Science, Inha University

Abstract

Address Resolution Protocol (ARP) is used for mapping a network address to physical address in many network technologies. However, since ARP protocol has no security feature, it always abused by attackers for performing ARP-based attacks. Researchers presented many technologies to improve ARP protocol, but most of them require a high implementation cost or scarify the network performance for using ARP protocol securely. In this paper, we present an ARP-disabled network system to neutralize the ARP-based attacks. "ARP-disabled" means suppress the ARP messages like request, response and broadcast messages, but not the ARP table. In our system, ARP tables are used for managing static ARP entries without prior knowledge (e.g. IP, MAC list of client devices). This is possible because the MAC address was designed to be derived from IP address. In general, our system is safe from the ARP-based attacks even the attacker has a strong power. Moreover, we saved network bandwidth by disabling the ARP messages.

1. Introduction

Address resolution protocol (ARP) is a data link layer protocol, which resolves logical address (IP) to its physical address (MAC). In order to successfully communicate within the local network, the host must have the ARP cache, which maps other hosts' IP addresses to their corresponding MAC addresses. ARP cache is updated by ARP request and response messages.

Since ARP is a stateless protocol, any attacker can modify ARP mapping. This is one of the best known and malicious attacks called ARP cache poisoning, or ARP spoofing. After feeding the hosts ARP cache table with fake information, attacker can perform Man-in-the-Middle (MITM), Denial of Service (DOS), and such attacks. Due to its importance, several types of detection, prevention techniques were proposed: Secure ARP [1], Ticket based ARP [2], Probe based technique [3], Enhanced ARP [4] etc. But most of these approaches require a high implementation cost or scarify the network performance.

In this paper, we present a method that successfully prevent any ARP based attacks through an economic, efficient, and easily implementable scheme. For neutralizing the ARP-based attack, we define static ARP entries in ARP table of all sub-group IPs. At the same time, we disabled all ARP messages in both server and client sides so that attackers have no chance to update fake ARP caches. Our system does not require prior knowledge (e.g. IP, MAC List) for managing static ARP entries, because the MAC address for constructing the ARP entry was designed to be derived from IP address. Our contribution: 1. Our system is safe from ARP-based attacks, since we use static ARP entries and the ARP message is disabled. 2. The performance of network can be improved because the bandwidth occupied by ARP messages is released. 3. Our system does not require any prior knowledge (e.g. IP, MAC List).

2. Related Work

Number of methods proposed for preventing and detecting ARP spoofing. Those methods can be divided in two categories: cryptographic method and non-cryptographic method.

Cryptographic method modifies standard ARP, and interferes with network layering architecture. Due to encryption, and decryption process, the processing time of this method is always higher compared to non-cryptographic methods. This slows down ARP by 17 percent or more.

Non-cryptographic methods are the ideal approaches for ARP. The processing time is less than cryptographic methods, also doesn't interfere with standard network layering architecture.

Secure ARP [1] (SARP) is cryptographic method which any SARP enabled network has its public/private key pair. Thus Authoritative Key Distributor (AKD) is connected to the network to get the public key, and hosts are authenticated by its signature. This method uses extra devices on network, and have high computational cost compared to others.

Ticket based ARP [2] (TARP) distributes its network from centrally generated IP, and MAC mapping called ticket, and signed by Local Ticket Agent (LTA). Client attaches this ticket to ARP replies so the server can verify the valid address. This method requires upgrade of DHCP server, and deployment is not easy.

Probe based approach [3] sends two ICMP request messages instead of one. Legitimate host generate reply for only first request, while attacker generate reply for second. This method seems vulnerable if attacker take advantage of

correct protocol stack.

Enhanced ARP [4] uses voting-based technique which gathers information about legitimate client from multiple neighbors. This method increases traffic depend on number of neighbors.

Network Intrusion Detection and Prevention System [5] (NIDPS) uses agents collecting IP, and MAC mapping from clients. This mapping used as static ARP entries to correct any wrong mapping detected. However, agents aren't authenticated, also increases network traffic inside LAN segment.

An Automated Approach using Static ARP [6] method uses modified register messages sent from server to all clients. This method seems ideal in terms of reducing false positives, but network discovery is still possible. Which leads vulnerable to modified ARP requests.

There are also comparative studies for various techniques [7], and proposed ideal requirements that preventing ARP spoofing [8].

3. Threat Model

ARP is used for map an IP address to given MAC address, so that packets can be transmitted across LAN. ARP messages are exchanged when one host knows the IP address of remote host, but not aware of MAC address. Therefore, host sends multicast ARP request message to every host on network, asking the matching MAC address. Other host announcing the IP address as its own, and replies with unicast ARP response message to requested host.

The IP, and MAC address mapping is saved in ARP cache table. ARP cache is first checked before sending ARP request. ARP cache entries can be static or dynamic. Static entries are manually configured, and stays until system restart. Dynamic entries are updated through ARP request, and reply messages, and expires after short time if not referenced.

This enables ARP spoofing possible. ARP spoofing, or ARP cache poisoning, is a technique which attacker sends spoofed ARP messages to target host, by inserting fake IP and MAC mapping entry in targets ARP cache.

Figure 1, and 2 shows a scenario of ARP spoofing attack. First, the attacker prepares fake ARP reply messages for both gateway server and clients. When the gateway server sends a ARP request message for asking the MAC address of $IP_{PC:2}$, the attacker sends back a fake ARP response message to the gateway server to announce that the MAC address of $IP_{PC:2}$ is $MAC_{Attacker}$. Also the attacker does the same progress when PC:2 asking the MAC address of gateway server to announce that the MAC address of $IP_{Gateway}$ is $MAC_{Attacker}$ (Figure 1). Then, the gateway server and clients will update its ARP cache as shown in tables of Figure 2. The attacker can relay the data packets between the gateway server and PC:2 to capture more data packets for performing MITM attack.

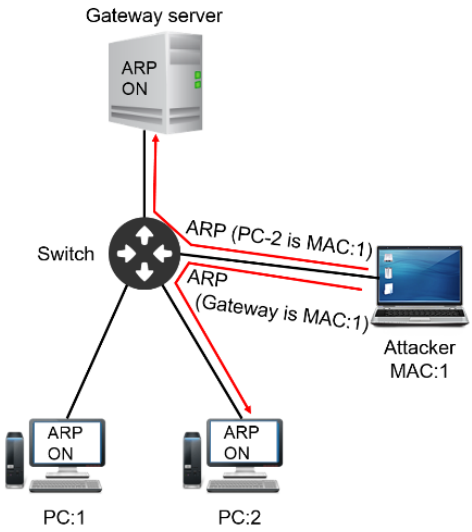

Figure 1 The Attacker sends fake ARP response messages to both gateway server and PC:2.

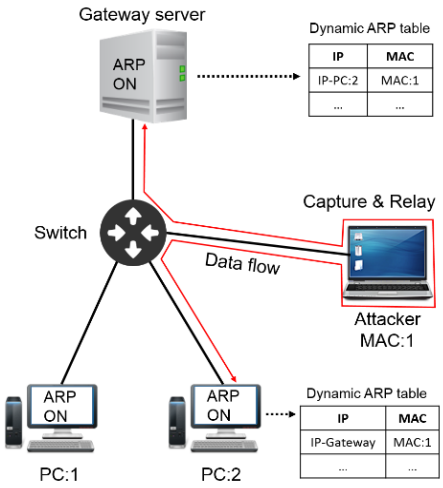

Figure 2 The gateway server and PC:2 update wrong caches due to fake response messages from the attacker.

4. System Model

Figure 3 shows the architecture of our system model. Our system focus on the managed group networks which network administrator associates fixed IP address for clients (e.g. government, enterprise, hospital, school). Also this method does not require any additional hardware and only needs lightweight implement cost for both server and client side.

ARP disable: As shown in previous section, since ARP function has no security feature, which can be abused by attackers to perform ARP spoofing attack. Thus, the reliable way for preventing is turning off the ARP function. Therefore, in our system, we disabled the ARP function when the gateway server initializes. We note that, ARP functions that

disabled in our system are ARP request, response and broadcast messages, but not ARP table. In most Linux system, ARP tables still work when ARP function is disabled.

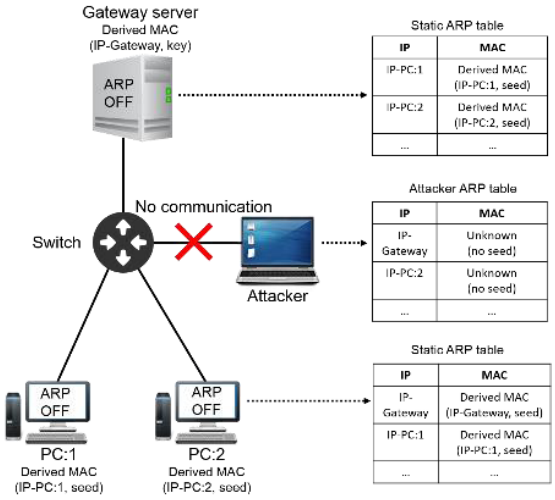

Figure 3 Architecture of the ARP-disabled network system

Seed-based MAC address derive function: Since we disabled the ARP message, the attacker cannot perform ARP based attacks anymore. But also clients also cannot get network connection, because there is no way for the gateway and clients to get the MAC address of gateway server.

For solving this problem our system employed a seed-based MAC address derive function which can derive MAC address from IP address.

$$MAC_{derived} = D(IP\ address,\ seed) \qquad (1)$$

Simply, function D appends 16-bit $seed$ to 32-bit IP address to generate 48-bit MAC address. We note that, the $seed$ is generated by the network administrator, and is given to every client. Server-side setup processes are as following:

Step 1. Turn off the ARP function.

Step 2. Update the physical MAC address of the local interface with $D(IP_{gateway},\ seed)$.

Step 3. Add ARP entry for each IP_{sub} in sub-network group to ARP cache table with IP and $D(IP_{sub},\ seed)$. pairs.

Since $Seed$ and $IP_{gateway}$ are public information, Clients can communicate with the gateway by adding static entry to the ARP cache table. Client-side setup processes are as following:

Step 1. Turn off the ARP function.

Step 2. Update the physical MAC address of the local interface of client PC with $D(seed,\ IP_{client})$.

Step 3. Add ARP entry of gateway server to ARP cache table with $IP_{gateway}$ and $D(seed,\ IP_{gateway})$ pair.

Since ARP cache static entries are possible, the network system works properly after setting up, even the ARP functions are disabled.

5. Experiment Setup

For disabling ARP functions in gateway server side, we constructed our gateway server in an AP hardware TP-Link AC1750 ver.2, which ran OpenWrt [9] (15.05, Chaos Calmer). For the client, we use a desktop equipped with an Intel Core i7-4770 CPU, 32GB RAM which ran Linux Ubuntu 14.04 (kernel ver.3.18.0-20-generic), and ran up to 8 Ubuntu Linux virtual machines configured as 1 GB RAM and bridged network settings. All operating systems provide a toolkit using IP [10] command to disable the ARP function well.

6. Discussion

In this section, we will discuss about security of our system, and the reason of using Seed-based MAC address derive function. In the end of this section, we show the bandwidth saving effect achieved from disabling ARP function through experiments.

1. Security

There are two kinds of attackers we can imagine: weak and strong attacker.

Weak attacker. If the attackers ARP function is on, and it attempts normal ARP poisoning attack, the attacker can't acquire any information about either router, or clients. In this case network discovery is completely blocked. In the gateway server, the client who sends ARP messages can be determined as an attacker.

Strong attacker. Assume the $seed$ and D is known by the attacker. And the attacker knows one of the free IP address. Even the attacker sends fake messages to the gateway server, and clients using $IP_{gateway}$ and $D(IP_{free},\ seed)$ pair, those fake messages will be ignored by both gateway server and clients, because ARP messages are disabled. In other word, the attacker has no chance to update ARP caches in both gateway server and clients.

For testing the security of our system, we perform 10 ARP attack in both ARP enabled system and ours. The ARP attack was constructed in Kali Linux (kernel ver.4.3.0) using ARPspoof [11] and sslstrip [12]. As a result, In ARP enabled system, the attacker successfully performed MITM attack for 10 times. And in our system, the attacker never succeeded.

2. Using seed-based MAC address derive function.

In a general way, preventing the ARP-based attack can be simply achieved by only using static ARP entries in both server and client side. However, it requires the network

administrator's awareness and management for IP and MAC address of all client's PC and provide them to every client for adding static ARP entries in tables. Therefore, there are problems about not only wasting human resources, but also privacy. By using seed-based MAC address derive function, these problem can be solved simply, since MAC addresses can be derived from IP address with $seed$ and function D. And our system is still safe even the attacker knows $seed$ and function D.

3. Bandwidth saving effect

Since we disabled ARP functions in our system, there are no ARP messages anymore in our network environment as shown in Figure 5. Then, Figure 4 shows 2 minutes' user traffic in ARP-enabled environment. In this experiment, a user was asked to use the network server without any constraint. As a result, there are 77 ARP messages generated by one user which is 8.1% of total traffics by sending active ARP requests. If there are many users active at the same time, ARP messages can be a factor to effluent the network speed. Thus, in our system, there is no concern about ARP messages caused network delay.

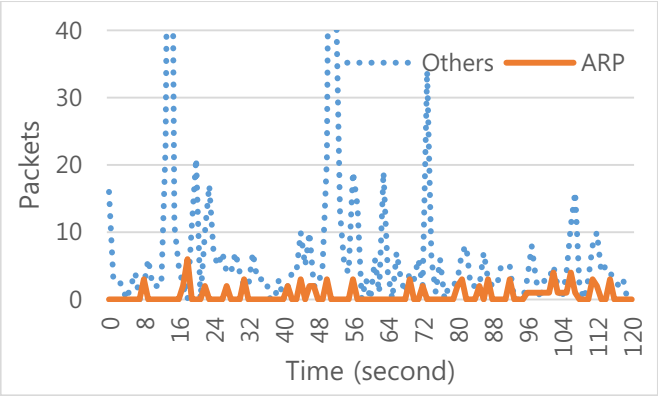

Figure 4 2 Minutes user traffics in ARP enabled environment

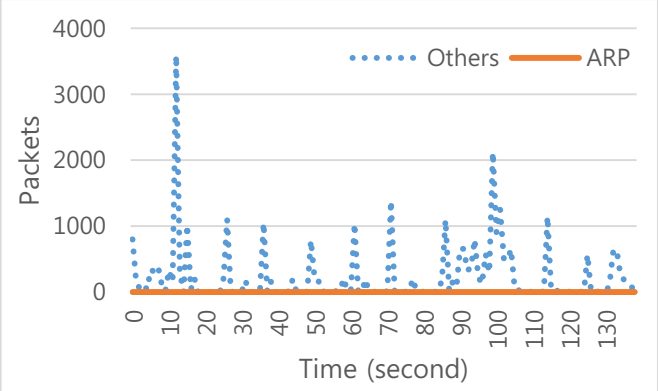

Figure 5 2 Minutes user traffics in ARP disabled environment

7. Conclusion and Future Work

In this paper, we presented an ARP-disabled network system which provides a strong security feature for preventing the ARP-based attacks. In the same time, our system does not require any additional hardware, also lightweight implement cost for both server and client sides.

Through several experiments, we proved feasibility and security of our system. For the future work, we would like to consist our environment with more client devices to prove scalability of our system.

References

[1] D.Bruschi, A.Ornaghi, E.Rosti, S-ARP: a Secure Address Resolution Protocol, 19th Annual Computer security Application Conference (ACSAC), 2003

[2] Wesam Lootah, William Enck, Patrick McDaniel, TARP: Ticket-based address resolution protocol, ScienceDirect, Computer Networks 51:4322-4337, 2007

[3] Poonam Pandey, Prevention of ARP spoofing: A Probe Packet based Technique, Advance Computing Conference (IACC), IEEE 3rd International, 2013

[4] Seung Yeob Nam, DongWon Kim, Jeongeun Kim, Enhanced ARP: Preventing ARP poisoning-based Man-in-the-middle attacks. IEEE Communications Letters (ICL) 14 (2):186-189, 2010

[5] Dr. S.G.Bhirud, Vijay Katkar, Light Weight Approach for IP-ARP Spoofing Detection and Prevention, Second Asian Himalayas International Conference on Internet (AU-ICI), pages:1-5, 2011

[6] Ahmed M.Abdelsalam, Wail S.Elkilani, Khalid M.Amin, An Automated approach for Preventing ARP spoofing Attack using Static ARP Entries, International Journal of Advanced Computer Science and Applocations (IJACSA), vol.5, No.1, 2014

[7] Zouheir Trabelsi, Wassim El-Hajj, ARP spoofing: a comparative study for education purposes, Information Security Cirriculum Development Conference, InfoSecCD09, 2009

[8] Cristina Abad, Rafael I.Bonilla, An Analysis on the Schemes for Detecting and Preventing ARP Cache Poisoning Attacks, Distribushed Computing Systems Workshops, ICDCSW`07, 2007

[9] https://openwrt.org/

[10] https://linux.die.net/man/8/ip

[11] http://su2.info/doc/arpspoof.php

[12] http://tools.kali.org/information-gathering/sslstrip