Journal of Information Hiding and Multimedia Signal Processing ©2015 ISSN 2073-4212
Ubiquitous International Volume **6**, Number **5**, September **2015**

A Novel Address Resolution Process Based on The Seek Secret Man Protocol

Guang-Jia Song and Zhen-Zhou Ji

School of Computer Science and Technology
Harbin Institute of Technology
Harbin 150001, P.R. China
songguangjia@sohu.com; jizhenzhou@hit.edu.cn

Received May, 2015; revised July, 2015

Abstract. *Address resolution is an important component of network communication. It determines the correspondence relationship between the IP and MAC of the host. Given the public destination of address resolutions, existing address resolution protocols are vulnerable to deception attacks, such as man-in-the-middle or DoS. Thus, a new security protocol called seek secret man (SSM) is proposed to address this drawback. Further, we utilize the SSM protocol as prototype and design a secure address resolution process called AR-SSM. This technique uses encryption technology to hide the destination of the address resolution, thereby preventing an attacker from launching a pointed attack. The experimental and comparison results show that in the presence of malicious nodes, the address cache pollution rate of AR-SSM is far lower than that of the traditional address resolution process.*

Keywords: Network security; Address resolution; Neighbor discovery; Seek secret man; Security protocol.

1. Introduction.

Address resolution protocols (ARPs) play an important role in the hierarchical network architecture. With layering design, each layer of the network only needs to pay attention to its related problems, such as how to communicate with the peer layer or how to provide service for the upper layer. To reduce the coupling between layers, each layer uses different communication addresses. For example, layer 2 (physical layer) uses MAC as the communication address, layer 3 (network layer) uses IP as the communication address, layer 4 (transport layer) using <IP, Port> as the communication address. So a new problem arises, communication addresses correspondence problem. ARPs are designed to solve the communication address correspondence problem between layer 2 and layer 3. The header field of the message is filled with the IP address of the target host when the message translates from layer 3 to layer 2. By contrast, it is necessary to know the target host's MAC address to forward the message in layer 2. The process of acquiring corresponding MAC address by the target's IP address is called address resolution (AR).

In general, the ARPs in this paper, which include address resolution protocol (ARP), neighbor discovery protocol (NDP), secure neighbor discovery (SEND), and others, are composed of three parts:

(1)AR process: to solve the corresponding relationship between the IP address and the MAC address, which does not only refer to the AR process in IPv4, but also includes the neighbor discovery process in IPv6;

(2)Duplicate address detection (DAD): to determine whether an IP address is in conflict with the address of other hosts in LAN;

(3)Address cache maintenance process: to complete update, delete, insert, and other operations of the address cache.

AR process is completed by the ARP in IPv4; In IPv6, it uses NDP for the same purpose. NDP is proposed to extend the function of ARP. NDP also provides prefix discovery, neighbor unreachability detection (NUD), DAD, stateless address auto configuration (SLAAC) and other functions. Security is a major threat to AR because the design of the ARPs does not consider security threats. In practical applications, however, security is a problem that cannot be ignored. As the function of AR is to solve the corresponding problem of the communication attributes of layer 2 and layer 3, the main means of attacking is deception, that is, sending false correspondence. For example, when host A resolution IP_Y (host B's IP address), host C sends the $<IP_Y, MAC_C>$ to respond. Host A does not have the ability to distinguish the response, he only accepts it. The man in the middle attack and the DoS attack will pose a huge threat to LAN [1][2].

Several scholars have conducted research on spoofing attack. Literature [3] [4] use discrete event system (DES) in intrusion detection system to monitor all ARP messages in LAN. If an ARP request appeared, it will use the active probe message to resolve the source address. The resolution results are then used to determine whether there is deception. If a cheat behavior exists, the DES will receive more than one ARP reply and will lead event timing different from normal conditions. This method needs a trusted host in LAN to run the DES and to enable the host to monitor all network traffic, so there is a single point of failure. Literature [5], and [1] do not directly trust the ARP packets, especially ARP broadcasting. They use the ICMP message for reverse exploration to test whether the source address of the ARP message is true, and use the test results to decide whether to trust the packets. If the nodes have strong ability to deceive as it can respond to any message, Literature [5] can detect these nodes, but they cannot prevent their attacks. Literature [1] requires the host to maintain a permanent and additional level 2 cache. The cache should ensure that the IP and MAC are one-for-one correspondent. However, if the firewall of the host blocked port 8, this method will not be able to play a role.

Gouda et al. proposed a new AR framework, which includes a secure server and two kinds of new protocol, namely, Invite-Accept and Request-Reply. The Invite-Accept protocol is used by the host to register its <IP, MAC> mapping on the secure server. The Request-Reply protocol is used by the host to request other hosts MAC from a secure server [6]. Brushi et al. proposed Secure-ARP (S-ARP) as an improvement of ARP. S-ARP uses asymmetric encryption technology to authenticate the ARP message and prevent ARP spoofing. S-ARP needs to deploy an authoritative distribution of key server (AKD) in the network to store the IP address and public key of each host. The S-ARP host needs to connect to the AKD server to obtain the public key of the host who sends the ARP message to validate the ARP message received. The host can also cache the public key to improve authentication speed [7]. Issac et al. proposed a unicast ARP protocol combined with DHCP called S-UARP [8]. S-UARP needs to deploy and extend a DHCP server to support AR. This method needs to change ARP and DHCP simultaneously and ensure that the DHCP server is always safe, where a single point of failure also exists.

Source address validation architecture (SAVA) is a new security method. Its main idea is to filter packets based on their source address information. One advantage of this method is that it can prevent attacks directly from the source, and it can provide convenience for source address tracking, traceability, network diagnosis, and management [9] [10]. At present, Source address validation implementation (SAVI) is still in its experimental stage. Source address validation in autonomous region needs router support and the router must complete several centralized computing to affect network performance. SAVI leads to difficult to deployment because the equipment manufacturers implemented SNMP protocol in various ways [11, 12].

In general, the research on address resolution security is still at the experience stage; that is, most studies are based on experience but are lacking theoretical support. Moreover, the development of address resolution protocols (includes ARP, NDP, SEND, etc.) is hindered by the single point of failure, hardware costs, and operating complexity. The main contributions of this paper are as follows. (1) The particularity of the address resolution is summarized. A new kind of security problem is proposed, called seek secret man (SSM) problem, which is the problem of address resolution in an SSM instance. (2) A security protocol, called SSM protocol, is proposed to solve the SSM problem. (3) The AR-SSM is designed, which is an address resolution protocol based on SSM that uses information-hiding technology to prevent spoofing attacks. To prove its security, a simulation attack experiment is conducted, and its performance is compared with those of existing solutions.

The remainder of this paper are organized as follows. In Section 2, the particularity of the address resolution is summarized. Moreover, the security protocol for solving the address resolution problem is proposed and two feasible schemes are given. In Section 3, we designed the address resolution process based on SSM protocol. The protocol flowchart and packet formats are also described. Section 4 is the experimental part, wherein the security of AR-SSM is tested through an experiment. Comparison of AR-SSM with other solutions is also presented in this section. The summary of this paper is detailed in Section 5.

2. **Seek Secret Man Protocol.**

2.1. **The particularity of AR.**

An address resolution has its own characteristics, and its purpose is to establish a connection between hosts A and B. Before a connection is established, the hosts could not communicate point-to-point, meaning that the address resolution protocol is the precondition for point-to-point communication.

Existing security protocols cannot be used to protect address resolution security. Existing security protocols mainly focus on how to carry out key agreement and key distribution, how to identify the integrity of data, how to identify the identity of the other party, and how to realize digital signature etc. These solutions are based on a premise that A (Alice) and B (Bob) have established contact. A can send message to B directly (regardless of safety). Thus, both sides can perform encrypted communication, key agreement, etc. The Diffie-Hellman key agreement protocol is described below.

(1) U selects a random number a_U, $0 \leq a_U \leq n-1$, calculates $b_U = \alpha^{a_U}$, send b_U to V;

(2) V selects a random number a_V, $0 \leq a_V \leq n-1$, calculates $b_V = \alpha^{a_V}$, send b_V to U;

(3) U calculates $K = (b_V)^{a_U}$, V calculates $K = (b_U)^{a_V}$.

Here U can send message to V, V can also send message to U. So, after the execution of Steps (1) to (3), the same key K was generated between U and V [13].

AR is a special problem. The purpose of AR is to build a connection between host A and B. However, host A and B cannot communicate with each other before the establishment of the connection. AR problems do not only exist in the network environment. The following issues are similar to AR principle.

(1) In a war, you know there is a secret agent in the enemy country, but you have never seen him and you only know that his ID is "snake". He was exposed to the enemy,

including his ID. For security, he hides himself and you lose contact with him. Broadcasting is the only way to re-establish contact with him. How should you design a broadcast strategy to find him?

(2) Looking for relatives. Person A needs to seek a relative in City X to give him a gift, but A has never seen this relative before. A only knows that his relative lives in City X and his name is Bob (he may be alive or dead). A drove to City X. How can A find his relative?

(3) Wallet problem. Person A is living in City Y. One day, he picks up a wallet with some money in it. Person A wants to find the owner and the owner may have already left City Y. How can A find the owner?

(4) Finding the same object problem. An artist created a couple of exquisite artworks, one of which is owned by you. You want to buy the other one, but you do not know where it is and who owns it now (it may no longer exist). How can you find it?

(5) Collecting antiques problem. You want to collect antiques such as a piece of a particular style of blue and white porcelain made in the Song Dynasty, but you do not know where it is and who owns it now (this type of porcelain may have disappeared). How can you find it?

There are many similar problems. The above issues have the following characteristics:

(1) A knew B's particular features;

(2) A does not know where is B;

(3) B may or may not exist;

(4) Opening B's features is an effective method to find B, but will result in spoofing attack.

In Problem (5), a public statement was made of one's desire to purchase a particular type of an antique porcelain vase. The situation that someone will bring a fake item for sale is very likely. In Problem (2), the public disclosure of a relatives name may invite impersonators who have malicious intents. In Problem (3), disclosing the amount of money contained in the wallet may attract false claims. All problems with the aforementioned characteristics are termed as the issue of the **seek secret man**(SSM). The address resolution problem is an instance of SSM.

The issue can be resolved more easily if an authority exists (assumed to be G) to provide identification services. With the existence of an identity query system, issue (2) can be easily solved. However, in most cases, especially in the network environment, there is no such authority. For example, there is no authority in LAN that can tell you whether a host claiming to have the IP_X address is actually legitimate. There is no such authority in theory.

2.2. **Seek secret man protocol.**

2.2.1. *The description of SSM problem.* Let S be a set of entities. Each entity has n attributes. Let the attributes be denoted by $R_1, R_2, \ldots, R_n$. A particular entity $S1(S1 \in S)$ can be described by a vector $< S1_{r_1}, S1_{r_2}, \ldots, S1_{r_n} >$, where $r_i \in R_i, i = 1, 2, \ldots, n$.

The constraints are as follows:

(1) The attribute vector of each entity is private information. Any entity, such as $S1$, cannot get attribute information of the other entity through active query;

(2) $S1$ cannot determine whether a certain attribute vector is valid or not;

(3) R_n is the communication attribute and each entity has a different value for its communication attribute. An entity can send information to a target entity through unicast communication, but only when the value of the target's communication attribute is known to the sender.

Seek secret man problem is defined as follows:

A certain entity, $S1(S1 \in S)$, wants to find another entity, $S2(S2 \in S)$. $S1$ does not have complete knowledge of the attributes of $S2$. However, it knows that the attributes of $S2$ meet the following conditions:

$$\begin{array}{c} S2_{r_{i_1}} = a_1 \\ S2_{r_{i_2}} = a_2 \\ \dots \\ S2_{r_{i_m}} = a_m \end{array}$$

where m is an integer $(1 \leq m \leq n)$. $i_1, i_2, \dots, i_m$ are integers in section $[1, n]$ and $i_1 < i_2 < \dots < i_m$. Therefore, seek secret man problem can be described as follow:

Definition 2.1. *SSM problem: in the case of existing adversary, how $S1$ can safely obtain the communication attribute of $S2$.*

On the surface, the seek secret man problem is similar to a zero-knowledge proof or an entity authentication problem, but essentially it is a different problem. A zero-knowledge proof problem requires A to demonstrate to B that he knows a certain secret, but he will not reveal the secret itself in the process of demonstration [14][15]. An entity authentication problem requires A to find a way to convince B that he himself is A [16][17]. SSM problem usually occurs prior to both of the above problems because A does not know who B is at that stage.

2.2.2. *Seek secret man protocol.* $S1$, who satisfies specific criteria, must make public $S2$'s characteristic in the first place. If this task is performed using plaintext, an adversary can easily take advantage through impersonation. As $S1$ and $S2$ are yet to establish connection, if $S1$ uses ciphertext (encrypted text) to make the announcement, $S2$ will not be able to decrypt the ciphertext upon receiving it, let alone know about the fact that $S1$ is searching for him. Hence, the crux to the protocol design is how to publicize characteristics.

The steps for protocol design are as follows:

Stage 1: $S1$ designs the signal.

Stage 2: $S1$ sends out the signal and the other entities decide whether to respond when receiving the signal.

Stage 3: $S1$ authenticates the responses and decides whether to accept.

The specific details are elaborated upon below:

Given that certain risks cannot be avoided when the characteristic is made public, the main consideration for entity $S1$ at Stage 1 is to design a signal that conceals the characteristic and yet allow the entities that comply with the characteristic to recognize the signal at the same time. The entities that satisfy the characteristics will respond in Stage 2. Upon the receipt of the responses, entity $S1$ will carry out authentication before deciding whether to accept those responses.

Option 1: Using a one-way function to seek the secret man

(1) Set of entities S jointly selects a one-way function h.

(2) Entity $S1$ calculates $h(< a_1, a_2, \dots, a_m >)$.

(3) S1 sends out the signal $[h(< a_1, a_2, \dots, a_m >, < i_1, i_2, \dots, i_m >, S1_{r_n}]$; the other entity (assumed to be SX) receives the signal.

(4) The other entities compute $h(< r_{i_1}, r_{i_2}, \dots, r_{i_m} >)$. If it is equal to $h(< a_1, a_2, \dots, a_m >)$, they will proceed to reply and send $[< r_{i_1}, r_{i_2}, \dots, r_{i_m} >), SX_{r_n}]$ to S1.

(5) $S1$ authenticates the reply and decides whether to accept it.

Option 1 is based on random oracle model [18] and its security is subject to the one-way function. If the one-way function h is secure, then the option will also be safe. In Step (4), the information is transmitted via plaintext and the adversary may intercept the information and carry out the man in the middle attack by disguising himself as $S2$ by using the distorted SX_{r_n}. The method is shown in Fig.1, wherein Oscar is the adversary.

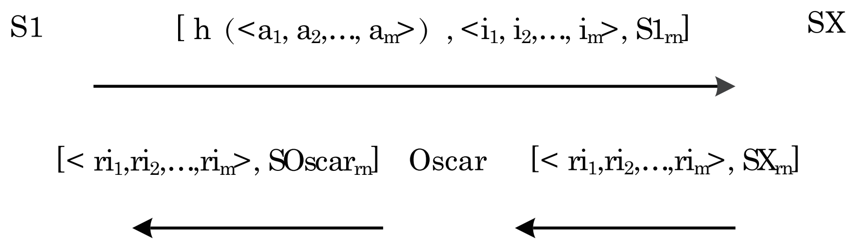
S1
[h (<a1, a2,..., am>) , <i1, i2,..., im>, S1rn]
SX
[< ri1,ri2,...,rim>, SOscarrn]
Oscar
[< ri1,ri2,...,rim>, SXrn]

FIGURE 1. Impersonation attack

This type of attack cannot be launched in some unicast channels. Using a switch-based wired ethernet as an example, the communication attribute is a physical (in an IP system, the MAC address can be considered as a communication attribute in link layer) address. When S1 broadcasts and transmits the signal in Step (3), the signal contains its own MAC address. All the nodes will be able to receive the signal [19][20]. In Step (4), given that S2 already knows the MAC address of S1, its response SX is sent using the unicast mode (port forwarding between switches or point-to-point transmission). Oscar cannot monitor, intercept, or tamper with the signal unless physical methods are applied to cut into the unicast channel between S1 and SX.

Option 2: Using a multi-step interactive program

(1) Set of entities S jointly selects a one-way hash function h and a public-key encryption scheme. Each entity generates its respective publicprivate key pair (ek_i, dk_i).

(2) Entity $S1$ calculates $y = h(< a_1, a_2, \dots, a_m >)$, and then splits y, such that $y = y_1||y_2||\dots||y_k$, where k is an even number.

(3) $S1$ sends out the signal $[y_1, < i_1, i_2, \dots, i_m >, S1_{r_n}]$. The other entity (assumed to be SX) receives the signal.

(4) The other entity computes $y' = h(< r_{i_1}, r_{i_2}, \dots, r_{i_m} >)$ before splitting it to derive $y' = y'_1||y'_2||\dots||y'_k$. If y'_1 and y_1 are identical, they will respond and interact. The interaction sequences are as follows:

a) SX sends out $[y'_2, ek_X, SX_{r_n}]$.

b) $S1$ verifies whether $y'_2 == y_2$. If not, communication is terminated; if yes, it will send $[ek_X(y_3), ek_1]$ to SX.

c) SX verifies whether $dk_X(ek_X(y_3)) == y'_3$. If not, communication is terminated; if yes, it will send $[ek_1(y'_4)]$ to $S1$.

......

k-1)SX verifies whether $dk_X(ek_X(y_{k-1})) == y'_{k-1}$. If not, communication is terminated; if yes, it will send $SX[ek_1(y'_k)]$ to $S1$.

k) $S1$ verifies that $dk_1(ek_1(yk')) == y_k$.

(5) $S1$ decides whether to accept the response.

For Step (5) of both in Options 1 and 2, $S1$ has to decide whether to accept the response after verification. If multiple entities are verified, $S1$ has to decide how to accept the responses. One method is first-come first-served (FCFS), which means the first verified entity is accepted.

For Oscar to carry out a spoofing attack, he must modify SX_{r_n} to its own communication attribute during Step a) of the interaction sequence. Even after successful modification, Oscar should still complete all subsequent steps of the interaction sequence before successfully deceiving $S1$. The premise for executing the subsequent interaction is that

Oscar must know the remaining $y'_4, y'_6, \ldots, y'_k$. However, in addition to the three pieces of public information $< i_1, i_2, \ldots, i_m >, y_1$, and $S1_{r_n}$, the other information that Oscar can at most obtain are ek_X, ek_1, SX_{r_n}, and y'_2. Without any additional information, the subsequent steps of the interaction sequence cannot be completed.

This form of interactive verification involves a process of mutual authentication by both parties. $S1$ can only confirm that the other party has $y'_2, y'_4, \ldots, y'_k$, but it is not possible to verify whether the other party owns $y_1, y_3, \ldots, y_{k-1}$. Therefore, deception may be possible. If $|y| = L$, then the probability of deception is $2^{(-L/2)}$. Nowadays, the common length of hash output is 128 bit or longer. As a result, the probability of being deceived is negligible.

3. A novel address resolution protocol based on SSM.

The fundamental reason that causes the vulnerability of address resolution to attacks is that the host has to make the destination of address resolution (assuming IP_X) public to find the MAC address of the target IP address. Consequently, the attack node could easily send a spoofing reply, preventing the victim node from distinguish between a real reply and a deception reply.

Given that the problem of address resolution is an instance of SSM, the address resolution can be resolved by the characteristics of the SSM protocol. The method is described as follows. First, the destination address is opened after a hash operation. This allows no one but the host to possess IP_X can read the destination address. Other hosts (including attack node) cannot figure out the destination address, disallowing them to send a reply or carry out an attack and consequently ensuring the security of address resolution. Therefore, we design a new address resolution process based on SSM protocol, called AR-SSM.

3.1. Introduction of hash.

Hash function h is a mapping function, $h : \{0,1\}^* \rightarrow \{0,1\}^n$. The $\{0,1\}^*$ represents a set of arbitrary length bit string. The $\{0,1\}^n$ denote the set of n bit length string [21]. By definition, hash function h can map a message x of arbitrary length to a shorter message y with a fixed length. That means $y = h(x)$, x commonly known as the preimage, y commonly called the message digest. SHA-1, MD5, and RIPMED are common hash functions. A hash function is known to be safe if it has the following three properties:

(1) Anti-preimage attack: For any given output y, find x to arrive at $h(x) = y$, wherein calculation is infeasible;

(2) Anti-second preimage attack: For any given input x, find x', which is different from x to arrive at $h(x) = h(x')$, wherein calculation is infeasible;

(3) Anti-collision attack: Find two different input x and x' to arrive at $h(x) = h(x')$, wherein calculation is infeasible.

3.2. AR-SSM protocol.

AR-SSM takes NDP as the prototype and uses of neighbor solicitation (NS) and neighbor advertisement (NA) to complete the AR process. The message format was modified to support the new protocol. AR-SSM uses SSM scheme 1. The hash function is MD5.

Assume there are three hosts in the network, namely, A, B, C, and their address configuration information is shown in Table 1.

First, we provide the AR-SSM message format, as shown in Fig.2. Compared with the NDP message, AR-SSM increased in two fields, namely, start_bit (64-bit) field and hash_64 field (64-bit) field. Start_bit field provides the start byte for comparison, whereas hash_64 field provides the 64 bit hash value of the target address (from start_bit to start_bit+63, total 64-bit) for detection. The value of this field is calculated as follows: suppose host A needs to resolve the IPv6 address 1::2:B, host A should first compute the hash value

TABLE 1. Basic information of Hosts

Host	IP	MAC	hash
A	1::2:A	0800-2700-0001	1d501fb0fe53ee99bbab3ef4685f2001
B	1::2:B	0800-2700-0002	8ef841bd7e18a75e47941fa979a4bbad
C	1::2:C	0800-2700-0003	20a6d4738c32a5f8b88d17760be9acd5

of 1::2:B, and then generate a random start_bit field (from 0 to 63). As shown in Figure 3, the MD5 value of 128-bit address is first computed, then 64-bit length is intercepted from start_bit and written to the hash_64 field. We defined this process as function H64 (start_bit, IPv6_addr). The other fields are defined as follows: target address field is usually filled with resolve destination address. The option field has different meaning in different types of message, which is generally referred to as the link layer address. The type field represents the message type, the NS type value is 135, and the NA type is 136. The flags field is only valid in NA. To separate this field from the NDP, the type value of $NS_{AR-SSM}$is 200, and the type value of NA_{AR-SSM} is 201.

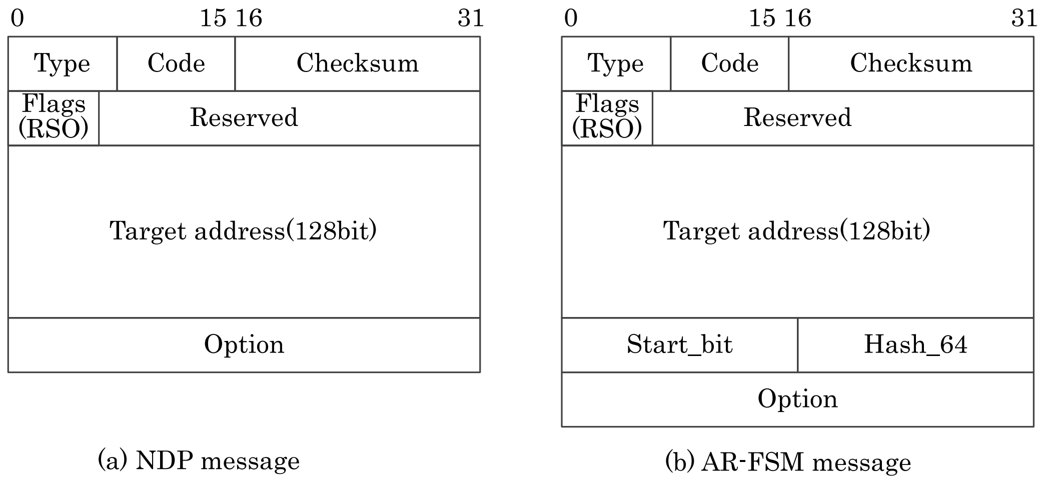

FIGURE 2. Message format comparison of NDP and AR-SSM

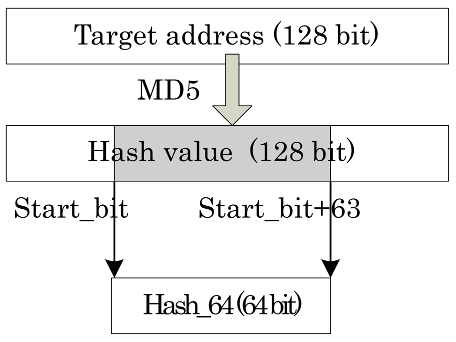

FIGURE 3. The calculation process of Hash_64 field

The process of AR-SSM is shown in Figure 4, which is described as follows: when host A performs an AR (suppose the target address is IP_X), host A needs to generate a random start_bit and write it to start_bit field, then calculate the hash value of IP_X and intercept a 64-bit length string from start_bit of the hash value, write the string to hash_64 field of

NS_{AR-SSM}, and multicast NS_{AR-SSM}. Unlike in NDP, the target address of NS_{AR-SSM} is filled in empty address ("::") to hide the address that will be resolved. The other nodes on the same link will detect the hash_64 field after receiving the NS_{AR-SSM} to check if their own address meet the demand of H64(start_bit,IPv6_addr)=hash_64. If the address meets the demand, assume that it is IP_Y, The node will then send NA_{AR-SSM} to reply. In NA_{AR-SSM}, IP_Y is filled in the target address field, and H64(start_bit, IP_Y) is filled in the hash_64 field. If a node has more than one address and meets the hash value matching condition, then the NA_{AR-SSM} reply is sent separately. Within the stipulated time, host A will check all the received NA_{AR-SSM}. Checking is performed in two steps:

Step 1: Check whether NA_{AR-SSM} and NS_{AR-SSM} have the same hash_64 field value. If not, then discard the packet, otherwise, go to Step 2.

Step 2: Check if the target address field of NA_{AR-SSM} is the same as the address to be resolved. If it is, then it means the AR is successful, otherwise, perform address validation, H64(start_bit,target address)=hash_64. If equal, discard the packet, if not, blacklist the node that send the NA_{AR-SSM} (optional step). If host A did not receive any NA_{AR-SSM} whose target address field address is the same with the address to be detected within the stipulated time (typically 3 seconds), then AR fails.

The blacklist mechanism is optional in Step 2 is based on the following two principles:

Principle 1: Given that the hash_64 field value of NS_{AR-SSM} is known, if the hash_64 field values of NA_{AR-SSM} and NS_{AR-SSM} are not the same, then speculate the node to be malicious and blacklist its MAC address.

Principle 2: If the hash_64 field values of NA_{AR-SSM} and NS_{AR-SSM} are the same, but the target address field values of NA_{AR-SSM} and NS_{AR-SSM} are not and H64 (start_bit, target address) $\neq$ hash_64, then speculate the existence of deceit behavior of the node and blacklist its MAC address.

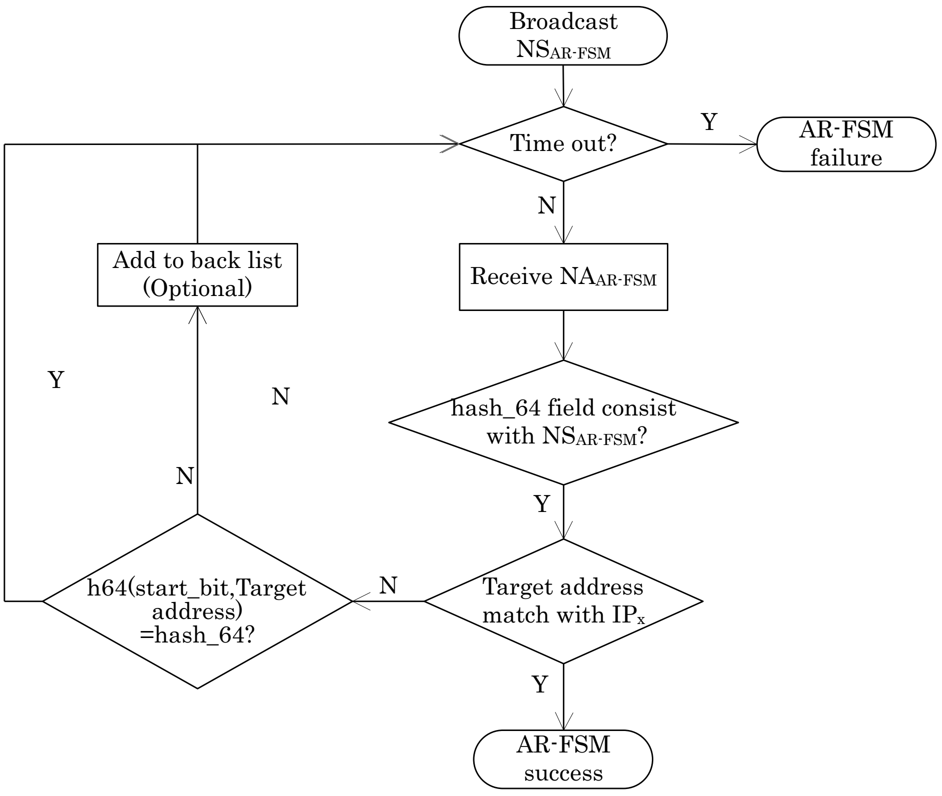
Broadcast NSAR-FSM
Time out?
Y
AR-FSM failure
N
Receive NAAR-FSM
Add to back list (Optional)
Y
N
hash_64 field consist with NSAR-FSM?
N
Y
h64(start_bit,Target address) =hash_64?
N
Target address match with IPx
Y
AR-FSM success

FIGURE 4. Flow chart of AR-SSM

We illustrate AR-SSM by an example. Assume that host A wants to find the MAC address of host B whose address is 1::2:B. A then sends NS_{AR-SSM} to perform AR. Assume the start_bit is 0, the hash value of the address 1::2:B is "8ef841bd7e18a75e47941fa979a4bbad". In the "hash_64" field, host A fills the pre 64 bits of hash value "8ef841bd7e18a75e" and the target address field is filled in with "::", which means an empty address. Then, multicast NS_{AR-SSM}.

After host B receives the NS_{AR-SSM}, it will check its own address and the hash_64 field of NS_{AR-SSM}. B finds that the pre 64-bit length hash value of its own address 1::2:B is in accordance with hash_64 field. B thens send NA_{AR-SSM} to reply. The $\text{NS}_{AR-SSM}$and NA_{AR-SSM} used in the above process are shown in Fig. 5. If host B owns another address IP_Z, of which the hash value is "8ef841bd7e18a75ef47941fa979a4bba", the pre 64-bit also matches the "hash_64" field, then host B needs to send NA_{AR-SSM} to reply again.

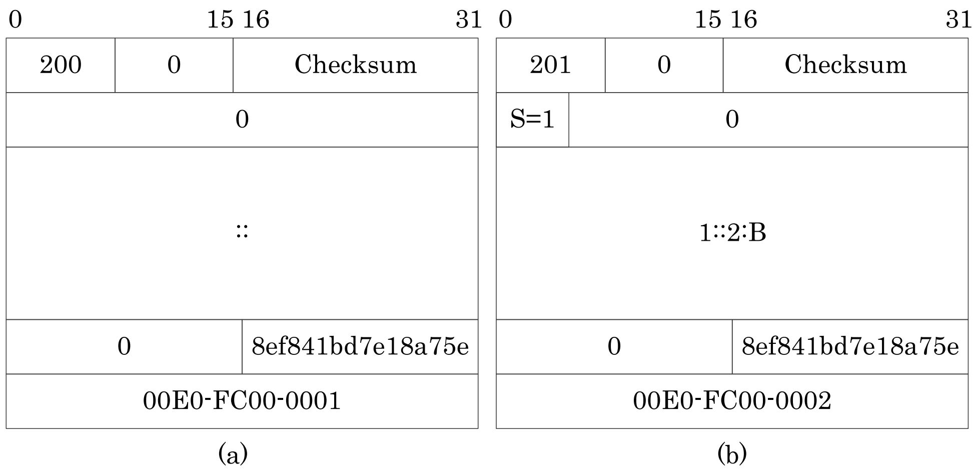

FIGURE 5. NS_{AR-SSM} and NA_{AR-SSM}

4. Security Analysis and Experiments.

4.1. Security analysis.

4.1.1. *The hash_64 field length selection.* Suppose there are n nodes in the network, and each node has m IPv6 address, which is uniformly distributed. The length of hash_64 field is L bit, and the number of reply packet in AR process is about $2^{(L-128)} \times m \times n$. Thus, length L is undoubtedly very important because, in some degree, it determines the number of nodes that should reply to the NS_{AR-SSM} message. The shorter the field length, the safer it will be. However, more nodes will be required to reply, thus, it will cause greater disturbance to the network. The longer the field length, the more vulnerable it is to attacks, but the nodes required to reply will be reduced. The value of length L is 64 in this paper. In AR, if there are 2^{16} nodes in the network and each node owns 2^8 IP address, then the amount of packet reply is nearly $2^{(-40)}$.

4.1.2. *Security for hash_64 field length.* Birthday attack shows that the n length message digest, random plaintext selection $O(2^{(n/2)})$ times has a 50% probability of causing a collision. For the iterated function with MD structure, differential attack is an effective attack way of searching for collision. Wang et al. presented an effective attack method for hash function with MD structure in literature [22][23]; Collision for MD5 will be found after cycling for no more than 2^{39} times, but MD4 only requires 2^8 MD cycles. As hash_64 field is generated by a 128-bit IPv6 address, the preimage has a fixed length and the collision attack would be easier. Therefore, from the perspective of collision, AR-SSM using MD5 algorithm is not safe. However, intercepting 64-bits length can make up for this deficiency. Unlike the integrity of data, digital signatures, or other applications, the attacker in AR only needs to find a collision, a second preimage is not enough. Oscar must find the preimage to realize an effective attack. Creating a preimage attack is theoretically infeasible because hash_64 field did not give out a complete message digest.

4.1.3. *Analysis of attack methods of malicious nodes.* Assume that host C is a malicious node. The optional attack methods are shown as follows:

Method 1: Normal attack or an attack according to the destination address to be resolved sends out a false reply. In AR-SSM, given that NS_{AR-SSM} did not give out the destination address to be resolved, the normal attack against the destination address is not feasible.

Method 2: This method is based on the network prefix advertised by the router to generate large random addresses and send a large number of NA_{AR-SSM} to increase the success rate of attack. However, due to AR-SSM uses blacklist mechanism, this attack will be blacklisted. Then, all packets are discarded. Thus, this method of attack is not feasible.

Method 3: Pseudo collision attack, wherein host C will analyze hash_64 field after receiving NS_{AR-SSM} to find more than one pseudo-collision address. Host C then sends multiple NA_{AR-SSM} to reply to increase the success rate of attacks.

Assuming that there are k network prefix in LAN. Host C can find a large number of pseudo collision addresses according to these network prefix. C then sends the pseudo NA_{AR-SSM} to reply. Theoretically, host C's search space is $k \times 2^{64}$. The more pseudo collision addresses C finds, the higher the success rate of attack. Given that the AR process was generally completed within 3 seconds, host C will not find a large number of pseudo-collision address in such a short time. Therefore, this method of attack is infeasible.

4.2. **Experiments.** The simulation software used is OPNET. Network environment is local area network, which includes a switch node, a cheat node, and seven normal nodes. Normal nodes periodically performs AR. Normal nodes include two packet transmission sources, namely, Src1 and Src2. Src1 is used to generate background traffic, and its distribution is derived according to the 30-day data traffic of a university firewall (data collection software is Solarwin orin, firewall is Hillstone M6860). The statistics and the distribution are shown in Fig. 6 and 7. Src2 is used to generate AR packets, and uses the average distribution, the mean is 1.

In the experiment, assume that each node has 2^{10} addresses. There are 2^{8} network prefixes in LAN, and the random address space is 2^{32}. The experiment is divided into two scenarios. In scenario one, the address cache rate of traditional AR and AR-SSM were simulated with the existence of malicious node. The cache pollution rate is calculated as the ratio of the error address cache numbers and the total address cache numbers. The attack node mainly uses two kinds of attack method: aiming for the traditional AR and according to the "target address" in NS to falsify NA for reply. For AR-SSM, the attacker uses random address to reply; hash_64 field NA_{AR-SSM} is same as in NS_{AR-SSM}. Contrast can be observed in Fig. 8. If attack node exists, the cache pollution rate of the AR-SSM is lower than NDP.

We compare AR-SSM with five other representative solutions, see table 2. The use of encryption technology affects the protocol performance to some extent. This phenomenon is obvious in the method in [7]. The methods used in [3, 6, 7, 8] need to add an additional server to the network and ensure that the server is always secure. This greatly increases

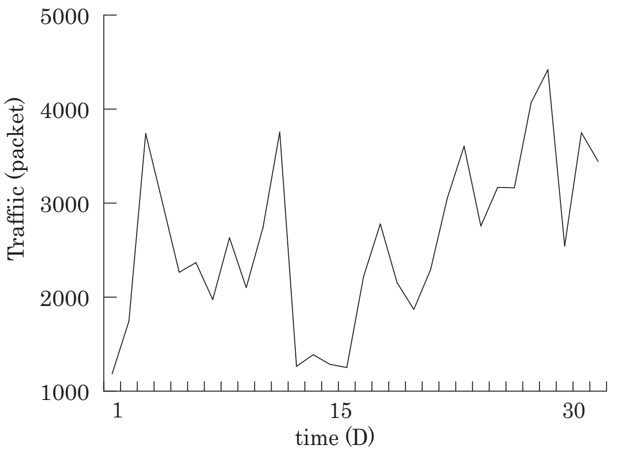

FIGURE 6. Average traffic of 30 days

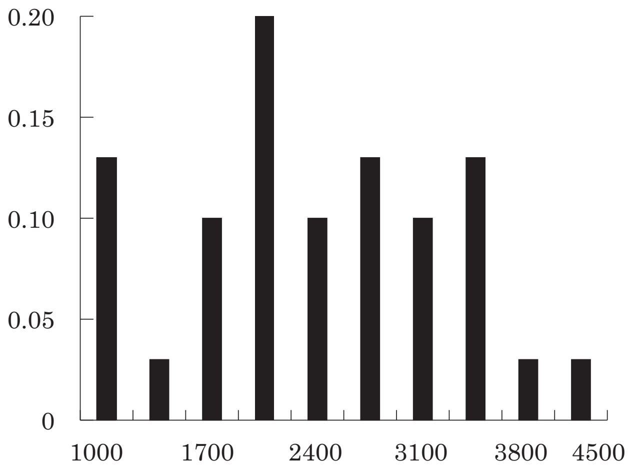

FIGURE 7. Probability distribution of traffic

the deployment costs of these methods, and causes a single point of failure to exist. In [3, 7, 8], a security server is added in the LAN; thus, the host only needs to query the MAC address of the target IP in the security server. Broadcast, which greatly reduces the communication overhead, is not needed. In [3], IDS needs a mirror port on the switch to monitor all network traffic, which requires switch support. AR-SSM does not need to monitor network traffic, and the complexity of the hash algorithm is also much less than that of asymmetric encryption algorithm. Thus, the effect on the protocol performance is low. Moreover, AR-SSM does not need a security server, thus its deployment cost is low.

5. **Conclusion.** The hierarchical network architecture simplifies the network protocol design. Each layer is forced to use independent communication addresses. Given that communication addresses are independent of each other, a definite correspondence must exist between the upper and lower layers communication addresses. This is the main reason for the existence of address resolution protocols. All existing address resolution protocols use broadcast (multicast) to make the destination of address resolution public, as it provides convenience for a malicious node to carry out attacks. To overcome this weakness, we propose a more secure protocol called AR-SSM. This protocol uses information-hiding technology to make the destination address public. Moreover, only specific nodes can figure out the destination address, whereas other nodes (including

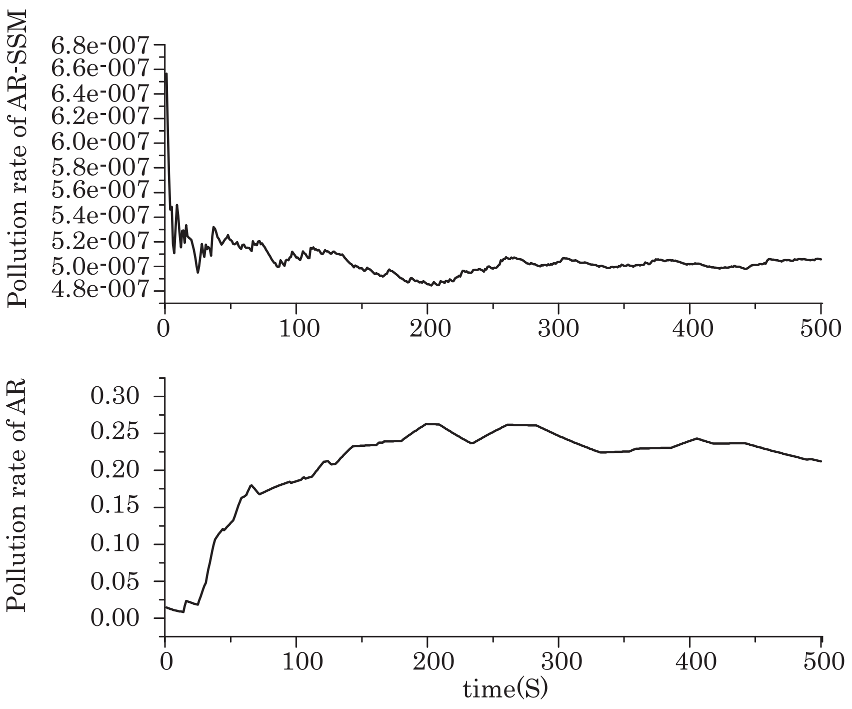

FIGURE 8. Cache polution rate comparison of NDP and AR-SSM

TABLE 2. Comparison with existing solutions

Existing solution	Crypo-grahpy	Secure server used	Traffic monitor	Performance degradation	Communication overhead
AR-SSM	Yes	No	No	Low	O(n+1+$2^{(L-128)} \cdot m \cdot n$)
ES-ARP[24]	No	No	No	Low	O(2n)
S-ARP[7]	Yes	Yes(AKD server)	No	High	O(n+3)
Active DES[3]	No	Yes(IDS)	Yes	Very low	O(n+3)
Gouda[6]	No	Yes(Secure server)	No	Low	O(2n+2)
S-UARP[8]	Yes	Yes(DHCP+ Secure server)	No	Low	O(n+3)

malicious nodes) cannot, making them unable to launch attacks. Experiments and horizontal comparison show that AR-SSM is better than existing security solutions in terms of security, deployment cost, protocol performance, and so on.

Acknowledgment. This work was supported by the National Natural Science Foundation of China (Grant No.61173024).

REFERENCES

[1] J. H. Gao, and K. J. Xia, ARP spoofing detection algorithm using ICMP protocol, *Proc. of International Conference on Computer Communication & Informatics*, IEEE, pp.1-6, 2013.

[2] P. Nikander, J.Kempf and E.Nordmark, RFC 3756: IPv6 Neighbor Discovery (ND) Trust Models and Threats, IETF, Work in Progress, 2004.

[3] F. A. Barbhuiya, S. Biswas and S. Nandi, An active DES based IDS for ARP spoofing, *Proc. of International Conference on Systems, Man & Cybernetics*, IEEE, pp.2743-2748, 2011.

[4] H. Neminath, S. Biswas, S. Roopa, A DES approach to intrusion detection system for ARP spoofing attacks, *Proc. of the 18th Mediterranean Conference on Control and Automation*, IEEE, vol. 20, no. 1, pp.695-700, 2010.

[5] P. Pandey, Prevention of ARP spoofing: A probe packet based technique, *Proc. of Conference on International Advance Computing*, IEEE, pp.147-153, 2013.

[6] Mohamed G. Gouda and Chin-tser Huang. A secure address resolution protocol, *Proc. of Computer Networks the International Journal of Computer & Telecommunications Networking*, vol.41, no.1, pp.57-71, 2003.

[7] D. Bruschi, A. Ornaghi and E. Rosti, S-ARP: a secure address resolution protocol, *Proc. of the 19th Annual Conference on Computer Security Applications*, IEEE, pp.66-74, 2003.

[8] B. Issac and L.A. Mohammed, Secure unicast address resolution protocol (S-UARP) by extending DHCP, *Proc. of the 13th International Conference on Networks*, IEEE, pp.363-368, 2005.

[9] J. Wu, J. Bi, X. Li, G. Ren and K. Xu, RFC5210: A source address validation architecture (sava) testbed and deployment experience, IETF, 2008.

[10] J. Wu, G. Ren and X. Li, Source address validation: architecture and protocol design.*Proc. of International Conference on Network Protocols*, IEEE, pp.276 - 283, 2007.

[11] P. Xiao and J. Bi, OpenFlow based intra-AS source address validation, *Journal of Chinese Computer Systems*, vol.34, no.9, pp.1999-2003, 2013.

[12] J. Li, J. Wu, K. Xu and W. Chen, An hierarchical inter-domain authenticated source address validation solution, *Chinese Journal of Computers*, vol.35, no.1, pp.85-100, 2012.

[13] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, vol.22, no.6, pp.644-654, 1976.

[14] D. Chaum, J H. Evertse and J. Van De Graaf, An improved protocol for demonstrating possession of discrete logarithms and some generalizations, *Advances in Cryptology-EUROCRYPT'87*, Springer Berlin Heidelberg, 1988.

[15] D. Chaum, J H. Evertse and J. Van De Graaf, Demonstrating possession of a discrete logarithm without revealing it, *Advances in Cryptology-CRYPTO'86*, Springer Berlin Heidelberg, 1987.

[16] J. A. Evans, W. Kantrowitz and E. Weiss, A user authentication scheme not requiring secrecy in the computer, *Communications of the ACM*, vol.17, no.8, pp.437-442, 1974.

[17] L. Lamport, Password authentication with insecure communication, *Communications of the ACM*, vol.24, no.11, pp.770-772, 1981.

[18] S. Goldwasser and M. Silvio, Probabilistic encryption, *Journal of computer and system sciences*, vol.28, no.2, pp.270-299, 1984.

[19] J. Postel, RFC 791: Internet protocol, IETF, 1981.

[20] S. Deering and R. Hinden, RFC 2460: Internet protocol version 6 (IPv6) specification, IETF, 1998.

[21] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C*, John wiley & Sons, 2007.

[22] X. Wang, X. Lai, D. Feng, H. Chen and X. Yu, Cryptanalysis of the hash functions MD4 and RIPEMD, *Advances in Cryptology-EUROCRYPT'2005*, Springer Berlin Heidelberg, pp.1-18, 2005.

[23] X. Wang and H. Yu, How to break MD5 and other hash functions, *Proc. of International Conference on Theory & Applications of Cryptographic Techniques*, pp.19-35, 2005.

[24] M. Ataullah, and N. Chauhan, ES-ARP: An Efficient And Secure Address Resolution Protocol, *Proc. of Students Conference on Electrical Electronics & Computer Science*, IEEE, pp.1-5, 2012.