Collaborative Approach to Mitigating ARP Poisoning-based Man-in-the-Middle Attacks

Seung Yeob Nam[a], Sirojiddin Djuraev[a], Minho Park[b]

[a]*Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 712-749, Republic of Korea*
[b]*School of Electronic Engineering, Soongsil University, Dongjak-gu, Seoul 156-743, Republic of Korea*

Abstract

In this paper, we propose a new mechanism for counteracting ARP (Address Resolution Protocol) poisoning-based Man-in-the-Middle (MITM) attacks in a subnet, where wired and wireless nodes can coexist. The key idea is that even a new node can be protected from an ARP cache poisoning attack if the mapping between an IP and the corresponding MAC addresses is resolved through fair voting among neighbor nodes under the condition that the number of good nodes is larger than that of malicious nodes. Providing fairness in voting among the nodes that are heterogeneous in terms of the processing capability and access medium is quite a challenge. We attempt to achieve fairness in voting using the uniform transmission capability of Ethernet LAN cards and smaller medium access delays of Ethernet than for wireless LAN. Although there is another scheme that resolves the same issue based on voting, i.e. MR-ARP, the voting fairness is improved further by filtering the voting reply messages from the too-early responding nodes, and the voting-related key parameters are determined analytically considering the fairness in voting. This paper shows that fairness in voting can be achieved using the proposed approach, overcoming the limitations of other voting-based schemes, and ARP poisoning-based MITM attacks can be mitigated in a more generalized environment through experiments.

Keywords: Address Resolution Protocol (ARP), ARP cache poisoning, Man-in-the-Middle attack, voting, voting fairness

1. Introduction

The Address Resolution Protocol (ARP) is used to find the Media Access Control (MAC) address corresponding to the IP address of a node in the same subnet [1, 2]. The resolved address is kept temporarily in the ARP cache to reduce the resolution time for subsequent queries for the same IP address [3]. ARP poisoning attack refers to the malicious behavior of registering a false (IP, MAC) address mapping in the ARP cache of a remote machine.

If the ARP poisoning attack is successful, the attacker can eavesdrop in the communication between the other nodes, modify the packet contents, and hijack the connection. Furthermore, ARP cache poisoning can be used to mount other types of attacks such as Denial of Service (DoS) [4], Man-in-the-Middle (MITM) attacks [5], and JavaScript insertion attack [6].

Email addresses: synam@ynu.ac.kr (Seung Yeob Nam), sirojiddin1987@gmail.com (Sirojiddin Djuraev), mhp@ssu.ac.kr (Minho Park)

Several attempts have been made to resolve the ARP cache poisoning problem in a wired or wireless LAN environment. Nevertheless, it is still not easy to find a mechanism that can prevent ARP poisoning-based MITM attacks in a practical environment where many mobile nodes easily join and leave the considered subnet. Conventional approaches can be classified into two categories depending on the need to upgrade the Ethernet switches. Dynamic ARP Inspection (DAI) [7] corresponds to an approach requiring the support of Ethernet switches. DAI might prevent ARP poisoning, but this requires manual configuration by network managers. The wireless nodes connected via the same AP may not be protected by DAI. The approaches that do not require the support from Ethernet switches can be classified into two categories depending on the use of cryptography. Antidote [8] employs a non-cryptographic approach, and attempts to prevent ARP cache poisoning by contacting and giving a higher priority to the previous owner of a given IP address in the case of MAC conflict. However, Antidote cannot prevent poisoning

for a new IP address if a malicious ARP reply arrives first [9].

S-ARP [9], a modified version of S-ARP [10], and Ticket-based ARP (TARP) [11] are well-known cryptography-based approaches. 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. Furthermore, there is a cost of manual configuration to disseminate the public key and the MAC address of the central server to newly arriving hosts. This manual setting might not be appropriate for the environment of Wi-Fi hot-spots where the wireless nodes can enter and leave easily.

Philip [12] investigated an approach to prevent ARP cache poisoning in wireless LAN by implementing the defense mechanism in the AP. The AP constructs a list of correct IP-to-MAC address mapping by monitoring the DHCP ACK messages or referring to the DHCP leases file, and blocks all the ARP packets with false mapping based on the established list. However, this approach can be effective only for the dynamic IP addresses allocated through DHCP, and cannot prevent the ARP cache poisoning that occurs in the wired LAN. In this paper, we attempt to protect the upgraded nodes from ARP poisoning-based MITM attacks, regardless of whether they are wired or wireless nodes.

Recently, Nam *et al.* [13] proposed an enhanced version of ARP, called MR-ARP, to prevent ARP poisoning-based MITM attacks in the Ethernet by employing the concept of voting. MR-ARP attempts to determine the owner of a given IP address by giving a higher priority to the previous owner in case of conflict on the MAC address of the owner. This is similar to the mechanism of Antidote. If an MR-ARP node observes conflict on the owner of an IP address that is not registered in its own ARP cache, the MR-ARP node triggers voting on the owner of that IP address among the neighbor nodes and makes a decision based on the voting result. In this mechanism, it is important to maintain fairness in voting among the different nodes by having each of them send a similar number of votes. MR-ARP achieves fairness among the wired nodes because the nodes connected to the same Ethernet have the same transmission rate, and are likely to send a similar number of packets during a common fixed interval. On the other hand, MR-ARP might not guarantee fairness in voting in a wireless LAN environment, because the transmission rate may not be uniform among different wireless nodes due to the traffic rate adaptation based on the signal-to-noise ratio (SNR), i.e. Auto Rate Fallback (ARF) [14]. Nam *et al.* [15] attempted to resolve the unfairness problem of MR-ARP in a wireless LAN by incorporating computational puzzles [16–18] in the voting procedure of another security-enhanced version of ARP, EMR-ARP. On the other hand, EMR-ARP requires the condition that the computation power of the neighbor machines is no more than a factor of 2 different, and this assumption may not be valid even among the popular tablet machines.

Therefore, we investigate a new mechanism that can prevent ARP poisoning-based MITM attacks with a relaxed assumption and less infrastructure upgrade overheads. The proposed scheme is also based on voting among the neighbor nodes, and an attempt is made to provide fairness in voting by relying on the uniform transmission rate in the Ethernet, in a similar manner to MR-ARP. However, the voting scheme is refined further to provide fairness in voting in a more generalized environment compared to other voting-based schemes, i.e. MR-ARP and EMR-ARP. The contribution of this paper can be summarized as follows:

- The proposed ARP scheme can be deployed by upgrading the operating system, and the proposed scheme can protect the upgraded machines from ARP poisoning-based MITM attacks, even though the computation power of different wired/wireless machines is different by more than a factor of 2.
- The fairness in voting is improved compared to MR-ARP by dropping too early reply packets. The impact of the early packet filtering scheme on the voting fairness is analytically investigated for an example case.
- The voting traffic overhead of the proposed scheme is lower than for other voting-based schemes, i.e. MR-ARP and EMR-ARP.
- The voting related parameters, including the number of reply messages required for each neighbor node, are determined analytically considering the fairness in voting.

The proposed scheme also shares the following advantages with other voting-based schemes. The manual setup or configuration is not required for the newly arriving wired or wireless nodes, because the public key cryptography is not required, and it is free from the single point of failure problem due to the absence of a centralized server. This scheme is backward compatible with existing ARP, and the infrastructure upgrade cost is minimal, because it does not require the upgrade of Ethernet switches or modification of DHCP.

The remainder of the paper is organized as follows. In Section 2, the proposed mechanism to prevent ARP

poisoning-based MITM attack is discussed in detail. Section 3 determines the voting-related parameters analytically considering the fairness in voting. In Section 4, an algorithm to discard too early reply packets is introduced to improve the voting fairness, and the impact of the early packet filtering algorithm on the voting fairness is analyzed. In Section 5, the performance of the proposed scheme is compared with that of other voting-based schemes based on the analysis and experiment results. Finally, conclusions are reported in Section 6.

2. Voting-based Security Enhanced ARP

The voting-based MITM prevention mechanisms, i.e. MR-ARP and EMR-ARP, usually require some assumptions for normal operation. The following two assumptions are needed for MR-ARP:

- *Assumption 1:* The number of the voting-cognizant good nodes should be larger than that of the malicious nodes in the same subnet.
- *Assumption 2:* All the voting-cognizant good nodes and malicious nodes should be connected to the subnet through a wire.

EMR-ARP requires the following two assumptions:

- *Assumption 3:* The number of the voting-cognizant good nodes should be larger than that of the malicious nodes in the same subnet.
- *Assumption 4:* The computation power of different machines is no more than a factor of 2 different.

Assumption 1 for MR-ARP is identical to *Assumption 3* for EMR-ARP. The main goal of this study is to develop a new ARP protocol that can prevent ARP poisoning-based MITM attacks with fewer assumptions compared to MR-ARP or EMR-ARP. In more detail, we attempt to drop *Assumption 2* for MR-ARP and *Assumption 4* for EMR-ARP, and devise a new ARP, i.e. Generalized MR-ARP (GMR-ARP), under the following assumption.

- *Assumption 5:* The number of the voting-cognizant good nodes is larger than that of the malicious nodes among the neighbor nodes in a wired network.

At least one voting-cognizant good node is needed in the wired network to satisfy the above condition, i.e. *Assumption 5*. If an internal node wants to access the external network, then the node needs to know the MAC address of the gateway router, because the packets from that node need to be delivered to the gateway router first. Therefore, each internal node normally attempts to find the MAC address of the gateway router repeatedly as the timer for that entry expires in the ARP cache, and the malicious node tends to launch MITM attacks between an internal node and the gateway router. Since the voting-based MITM prevention schemes can protect only the nodes whose ARP code is upgraded, we assume that the new ARP code is deployed at the gateway router to prevent the most prevalent types of MITM attacks, i.e. between an internal node and gateway router, in a subnet, and we consider that this is a reasonable assumption.

The proposed MITM-resistant version of ARP, i.e. *GMR-ARP*, follows the basic approach of MR-ARP [13]. One of the key ideas is as follows. When Node A knows the correct (IP, MAC) address mapping for Node B, if Node A holds the mapping while B is alive, then ARP poisoning on Node A and the MITM attack between A and B can be prevented.

MR-ARP and EMR-ARP use a long-term (IP, MAC) mapping table to manage (IP, MAC) mapping for all the machines alive in the same subnet. If a new MR-ARP or EMR-ARP node attempts to find the MAC address corresponding to each IP address in a given subnet, the nodes in the same subnet might suffer from lower data throughput for an extended time period while the new node resolves the address mapping through voting. As an example, EMR-ARP spends approximately one second to find the owner of one IP address. If there are 60,000 nodes in a given /16 subnet, then a new EMR-ARP node might need to spend approximately 60,000 seconds to find the MAC addresses of all the nodes. The new MR-ARP/EMR-ARP node and other neighbor nodes in the same subnet are likely to suffer from a lower throughput during this time interval due to flooding of the voting packets for MR-ARP, or processing power consumed by cryptographic puzzles for EMR-ARP. To overcome this voting overhead problem of both MR-ARP and EMR-ARP, we let the new GMR-ARP node fill the long-term table as much as possible with a special voting procedure during the initialization stage.

Before describing the initialization stage, we discuss the long-term (IP, MAC) mapping table in more detail. Although MR-ARP and EMR-ARP attempts to maintain (IP, MAC) mapping for all alive machines in the same subnet, GMR-ARP basically attempts to manage (IP, MAC) mapping only for other GMR-ARP nodes because the goal of GMR-ARP is to protect the GMR-ARP nodes from ARP cache poisoning or MITM attacks. On the other hand, the IP address of non-GMR-

ARP node can also be maintained in the long-term table of a GMR-ARP node if the GMR-ARP node is interested in that IP address. 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. To avoid losing the mapping of (IP_a, MAC_a) for an alive host after 60 minutes, we send new ARP request messages for IP_a only to MAC_a through unicasting to check if the MAC_a is alive. In this case, 10 ARP request messages are sent at random intervals with a mean of 10 msec. If at least one ARP reply is returned, the mapping is registered in the short-term ARP cache and the corresponding long-term table timer is again set to 60 minutes. If no ARP reply returns, then the mapping of (IP_a, MAC_a) is considered invalid and the corresponding entry is deleted from the long-term table. Therefore, (IP, MAC) mapping can be retained in the long-term table as long as the binding is valid.

The MR-ARP or EMR-ARP node monitors every ARP request message to find all machines alive in the same subnet, and adds them to the long-term table. On the other hand, a new GMR-ARP node attempts to find all the other GMR-ARP nodes, and adds them to the long-term table as early as possible during the initialization stage to reduce the voting time overhead. The initialization stage is described in more detail below.

2.1. Initialization Stage

The initialization procedure is performed only once during the lifetime of a given machine. The initialization stage consists of two steps. In the first step, the rebooted or newly attached node advertises its own IP/MAC mapping through a gratuitous ARP request packet [19]. Each GMR-ARP node that receives the gratuitous ARP packet examines whether the source IP address of the received packet is registered in its own long-term table. If it knows the received IP address, it resolves the conflict by asking the previous owner of that IP address and giving it priority. If the GMR-ARP node does not know the received IP address, it just neglects the received gratuitous packet. Fig. 1 summarizes the task that is performed by the GMR-ARP node receiving a gratuitous ARP request packet. This first step of the initialization stage is identical to that of EMR-ARP [15]. This packet is sent first to update the (IP, MAC) mapping for a given IP easily without the burden of voting, particularly when the owner of a given IP address changes legitimately through DHCP.

The second step has three goals. The first goal is to ensure that the new GMR-ARP node can use the selected IP address without problems. The second goal

```
/* (IPa, MACa): the sender protocol (IP) and hardware (MAC)
        addresses of the received gratuitous ARP request packet */

if((IPa, MACa) is registered in the long-term table){
  register (IPa, MACa) in the short-term cache;
  set the long-term table timer to 60 minutes;
}
else if(IPa is in the long-term table, but the registered MAC
        is not MACa){
  /* conflict on IP and MAC mapping */
  send 10 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.
}
```

Figure 1: Short-term cache and long-term table update policy applied on the arrival of gratuitous ARP request packets

is to find the existing GMR-ARP nodes in the same subnet, and to register the IP addresses of the neighbor GMR-ARP nodes in the long-term table only when there is no conflict on those IP addresses. The third goal is to register the address mapping for the new GMR-ARP node in the long-term tables of the neighbor GMR-ARP nodes.

The second step begins 1 sec after transmission of the gratuitous ARP request packet. In the second step, the rebooted or newly attached node sends a special voting request packet for its own IP address to determine if the selected IP address is used by other machines. The new node can use the current IP address without problems, if there is no voting reply packet, or the MAC address of the voting node polls more than 50% of the votes. If there is at least one reply packet and the MAC address of the voting request node polls less than 50%, the node relinquishes the current IP address and requests a new IP address through the DHCP or manual configuration with the help of a network administrator. Therefore, the first goal is met by this procedure.

When other GMR-ARP nodes receive a special voting request message, if the (IP, MAC) mapping for the selected IP address exists in the long-term table, the neighbor nodes reply by broadcasting a single voting reply message containing the mapping. If a neighbor GMR-ARP node does not have (IP, MAC) mapping for the selected IP address, the node replies by broadcasting a voting reply message containing the mapping (IP, 0). Although this mapping does not have any information and is not counted in voting, the voting request node can know the IP and MAC addresses of a neighbor node

through this reply message.

Based on these voting reply messages, the voting request node calculates the ratio of votes supporting its current (IP, MAC) address mapping, and also registers the (IP, MAC) mapping of the neighbor GMR-ARP nodes in its own long-term table. The malicious nodes might attempt to corrupt the long-term table by sending reply messages with a spoofed IP or MAC address. If the voting request node identifies a conflict on some IP address due to IP or MAC spoofed packets by some malicious node, the voting request node leaves the MAC field corresponding to this IP address empty, and resolves the conflict later only when it needs to know the MAC address for this IP address. Therefore, the second goal of registering the obvious (IP, MAC) mapping of the neighbor GMR-ARP nodes in the long-term table can be achieved through this procedure.

The neighbor GMR-ARP nodes, which do not have the (IP, MAC) mapping for the IP address selected by the new GMR-ARP node in the long-term table, monitor the reply packets from the other GMR-ARP nodes for an interval of 1 sec. If they observe only the mapping of (IP, 0), or the MAC address of the voting node polls more than 50% of the votes excluding the votes with the mapping of (IP, 0), then those neighbor GMR-ARP nodes accept the sender protocol and sender hardware address mapping of the special voting request packet into the ARP cache and the long-term table. Otherwise, the mapping is not registered in the long-term table of the neighbor GMR-ARP nodes.

If we consider the subnet without any adversary nodes, then we can easily find that all the existing GMR-ARP nodes can accept the (IP, MAC) mapping of the new GMR-ARP node into the long-term table through the first and second steps of the initialization stage. Therefore, the third goal can be achieved when there is no adversary node. When there are adversary nodes, the good neighbor nodes can learn the correct (IP, MAC) mapping for the new GMR-ARP node through a voting procedure, which will be described later. Fig. 2 summarizes the way in which the GMR-ARP nodes process a special voting request message that queries the MAC address for the IP address of the voting request node.

Therefore, according to the initialization procedure described above, when a GMR-ARP node sends the special voting request packet after reboot or initial deployment, if only attackers reply with a false MAC address for the queried IP address, the new node will attempt to avoid the use of the IP address in conflict. Thus, the fake replies from the malicious nodes are not likely to be helpful in corrupting the ARP caches of the neighbor nodes, particularly for the entry related to the new

```
/* (IP_a, MAC_a): the sender protocol (IP) and hardware (MAC)
          addresses of the received voting request packet,
   IP_g: the target protocol (IP) address of the received packet */

if(IP_g is registered in the long-term table){
  reply by broadcasting the MAC address for IP_g;
}
else{
  reply by broadcasting the mapping of (IP_g,0);
  temporarily store the mapping of (IP_a, MAC_a),
    and monitor the replies from other GMR-ARP nodes for 1 sec;
  if(only trivial mapping of (IP_g,0), or MAC_a polls over 50%)
    accept the new mapping in both short and long term tables;
  else
    drop the mapping;
}
```

Figure 2: Handling of the received special voting request packet whose sender protocol (IP) address is equal to the target protocol (IP) address

GMR-ARP node. On the other hand, if there is no fake reply from the adversary nodes for the special voting request, then all the normal GMR-ARP nodes will accept the mapping between the sender protocol address and the sender hardware address of the special voting request packet by the algorithm in Fig. 2. Consequently, the proposed initialization procedure can effectively protect the ARP cache of existing GMR-ARP nodes from an ARP cache poisoning attack on the entry for a new GMR-ARP node. The ARP cache of a new GMR-ARP node is protected by the second step of the initialization stage and the voting procedure described in the next subsection.

2.2. Voting Scheme

When a new GMR-ARP node wants to know the MAC address corresponding to an IP address that is not its own long-term table after the initialization stage, it basically resolves the mapping based on the normal ARP procedure, i.e. based on the handshake of the ARP request and ARP response messages. However, if the new GMR-ARP node cannot make a decision due to conflict on the mapping, the MAC address in conflict can be resolved by the voting scheme.

Two more types of ARP packets are used, i.e. voting request and voting reply packets, as in MR-ARP [13]. The packet format is the same as that of the ARP request/reply packets. The *operation* field is set to 20 and 21 for the voting request and reply packets, respectively. When Node A wants to find the correct MAC address for the IP address of IP_B, Node A broadcasts a voting request with IP_B in the *target protocol address* field to query the neighbor GMR-ARP nodes about the

(IP, MAC) mapping for that IP. If a neighbor GMR-ARP node receives an ARP voting request for IP_B, it replies by unicasting l ARP voting replies with the (IP, MAC) mapping for IP_B to the voting request node at the maximum rate when it has mapping in the long-term table.

Node A then counts the number of votes from each IP address of the neighbor nodes, and accepts the voting reply messages until the number of the voting reply messages from the node responding the earliest reaches l. All late reply messages are neglected. We consider only the neighbor GMR-ARP nodes who have sent more than or equal to $\zeta l (0 < \zeta < 1)$ voting reply messages in the voting, and each of those nodes can contribute only one vote, i.e. each GMR-ARP node can cast only one vote by sending a sufficiently large number of voting reply messages as soon as possible. After collecting a sufficient number of the voting reply packets, Node A makes a decision based on the majority of votes. The parameters l and ζ need to be determined carefully for reliable operation of the voting procedure, and this issue will be discussed in more detail in the next section.

We attempt to achieve two goals by this mechanism. The first goal is to achieve fairness in voting among the wired nodes even with different processing power, and another goal is to prevent node duplication attack by some adversary nodes. Some malicious nodes might attempt to send more than l voting reply messages while spoofing IP or MAC addresses to increase the ratio of the votes supporting their fake (IP, MAC) mapping, which is called a node duplication attack. However, if the voting reply packets are accepted according to the above policy, it is possible to suppress the node duplication attack. How the proposed voting scheme can resolve the conflict on the owner of a given IP address without vulnerability to a node duplication attack can be explained as follows. The voting request node can be either a wired node or a wireless node. These two cases are considered separately as follows.

2.2.1. Case 1 - Wired voting request node

In this subsection, we consider the case where the voting request node is connected to the subnet through a wire. If all voting-cognizant nodes are wired ones, the fairness can be guaranteed in the Ethernet for the following reason. Ethernet cards normally have the capability to send packets up to the link rate, and the Ethernet switch serves the packets from different ports in a fair manner. Therefore, each node is likely to send a similar number of voting reply packets under this fairness condition as long as every node is transmitting packets. The fairness may not be maintained if any node finishes its own transmission. Accordingly, the fairness might be guaranteed if the voting reply packets are accepted only until the earliest node completes its transmission. Under the assumption that each node, even including the malicious nodes, has only one network card, every node will be treated fairly in this voting, and node duplication attack is unlikely to be successful due to the fairness in voting.

We can also consider the case where some of the voting-cognizant nodes are connected through the wireless medium. The MAC access delay of a wireless node in the 802.11 environment is known to be of the order of tens of msec [20, 21]. On the other hand, it takes only 6 and 0.6 msec to transmit 50 1500-Byte packets on a 100-Mbps and 1-Gbps Ethernet, respectively. Therefore, in this case, the voting is likely to be terminated by the wired nodes before the wireless nodes send a sufficient number of voting reply messages, and the voting result is highly likely to be determined only by the wired nodes. A similar conclusion to the case where the voting-cognizant nodes are all wired nodes can be drawn.

2.2.2. Case 2 - Wireless voting request node

In this subsection, we consider the case where the voting request node is a wireless node. Let us assume that there are non-zero wireless voting-cognizant nodes and non-zero wired voting-cognizant nodes. We assume that the access point (AP) is connected to the Ethernet through a wire. In this case, the wired voting-cognizant nodes can receive the voting request message almost as soon as the voting request message reaches the AP because the transmission delay on the Ethernet is negligibly small compared to the MAC access delay in an 802.11 environment. However, the wireless nodes can receive a voting request message after an additional MAC access delay from the AP to the wireless nodes. Therefore, the voting reply messages from the wired nodes are likely to fill up the buffer in the AP earlier than the packets from the wireless nodes. Consequently, the voting result is likely to be determined by the wired voting cognizant nodes again. The fairness without a node duplication problem can be explained in the same way as in the previous case, i.e. Case 1.

3. Analysis of Voting-related Parameters

As explained in the previous section, it is very important to preserve the fairness among the wired nodes to prevent a node duplication attack. In the voting scheme described in the previous section, if l is too small, the fairness may not be achieved among different wired

nodes. Therefore, it is important to find and use a sufficiently large number for l to achieve reasonable fairness among the wired nodes in voting, and this problem is investigated in detail in this section.

If ζ is less than or equal to 0.5, then the proposed voting scheme might be vulnerable to a node duplication attack again. As an example, let us consider a case where ζ is 0.5. If the voting request node observes $0.5l$ voting reply messages from a neighbor node, then the voting request node will approve one valid vote for the neighbor node. In this case, a malicious node might attempt to contribute two votes by sending two streams of $0.5l$ voting reply packets from two different (spoofed) IP addresses. ζ needs to be maintained higher than 0.5 to reduce the possibility of this problem. In this work, we consider two values, 0.6 and 0.7, for possible values of ζ.

In the proposed voting scheme, each GMR-ARP node sends l voting reply messages for each voting request message, and the voting request node accepts voting reply packets only until the number of the voting reply messages reaches l for the earliest node. M denotes the number of GMR-ARP nodes in the same subnet, and K_i denotes the number of voting reply packets received from Node i ($1 \le i \le M$). One voting experiment finishes when $\max_{1\le i\le M} K_i = l$, and there will be only one node that satisfies the relation: $K_j = l$. Z is a random variable satisfying the relation: $\max_{1\le i\le M} K_i = l = K_Z$. We are interested in the probability that an arbitrary neighbor node can contribute one valid vote for a given value of l, which can be expressed as

$$P_{vv}(l,\zeta) = Pr(K_1 \ge \zeta l | K_Z = l). \tag{1}$$

In the above equation, Node 1 was selected as a tagged node without a loss of generality.

When the neighbor GMR-ARP nodes receive the voting request packet, they will send l voting reply packets at the maximum rate, i.e. usually close to the link rate, to the voting request node simultaneously. In this case, the neighbor nodes naturally contend to access the voting request node, and this contention is usually resolved fairly by the Ethernet switch [13]. To derive a mathematical formula for (1), we model the medium access attempt, to the voting request node, of each neighbor node as an independent Bernoulli experiment. Let p' denote the probability that a tagged node succeeds in transmitting the head-of-line voting reply packet earlier than the other $M-1$ nodes. Each node then has the same medium access probability p'. Because $\sum_{1\le i\le M} p' = 1$, $p' = 1/M$. By a similar reasoning, the following can also be obtained:

$$Pr(Z = i) = 1/M, \quad i = 1, 2, \ldots, M. \tag{2}$$

Then, $P_{vv}(l,\zeta)$ can be expressed as (3). The detailed derivation of this equation is given in Appendix A.

Although (3) is accurate under the assumption of the i.i.d. medium access probability of each neighbor node, the computational complexity is very high due to the large number of combinations for the innermost summations, i.e. the number of the solutions for the indeterminate integer equations, in both the numerator and the denominator. For example, the number of the non-negative integer solutions for the following indeterminate equations is $\binom{M+m-l-2}{m-1}$:

$$x_1 + x_2 + \cdots + x_{M-1} = m - l, \;\; x_i \ge 0 \, (i = 1, 2, \ldots, M-1).$$

Although the number of solutions for the indeterminate equation in (3) is less than for this case due to the upper bound for each element, the number of possible combinations is still high at a similar level. We find that it takes more than an hour to evaluate $P_{vv}(l,\zeta)$ for $\zeta = 0.7$, $M \ge 10$ and $l \ge 50$ on 1.3 GHz dual-core CPU machines.

Therefore, we investigate a new low-complexity approximation for $P_{vv}(l,\zeta)$ that can be used practically for large values of M and l. Let $D_i(t)$ denote the number of all voting reply packet transmissions from all the GMR-ARP nodes until the tagged node i transmits the t-th reply packet successfully. As $D_i(t)$ increases, the tagged node i will spend more time sending the t-th reply message successfully. Thus, $D_i(t)$ can also be considered to measure the delay to deliver a given number of the reply packets successfully. $P_{vv}(l,\zeta)$ can be expressed in terms of $D_i(t)$ as

$$\begin{aligned} P_{vv}(l,\zeta) &= Pr(K_1 \ge \zeta l | K_z = l) \\ &= Pr(D_1(\lceil \zeta l \rceil) \le \min_{1\le i\le M} D_i(l)). \end{aligned} \tag{4}$$

If we assume that $D_i(t)$ and $D_j(s)$ are mutually independent for $i \ne j$, then the following approximation result

$$P_{vv}(l,\zeta) = \frac{1}{M} + \frac{M-1}{M} \sum_{i=\lceil \zeta l \rceil}^{l-1} \frac{\displaystyle\sum_{m=l+i}^{(M-2)*(l-1)+l+i} \left(\frac{1}{M}\right)^m \binom{m-1}{l-1}\binom{m-l}{i} \sum_{\substack{x_2+\ldots+x_{M-1}\\ =m-l-i,\\ 0\le x_u\le l-1}} \prod_{2\le j\le M-1} \binom{m-l-i-\sum_{2\le k<j} x_k}{x_j}}{\displaystyle\sum_{m=l}^{(M-1)*(l-1)+l} \left(\frac{1}{M}\right)^m \binom{m-1}{l-1} \sum_{\substack{x_1+\ldots+x_{M-1}=m-l,\\ 0\le x_u\le l-1\ (1\le u\le M-1)}} \prod_{1\le j\le M-1} \binom{m-l-\sum_{1\le k<j} x_k}{x_j}} \tag{3}$$

can be obtained:

$$\begin{aligned} P_{vv}(l,\zeta) = {} & \frac{1}{M} + \\ & \sum_{i=l}^{\infty}\Bigg[\left\{1-\sum_{j=l}^{i-1}\binom{j-1}{l-1}\left(\frac{1}{M}\right)^l\left(1-\frac{1}{M}\right)^{j-l}\right\}^{M-1} \\ & -\left\{1-\sum_{j=l}^{i}\binom{j-1}{l-1}\left(\frac{1}{M}\right)^l\left(1-\frac{1}{M}\right)^{j-l}\right\}^{M-1}\Bigg] \\ & \times \sum_{k=\lceil \zeta l\rceil}^{i}\binom{k-1}{\lceil \zeta l\rceil-1}\left(\frac{1}{M}\right)^{\lceil \zeta l\rceil}\left(1-\frac{1}{M}\right)^{k-\lceil \zeta l\rceil} \\ & \times\left\{1-\sum_{j=l-\lceil \zeta l\rceil}^{i-k}\binom{j-1}{l-\lceil \zeta l\rceil-1}\left(\frac{1}{M}\right)^{l-\lceil \zeta l\rceil}\left(1-\frac{1}{M}\right)^{j-(l-\lceil \zeta l\rceil)}\right\}. \end{aligned} \tag{5}$$

The detailed derivation of the above equation is given in Appendix B.

Fig. 3 shows the change of the probability $P_{vv}(l,\zeta)$, i.e. the probability that a neighbor GMR-ARP node can contribute to the voting by sending a sufficient number (ζl) of the voting reply packets, for a range of values of M and l when ζ is fixed to 0.7. We can easily know that the fairness improves as this probability increases. The fairness cannot be guaranteed when l is very small regardless of the value of M, i.e. the number of neighbor GMR-ARP nodes. On the other hand, the probability approaches 0.9 as l increases to 100. Thus, a reasonable level of fairness might be achieved when the value of l is set to 100.

Although the approximation result by (5) does not agree with the accurate analysis result by (3), the error was found to be less than 10% in all the cases, and less than 5% when $l \geq 50$. Therefore, the approximation equation might be used to find the tendency of the probability for large values of l and M. The non-monotonic trend of the probability $P_{vv}(l,\zeta)$ comes from the discontinuity of $\lceil \zeta l \rceil$.

A large value of l can lead to high voting traffic overheads during the voting period. Thus, it is better if the value of l can be reduced further. Fig. 4 shows the

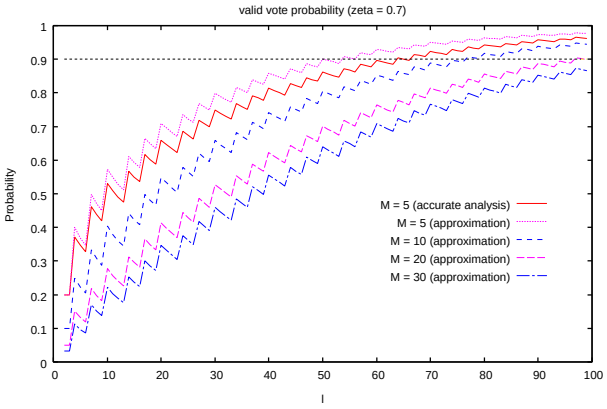


Figure 3: $P_{vv}(l,\zeta) = Pr(K_1 \geq \zeta l | K_z = l)$ for various values of l and M ($\zeta = 0.7$)

change of the probability $P_{vv}(l,\zeta)$ for a smaller value of ζ: $\zeta = 0.6$ in this case. We find that the valid voting probability approaches 0.9 even when l is around 50. If the value of ζ is lowered further, the value of l might be reduced. However, if ζ is lower than 0.5, the scheme might be more vulnerable to the node duplication attack. Therefore, we use 0.6 for the value of ζ hereafter, and the value of l is fixed to 50.

The reason why we attempt to maintain the probability $P_{vv}(l,\zeta) = Pr(K_1 \geq \zeta l | K_z = l)$ at least around 0.9 can be explained as follows. Let us assume that there are N_g good (GMR-ARP) nodes and N_b malicious nodes around a new GMR-ARP node. Let N_1 and N_2 denote the number of the good nodes and bad nodes that contribute to the voting successfully by sending a sufficient number of voting reply packets. Let us assume the probability that a neighbor node contributes to the voting successfully is p, and the success of one node is independent of the success of the other nodes. Then, p is identical to $P_{vv}(l,\zeta)$ discussed above, and N_1 and N_2 become independent. Under this condition, we calculate the probability that a voting request node makes an incorrect decision, P_{fd}. We can easily know that N_1 and

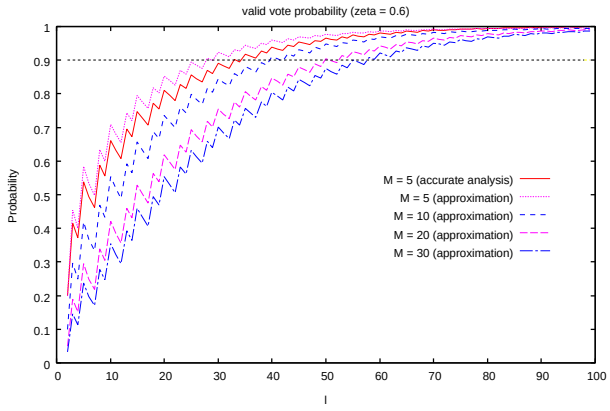


Figure 4: $P_{vv}(l, \zeta) = Pr(K_1 \geq \zeta l | K_z = l)$ for various values of l and M ($\zeta = 0.6$)

N_2 have the following binomial distribution:

$$Pr(N_1 = n) = \binom{N_g}{n} p^n (1-p)^{N_g - n}, \\ Pr(N_2 = s) = \binom{N_b}{s} p^s (1-p)^{N_b - s}. \tag{6}$$

The false decision probability P_{fd} can then be expressed as

$$P_{fd} = Pr(N_1 < N_2) \\ = \sum_{i=0}^{N_b - 1} \binom{N_g}{i} p^i (1-p)^{N_g - i} \sum_{j=i+1}^{N_b} \binom{N_b}{j} p^j (1-p)^{N_b - j}. \tag{7}$$

Fig. 5 shows the false decision probability P_{fd} when N_g is larger than N_b by only 1. According to Fig. 5, the false decision probability can be maintained at less than 0.05 only when $p \geq 0.98$. However, such a large value of p is not required if N_g is sufficiently larger than N_b, as shown in Fig. 6. Fig. 6 shows the false decision probability as N_b increases for a given value of N_g when p is fixed to 0.9. We can observe that the false decision probability can be maintained very low if N_b is not close to N_g, even though p is 0.9. Therefore, although the value of l is decided such that p or $P_{vv}(l, \zeta)$ is close to 0.9, it is expected that the false decision probability can be maintained low provided the number of good GMR-ARP nodes is sufficiently larger than the number of malicious nodes.

4. Early Packet Dropping to Improve the Fairness of Voting

Although the packet transmission rate is uniform among the different wired nodes belonging to the same

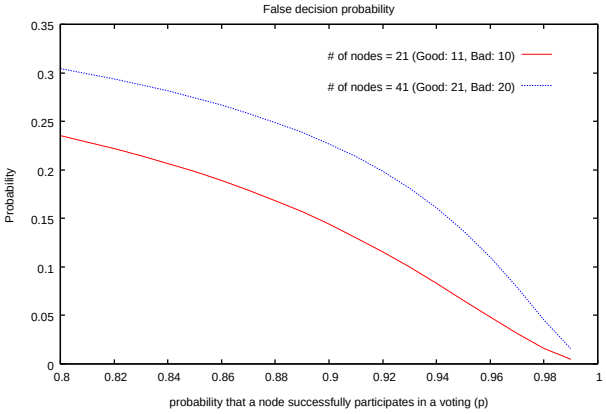


Figure 5: False decision probability (P_{fd}) when the number of good nodes (N_g) is larger than the number of malicious nodes (N_b) by 1

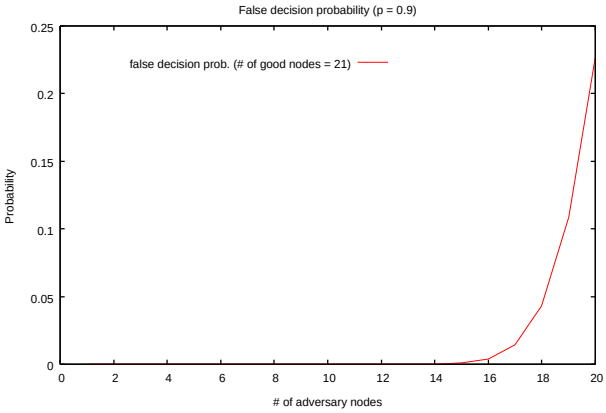


Figure 6: False decision probability (P_{fd}) as the number of malicious nodes (N_b) increases when the number of good nodes (N_g) is fixed to 21

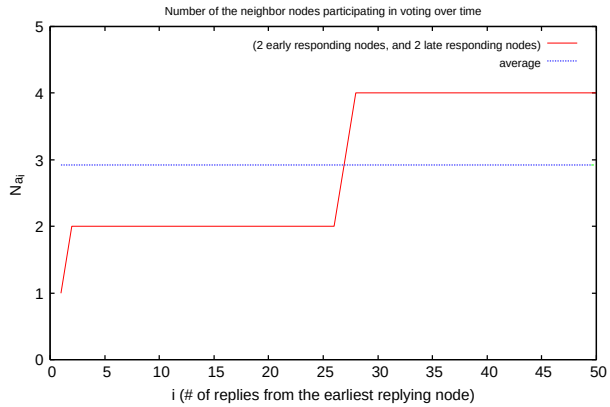


Figure 7: Change of N_{a_i} over the various values of i, i.e. the number of the voting reply messages from the earliest replying node

subnet, the voting reply messages of the different nodes cannot arrive at the voting request node simultaneously when the voting request packet has been sent. As a worst case, if the voting reply packets of a single malicious node arrive much earlier than the reply packets of good neighbor nodes, then a single attacker might win the voting even in the presence of multiple GMR-ARP nodes. We attempt to recover the impaired fairness in such a case by dropping the too early reply messages.

To derive the condition to determine the too early packet arrival, we consider an example case where 3 GMR-ARP nodes and 2 attacker nodes are connected through the Ethernet. We performed an experiment where one GMR-ARP node sends a voting request message, and each of the other 4 nodes responds by transmitting 50 voting reply messages. $a_i (i = 1, 2, \ldots, 50)$ denotes the time when the i-th reply message of the earliest replying node arrives at the voting request node. The *earliest replying node* means the node that delivers the 50 reply messages to the voting request node the earliest. N_t denotes the number of the neighbor nodes that have sent at least one voting reply message up to time t. Fig. 7 shows the change of N_{a_i} obtained from the experiment.

In this experiment, the earliest replying node is one of the attacker nodes, and two attacker nodes respond much earlier than two GMR-ARP nodes. In more detail, the replies from the GMR-ARP nodes arrived only after the arrival of the 26-th reply packet of the earliest attacker node, and the numbers of the reply messages from the two GMR-ARP nodes were measured to be less than 30. Because we fixed the values of l and ζ to 50 and 0.6, respectively, those two GMR-ARP nodes cannot contribute any valid vote to the voting, and the early replying attacker can win the voting even without the other attacker.

To resolve this unfairness problem by the difference in the arrival times of the reply packets of the different nodes, we discard too early reply packet arrivals in the following way. The following three parameters are calculated from N_{a_i}:

$$\begin{aligned} N_v &= \frac{1}{l}\sum_{i=1}^{l} N_{a_i}, \\ I_{th}^1 &= \max\left\{ j \,|N_{a_j} \le N_v, j \ge 1 \right\}, \\ I_{th} &= \min\left\{ I_{th}^1, (1-\zeta)l \right\}. \end{aligned} \tag{8}$$

N_v is simply the average of N_{a_i}. We can easily know that N_{a_i} is a non-decreasing function with respect to i, and I_{th}^1 finds the crossing point between the graph, $y = g(i) = N_{a_i}$, and the constant line $y = N_v$. The arrivals ahead of that cross-point are considered as too early arrivals, and those arrivals are discarded. To avoid the case where most of the packet arrivals of the earliest replying node are discarded, a lower bound I_{th} is calculated by the third relation of (8), and the arrival time of the $(I_{th} + 1)$-th packet of the earliest replying node is set to the threshold to discriminate the too early arrivals. In more detail, we only consider the voting reply packets which arrived in the interval of $[a_{I_{th}+1}, a_l]$ as valid arrivals. Then, the number of the valid voting packet arrivals from the earliest replying node is guaranteed to be at least ζl, i.e. 30 when $\zeta = 0.6$ and $l = 50$.

In this case, the rule to count valid votes needs to be modified slightly as follows. The maximum of the valid replies from each node (l') is calculated again after the too early arrivals are discarded. If the number of replies from a specific node is equal to or larger than $\zeta l'$, then that node contributes one valid vote. In the above sample experiment, the numbers of the valid voting replies from the GMR-ARP nodes were 25 and 24, and the numbers of the valid voting replies from the attacker nodes were 49 and 50. When the above filtering scheme was applied to this experiment result, the numbers of the valid replies for the GMR-ARP nodes did not change because those replies arrived rather late. On the other hand, the numbers of the valid replies for the attackers decreased to 29 and 30, respectively. Therefore, the above filtering, i.e. rejecting the too early arrivals, helps to improve the fairness in this example case.

4.1. Analysis of the impact of the filtering scheme on the voting fairness

In this subsection, we analyze the impact of the filtering scheme on the voting fairness in more detail. When

a voting request packet is transmitted, if all the other GMR-ARP nodes start to respond simultaneously, the filtering scheme explained above need not be applied since the fairness is good already. Thus, we consider the case where the response of the different neighbor nodes is not synchronized in this subsection. In more detail, the considered environment is described by the following assumptions.

- The number of the neighbor GMR-ARP nodes excluding the voting request node is N, and all the GMR-ARP nodes are connected through the same Ethernet switch.
- The Ethernet switch serves the packets from each GMR-ARP node in a fair manner, e.g. using round-robin algorithm.
- When $t_i^*(1 \le i \le l)$ denotes the transmission time of the i-th reply message of the earliest replying node, the response start times, i.e. the transmission times of the first reply messages, of the other neighbor nodes are uniformed distributed in $[t_1^*, t_l^*]$.

N_e is a random variable denoting the number of neighbor nodes contributing a valid vote to the voting. We attempt to measure the improvement in the voting fairness based on $E[N_e]$, since $E[N_e]$ increases as the fairness in voting improves.

If one neighbor node transmits the first reply message at t $(t_1^* \le t \le t_l^*)$, the number of replies that can arrive in time from this node becomes $\lfloor (t_l^* - t)/(t_l^* - t_1^*) \times (l-1) \rfloor$, or $\lfloor (t_l^* - t)/(t_l^* - t_1^*) \times (l-1) \rfloor + 1$ under the above assumptions. The earliest replying node can send l reply messages, but the other nodes cannot send more than $l-1$ messages by the definition of the earliest replying node. To simplify the analysis, we approximate the number of valid replies for this node as

$$\hat{K}(t) = \frac{t_l^* - t}{t_l^* - t_1^*} \times (l-1). \tag{9}$$

We assume that Node 1 is the earliest replying node among N neighbor GMR-ARP nodes without loss of generality. When the packet filtering scheme is not used, $E[N_e]$ can be expressed as

$$E[N_e] = 1 + \sum_{i=2}^{N} 1 \times \Pr(K_i \ge \zeta l), \tag{10}$$

where K_i is the number of voting reply packets received from Node i $(1 \le i \le N)$.

Under the uniform distribution assumption for the response start time of the neighbor nodes, $\Pr(K_i \ge \zeta l)(i \ge 2)$ can be expressed as follows using the approximation of (9):

$$\begin{aligned} \Pr(K_i \ge \zeta) &= \int_{t_1^*}^{t_l^*} \Pr(K_i \ge \zeta l \,|t = \tau) \frac{1}{t_l^* - t_1^*} d\tau \\ &\simeq \int_{t_1^*}^{t_l^*} \Pr(\hat{K}(t) \ge \zeta l \,|t = \tau) \frac{1}{t_l^* - t_1^*} d\tau \\ &= 1 - \left(\frac{l}{l-1}\right)\zeta. \end{aligned} \tag{11}$$

Combining (10) and (11) yields

$$E[N_e] = \left(1 - \left(\frac{l}{l-1}\right)\zeta\right) N + \left(\frac{l}{l-1}\right)\zeta. \tag{12}$$

Hereafter, we investigate N_v, I_{th}^1, I_{th} of (8) in more detail. $E[N_v]$ can be expressed as

$$E[N_v] = E\left[\frac{1}{l}\sum_{i=1}^{l} N_{a_i}\right] = \frac{1}{l}\sum_{i=1}^{l} E[N_{a_i}]. \tag{13}$$

Since we assume that every node transmits the reply packets at the link rate, we have

$$t_i^* = t_1^* + (t_l^* - t_1^*) \times \frac{i-1}{l-1}. \tag{14}$$

Since N_{a_i} is equivalent to the number of neighbor nodes that started the response before the transmission of the i-th reply packet from the earliest replying node, the distribution of N_{a_i} can be obtained as

$$\begin{aligned} \Pr(N_{a_i} = j) &= \binom{n-1}{j-1} \left(\frac{t_i^* - t_1^*}{t_l^* - t_1^*}\right)^{j-1} \left(\frac{t_l^* - t_i^*}{t_l^* - t_1^*}\right)^{n-j} \\ &= \binom{n-1}{j-1} \left(\frac{i-1}{l-1}\right)^{j-1} \left(1 - \frac{i-1}{l-1}\right)^{n-j}, \end{aligned} \tag{15}$$

where the second equality is valid by (14). Since $E[N_{a_i}] = \sum_{j=1}^{N} j \cdot \Pr(N_{a_i} = j)$, $E[N_{a_i}]$ can be expressed as follows using the above relation:

$$E[N_{a_i}] = 1 + (N-1)\frac{i-1}{l-1}. \tag{16}$$

Combining (13) and (16) yields

$$E[N_v] = \frac{N+1}{2}. \tag{17}$$

To simplify the analysis, we approximate N_v by $(N+1)/2$.

$E[I_{th}^1]$ can be expressed as

$$E[I_{th}^1] = \sum_{i=1}^{l} \Pr(I_{th}^1 \ge i). \tag{18}$$

Since N_{a_j} is a non-decreasing function with respect to j, we can obtain from the definition of I^1_{th}

$$\begin{aligned}\Pr(I^1_{th} < i) &= \Pr(\max\{j \,|N_{a_j} \le N_v, j \ge 1\} < i) \\ &= \Pr(N_{a_i} > N_v).\end{aligned}$$

Because of the approximation of $N_v \approx (N+1)/2$, the above relation can be changed into

$$\Pr(I^1_{th} < i) \simeq \Pr(N_{a_i} > (N+1)/2). \tag{19}$$

Combining (15), (18), and (19) yields

$$E[I^1_{th}] = l - \sum_{i=1}^{l} \sum_{(N-1)/2<k\le N-1} \binom{N-1}{k} \left(\frac{i-1}{l-1}\right)^k \left(1-\frac{i-1}{l-1}\right)^{(N-1)-k}.$$

It is possible to simplify the above relation further and get the following results:

$$E[I^1_{th}] = \begin{cases} \frac{l}{2}, & \text{for even } N, \\ \frac{l}{2} + \frac{1}{2}\sum_{i=1}^{l} \binom{N-1}{(N-1)/2} \left(\frac{i-1}{l-1}\right)\left(1-\frac{i-1}{l-1}\right)^{(N-1)/2}, & \text{for odd } N. \end{cases} \tag{20}$$

Thus, we find that $E[I^1_{th}] \ge 0.5l$. Since ζ is fixed to 0.6, $(1-\zeta)l = 0.4l$, and it is likely that $I_{th} = (1-\zeta)l$ by (8).

Then, the number of the valid reply messages from the earliest replying node will be ζl, i.e. 30, by the reason given below (8). Since too early packets are dropped when the filtering scheme is used, l', i.e. the maximum of valid replies from each node, will be ζl. If the number of the valid replies from a given GMR-ARP node is equal to or larger than $\zeta l' (= \zeta \times \zeta l = \zeta^2 l)$, then that node contributes one valid vote. If the number of the replies from a specific node is less than l', no packet dropping is expected for that node, because the response start time of that node is highly likely to be later than $a_{(1-\zeta)l+1}$, i.e. the threshold for the early packet filtering. If we let N^f_e denote the number of the neighbor nodes contributing a valid vote to the voting when the filtering scheme is used, $E[N^f_e]$ can be described as

$$E[N^f_e] = 1 + \sum_{i=2}^{N} 1 \times \Pr(K_i \ge \zeta^2 l).$$

The following relation can be easily derived in the same way as (12) has been derived for the non-filtering case:

$$E[N^f_e] = \left(1 - \left(\frac{l}{l-1}\right)\zeta^2\right)N + \left(\frac{l}{l-1}\right)\zeta^2.$$

When $N = 10$, $E[N_e] = 4.5$ and $E[N^f_e] = 6.7$. Thus, the number of the contributing neighbor nodes increases by about 49% due to the improvement in the fairness of voting by the early packet filtering.

5. Numerical Results

In this section, we investigate the performance of the proposed address resolution protocol, i.e. GMR-ARP, and compare it with the performance of other voting-based schemes, i.e. MR-ARP and EMR-ARP, in terms of the voting traffic overhead, reliability in the presence of the attackers, and voting procedure delay.

5.1. Analysis of Traffic Overhead

The traffic overhead of GMR-ARP is calculated and compared with the traffic overhead of MR-ARP and EMR-ARP. Because the traffic overhead of the voting-based scheme is due to the exchange of extra packets during the initialization procedure and the exchange of the voting request and reply packets during the voting procedure, we focus on those components to analyze the additional traffic overhead.

We consider a simple case with several assumptions to simplify the analysis. The IP addresses are assumed to be allocated through DHCP for all nodes in the same subnet. We assume that there are M GMR-ARP nodes, and $A_i (i = 1, 2, \ldots, M)$ denotes the i-th GMR-ARP node. Initially, A_i has the IP address of $IP_i (i = 1, 2, \ldots, M)$ assigned through DHCP, and we assume that each node changes its IP address in the following manner. Whenever a node is rebooted, a new IP address is assigned to the rebooted machine, and that IP address is selected from a pool of IP addresses, $\{IP_1, IP_2, \ldots, IP_{M+1}\}$. We also assume that the rebooting time of one machine never overlap with that of the other machines, and the rebooting time is negligibly small compared to the lifetime of each machine. We assume that each machine is rebooted at the same interval of T_D, but asynchronously. In addition, it is assumed that there is no attacker in the subnet to focus on the traffic overhead under normal conditions.

Let us consider the case where Node A_{M-1} changes its IP address from IP_{M-1} to IP_M after A_M changes its IP address from IP_M to IP_{M+1} to calculate the voting procedure-related traffic rate to Node A_{M-1}. The initialization stage of GMR-ARP consists of two steps: the first step is related to the gratuitous ARP request message, and the second step is related to the special voting request message. We first count the number of packets that Node A_{M-1} receives regarding the first step. A_{M-1} receives $(M-1)$ gratuitous ARP request messages from the other $(M-1)$ GMR-ARP nodes, i.e. one gratuitous ARP request message from each rebooted GMR-ARP node. A_{M-1} will send 10 unicast ARP request messages to the previous owner node whenever an existing IP is assigned to a newly rebooted node, but, there will be no

reply if the owner of the IP address is changed legitimately by the DHCP. On the other hand, when the IP address that has been used by A_{M-1}, i.e. IP_{M-1}, is assigned to some other node, e.g. A_{M-2}, then each neighbor GMR-ARP node will send 10 unicast ARP request messages to A_{M-1} as A_{M-1} was the previous owner of IP_{M-1}. Because there are $(M-1)$ neighbor GMR-ARP nodes, the aggregate number of unicast ARP requests destined to A_{M-1} is $10(M-1)$. Therefore, the number of packets that Node A_{M-1} receives regarding the first step is $11(M-1)$ ARP request messages.

Regarding the second step of the initialization stage, A_{M-1} receives $(M-1)$ voting reply messages for the special voting message sent by the node itself. A_{M-1} also receives $(M-1)$ special voting request messages, i.e. one special voting request from each rebooted machine, and $(M-2)(M-1)$ voting reply messages, i.e. $(M-2)$ voting reply messages for each special voting request message. Because the voting request and reply messages have the same format as the ARP request message in GMR-ARP, the mean overhead traffic rate to Node A_{M-1} can be expressed as

$$r_A^{\text{GMR-ARP}} = \frac{\{11(M-1) + M(M-1)\} \times 28 \times 8(\text{bits})}{T_D(\text{sec})} = \frac{0.224(M+11)(M-1)}{T_D}\text{Kbps}. \tag{21}$$

Although GMR-ARP attempts to maintain the (IP, MAC) mapping only for the GMR-ARP nodes, MR-ARP and EMR-ARP maintain the (IP, MAC) address mapping for all alive machines. If L denotes the total number of alive machines in the subnet considered, then $L \geq M$, and the voting-related traffic overhead r_A^{MR-ARP} and $r_A^{EMR-ARP}$ for MR-ARP and EMR-ARP, respectively, are known as [15]:

$$r_A^{\text{MR-ARP}} = \frac{11.4(L-1)(M-1)}{T_D}\text{Kbps}. \tag{22}$$

$$r_A^{\text{EMR-ARP}} = \frac{\{1.15L + 0.48(M-2)\}(M-1)}{T_D}\text{Kbps}, \tag{23}$$

A comparison of (21), (22) and (23) shows that the voting traffic overhead of GMR-ARP is lower than the traffic overhead of MR-ARP or EMR-ARP when $M \geq 3$. The gap between the overhead of GMR-ARP and that of MR-ARP or EMR-ARP decreases as L approaches M. Thus, let us consider the case where $L = M$. In this case, if we compare the overhead of those three schemes as M increases, then we can obtain

$$\lim_{M\to\infty} \frac{r_A^{\text{MR-ARP}}}{r_A^{\text{GMR-ARP}}} = \frac{11.4}{0.224} = 50.9,$$
$$\lim_{M\to\infty} \frac{r_A^{\text{EMR-ARP}}}{r_A^{\text{GMR-ARP}}} = \frac{1.63}{0.224} = 7.3.$$

Therefore, the voting traffic overhead is reduced in GMR-ARP compared to the other voting-based schemes.

5.2. Analysis of Reliability of the Proposed Scheme

In this section, we evaluate the reliability of the three voting-based schemes, i.e. the proposed GMR-ARP, EMR-ARP and MR-ARP, through the metric of the false decision probability. In the voting-based ARP scheme, the voting procedure is triggered when a conflict occurs on the (IP, MAC) address mapping by the fake voting reply of an attacker. In this case, if the voting request node selects an incorrect MAC address for a given IP address, the voting request node is considered to have made a false decision [1].

GMR-ARP, EMR-ARP and MR-ARP were implemented by modifying the ARP-related code of Fedora 9 Linux (kernel 2.6.25). EMR-ARP can work reliably only under the assumption that the computation power of different machines is not significantly different, i.e. no more than by a factor of 2 [15]. To test the performance of GMR-ARP when this assumption is not valid, we use the machines with a higher CPU performance, i.e. 2.66 GHz quad-core PCs, for the attacker nodes, and the machines with a lower CPU performance, 1.4 GHz Intel Celeron CPU netbooks, for the good neighbor nodes. The quad-core machines were used for the gateway router and the voting request node. The gateway router is always connected to the Ethernet through a 100 Mbps LAN card and LAN switch. Other machines can be connected to the same subnet through either a 100 Mbps LAN card, or a USB-type wireless LAN card supporting 802.11n.

To investigate if the proposed scheme, i.e. GMR-ARP, overcomes the limitation of the existing voting-based ARP schemes, i.e. EMR-ARP and MR-ARP, we consider the four scenarios described in Table 1. In the table, the number of good nodes implies the number of good neighbor GMR-ARP, EMR-ARP, or MR-ARP nodes that can understand and react to the voting request

[1] If the voting request node cannot make a decision due to the same number of votes supporting either the correct MAC address or the fake MAC address, then this case is not considered as a false decision.

messages. The number of good wired nodes is at least one in every scenario, because it was assumed that the new ARP code is deployed to the gateway router already in Section 2.

In Scenario 1, there is no wireless node. The number of good wired nodes is fixed to 6, and the number of malicious nodes is changed from 1 to 5. The voting request node is connected to the subnet through a 100 Mbps LAN card. The assumption for MR-ARP described in Section 2 is valid in this case. On the other hand, the assumption for EMR-ARP described in Section 2 is not valid in this and subsequent scenarios because the puzzle computation time on a high performance CPU machine is less than half of the computation time on a low performance CPU machine. In Scenario 2, there is only one wired good node, i.e. the gateway router, and there are 5 wireless good nodes. The attackers are connected to the subnet only through a wireless link, and the number of the attackers changes from 1 to 5. The voting request node is connected to the subnet through a wireless link. The assumptions for MR-ARP are not satisfied in this scenario. In Scenario 3, there is only one good neighbor node, that is connected to the subnet through a wire. The number of the wireless attackers changes from 1 to 5, and the voting request node is connected to the subnet through a wireless link. The assumptions for MR-ARP are invalid. We consider Scenario 4 to investigate the case where the attackers attempt to increase their impact on the voting by running multiple virtual machines (VMs) on each physical machine (PM). In Scenario 4, all the nodes are wired nodes, which is similar to the case of Scenario 1. We use quad-core machines for the malicious nodes, and the number of VMs on each malicious node is limited to the number of cores, i.e. 4. The number of VMs on the malicious nodes changes from 1 to 8 with two PMs for the malicious nodes. The number of good wired nodes is fixed to 6. Although the netbook machines are used for the good nodes in the scenarios 1, 2, and 3, 2.33 GHz dual-core machines are used for the good nodes in Scenario 4 to get rid of the effect by huge difference among CPU performance of different machines for the EMR-ARP scheme.

Fig. 8 compares the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 1. The experiment was run 25 times for each value of the number of the attackers. We find that the false decision probability is maintained at zero for GMR-ARP. On the other hand, the probability increases up to 1.0 for both EMR-ARP and MR-ARP as the number of attackers increases. As discussed previously, the assumptions for EMR-ARP are not valid in this scenario due to the difference between the computation power of the good nodes, i.e. netbook machines, and that of the malicious nodes, i.e. quad-core PCs. If the response time of the earliest responding node is t_1, the voting request node waits only up to $2t_1$ from the voting request transmission time, and makes a decision based on the replies that arrived on time. The CPUs of the attacker nodes have a similar performance to that of the voting request node. However, the CPUs of the good neighbor nodes, i.e. netbook machines, have much lower performance than of the attacker nodes. The replies from the good neighbor nodes do not arrive within the deadline of $2t_1$, and thus, the good neighbor nodes cannot contribute to voting except the gateway router.

Table 1: Test scenarios to evaluate the performance of GMR-ARP, EMR-ARP, and MR-ARP

Scenario	# of good wired nodes	# of good wireless nodes	# of bad wired nodes	# of bad wireless nodes
1	6	0	$1-5$	0
2	1	5	0	$1-5$
3	1	0	0	$1-5$
4	6	0	$1-8$ VMs* ($1-2$ PMs)	0

(* VM: Virtual Machine, PM: Physical Machine)

The assumptions for MR-ARP are valid in this case. However, the false decision probability increases to 1.0 for MR-ARP. We find that the voting replies from the bad machines begin to arrive at the voting request node much earlier than those from the good machines. Although the number of good nodes is 6 in this scenario, the number of replies arriving early from the bad machines were sufficient to reverse the decision when the number of attackers is 4 or more. The GMR-ARP scheme discards the reply messages arriving too early according to the algorithm described in Section 4. Therefore, there was no problem due to the early arrival of replies from the attackers.

Fig. 9 compares the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 2. The trends for GMR-ARP and EMR-ARP are similar to those observed in Scenario 1, i.e. in Fig. 8. Although the total number of good nodes is larger than that of bad nodes, EMR-ARP does not work well because the replies from the bad machines arrive much earlier than those from the good neighbor nodes due to the higher performance CPU. In Scenario 2, the voting request node is a wireless node. If T_w represents an average packet delay on a wireless link, the gateway router can receive the voting request message only after T_w from

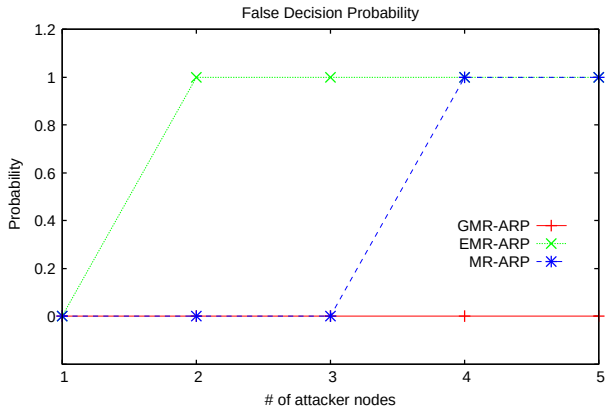


Figure 8: Comparison of the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 1

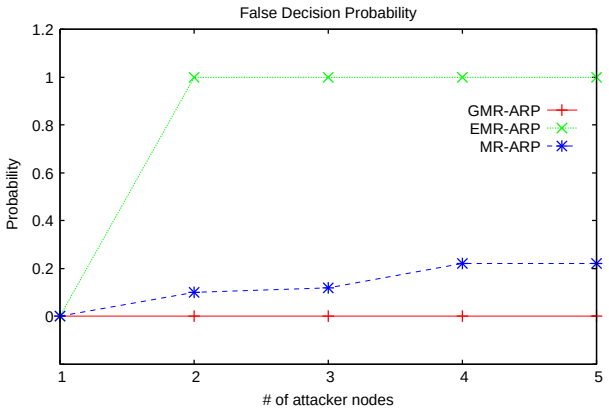


Figure 9: Comparison of the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 2

the time when the voting request node attempts to send the voting request message. On the other hand, other wireless neighbor nodes can receive the voting request message only after $2T_w$ on average. Because the gateway router is connected to the subnet through a wire, the voting reply packets from the gateway router are likely to fill up the buffer in the AP(Access Point) much earlier than the packets from the other wireless neighbor nodes. Because the replies from the gateway router arrives at the voting request node much earlier than the replies from the other nodes, GMR-ARP can make a correct decision early based only on the replies from the gateway router. Therefore, the false decision probability remains low for GMR-ARP, even though the number of attackers increases.

On the other hand, the voting request node needs to wait until it collects 120 messages before making a decision in the MR-ARP. Fifty of them will be the replies from the gateway router. On the other hand, we find that the total number of replies from the attackers sometimes exceeds the total number of replies from the good nodes because the numbers of replies from the different wireless nodes are not always uniform because of the short-term unfairness of IEEE 802.11 MAC protocol [22–24]. This is the reason for the non-zero false decision probabilities for MR-ARP.

Fig. 10 compares the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 3, and the behavior of GMR-ARP and EMR-ARP can be explained in the same manner as that for Fig. 9. However, MR-ARP shows the worst performance among all the scenarios. The reason can be explained as follows. The voting request node needs to wait until it collects 120 voting reply messages in MR-ARP. The gateway

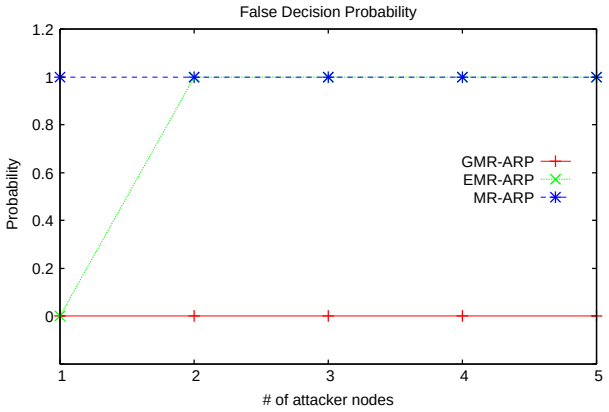


Figure 10: Comparison of the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 3

router, however, is the only good node, and it will send only 50 voting reply messages. In our experiment, we assume that the attacker does not follow the rule precisely and might send more than 50 reply packets for his/her benefit. Although there is only 1 attacker, the attacker sends more than 50 messages, 120 messages in the current experiment. Therefore, the attackers always win the voting in Scenario 3. On the other hand, there was no such problem in GMR-ARP, because the voting request node refused to accept any more packets if there is any node that has sent 50 messages. In this case, the gateway router is the earliest replying node. Hence, the voting request node makes a correct decision based on the replies from the gateway router.

Fig. 11 compares the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 4. In this scenario, all the nodes are connected to the subnet through wire, which is similar to the case of Sce-

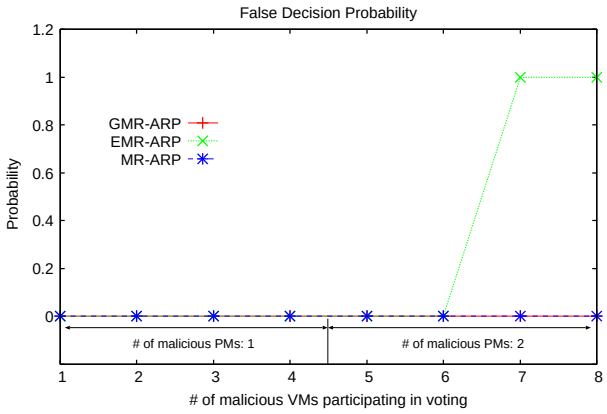


Figure 11: Comparison of the false decision probabilities of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 4

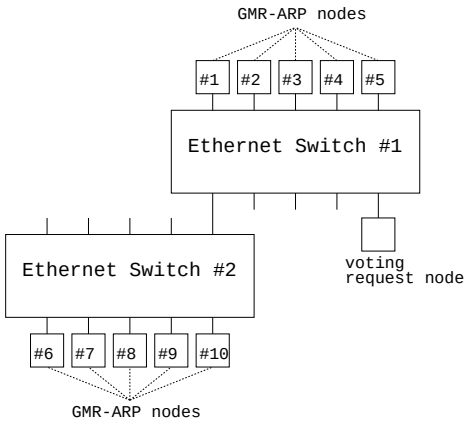


Figure 12: Example network topology to investigate the scalability of GMR-ARP

nario 1. Although the netbook machines were used for the good neighbor nodes in the scenarios 1, 2, and 3, we use dual-core CPU machines for the good nodes in this scenario to exclude the effect of low performance CPU on the performance of EMR-ARP. We use two PMs for the malicious nodes, and each of them has a quad-core CPU and runs up to four VMs to increase its own impact on the voting. We find that GMR-ARP and MR-ARP have zero false decision probabilities. Although there are four VMs on each PM, all those VMs share the same network card. Thus, the total number of voting reply messages sent by each PM cannot be increased significantly compared to the case of no VMs on each PM. The number of network cards or PMs is more important than the number of VMs for GMR-ARP and EMR-ARP.

On the other hand, EMR-ARP suffers from high false decision probability when the number of malicious VMs exceeds the number of good nodes. We find that four cores provides sufficient computation power to four VMs on average so that each VM can provide a valid vote after solving the corresponding puzzle. Although the number of valid votes from the good nodes is fixed 6, which is the number of good nodes, the number of valid votes from the malicious nodes can reach up to 8, which is the maximum number of malicious VMs participating in voting. The false decision probability increases to 1 when the number of malicious VMs exceed 6, especially for EMR-ARP. Thus, EMR-ARP cannot distinguish a VM with a sufficient computation power from a PM with no VM.

We consider one more scenario to investigate the scalability of the proposed scheme, i.e. GMR-ARP. It is not easy to recruit hundreds or thousands of physical machines to test the performance of GMR-ARP in a large-scale subnet. However, it is possible to infer how GMR-ARP will work in a large-scale subnet using multiple LAN switches and a limited number of machines. Fig. 12 describes the test environment considered to investigate the scalability of GMR-ARP. In Fig. 12, all the nodes are GMR-ARP nodes, and the five upper nodes from 1 to 5 are connected to the voting request node through only one Ethernet switch, and these nodes are referred to as close neighbor nodes. On the other hands, five lower nodes from 6 to 10 are two switches away from the voting request node, and they are referred to as far neighbor nodes. We use netbook machines for the close nodes, and quad-core CPU machines for the far nodes. The number of close nodes is fixed to 5, and the number of far nodes is changed from 1 to 5.

Fig. 13 shows the number of the valid voting reply messages from each neighbor node obtained under the condition of Fig. 12. The numbers of the far neighbor nodes are 1, 3, and 5 in Figs. 13(a), 13(b), and 13(c), respectively. From Fig. 13(a), 13(b), and 13(c), we find the number of the valid voting replies from the close nodes remains stable when the number of far nodes increases. On the other hand, the number of the valid voting replies from the far nodes decreases significantly as the number of far nodes increases. This can be explained as follows. In Fig. 12, all the close nodes and the voting request node are connected through the same switch, i.e. Ethernet switch #1, with a dedicated port for each node. However, the far nodes share a single port of Ethernet switch #1 to get connected to the voting request node. Since Ethernet switches serve their ports in a fair manner, the service rate of Ethernet switch #1 for each far node tends to decrease inversely proportional to the number of the far nodes sharing the same port.

Thus, the number of the valid voting replies from each far node decreases as the number of far nodes increases. From this experiment, we find that the voting process gets more dependent on the close neighbor nodes as the number of far nodes increases. Thus, GMR-ARP is likely to prevent ARP poisoning-based MITM attacks even in a large-scale subnet as far as *Assumption 5* of Section 2 is valid among the close neighbor nodes.

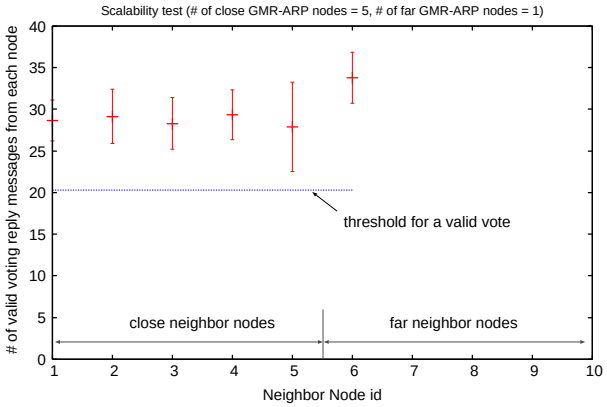
5.3. Analysis of Voting Procedure Delay

In this subsection, we compare the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for the four scenarios described in Table 1. The voting procedure delay means the duration of the time interval from the voting request transmission time to the time when the voting decision is made.

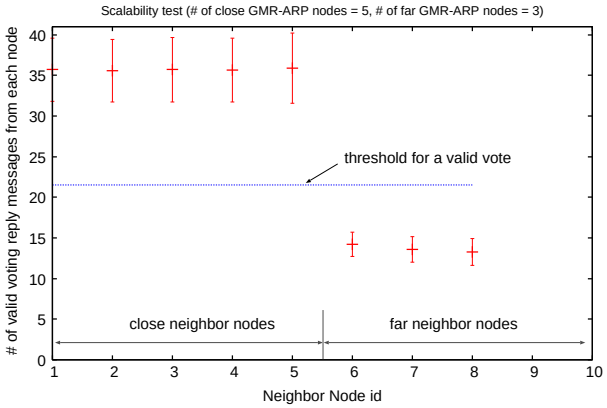
Fig. 14 shows the comparison results for Scenario 1. The voting procedure delay is highest for the EMR-ARP scheme because the iteration number m, which determines the complexity of the puzzle, is determined to maintain the puzzle computation time at the voting request node close to 450 ms. In the experiment, it took approximately 400 ms for the fastest machine to resolve the puzzle. If the response time of the earliest responding node is t_1, the voting request node waits only up to $2t_1$ from the voting request transmission time, and makes a decision based on the replies arrived in time. Therefore, it takes approximately 800 ms or more until the voting request node makes a decision in the case of the EMR-ARP scheme. The voting procedure delay increases slightly as the number of attackers increases, because the voting request node needs to spend more time to check the validity of more puzzle answers. On the other hand, GMR-ARP and MR-ARP spends much less time in the voting procedure, because the computational puzzle is not used in those schemes.

The voting procedure delays were measured to be less than 4 ms for both GMR-ARP and MR-ARP. The voting procedure delay was smallest for MR-ARP. In the case of MR-ARP, the voting request node makes a decision only after collecting 120 voting reply messages. However, the EMR-ARP voting request node waits until the number of the voting reply messages reaches 50 for at least one node. In Scenario 1, there are 6 good nodes connected to the subnet through a wire. If those nodes send the reply messages at a similar rate, then the voting request node might need to wait until it receives approximately 300 reply messages from the neighbor nodes. Therefore, GMR-ARP spends slightly more time than MR-ARP in Scenario 1.

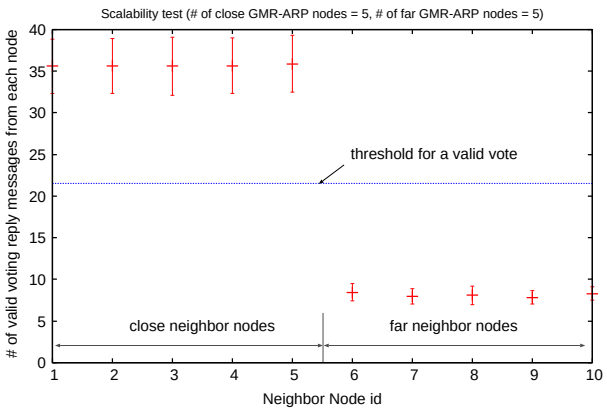
Fig. 15 shows the comparison results for Scenario 2. If we compare Fig. 15 with Fig. 14, then we find that



(a) # of close neighbor nodes is 5, and # of far neighbor nodes is 1



(b) # of close neighbor nodes is 5, and # of far neighbor nodes is 3



(c) # of close neighbor nodes is 5, and # of far neighbor nodes is 5

Figure 13: Comparison of the number of valid voting reply messages from the close neighbor nodes and that from the far neighbor nodes under the GMR-ARP scheme

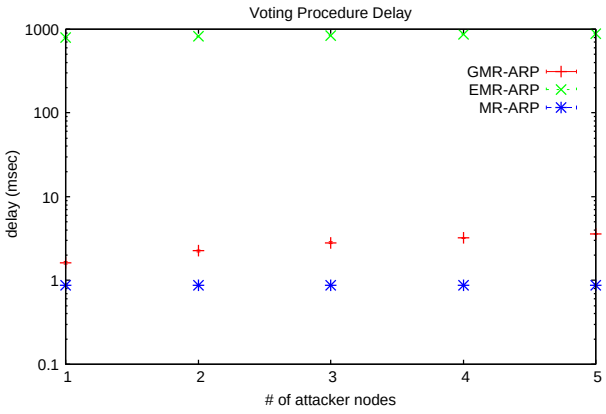


Figure 14: Comparison of the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 1

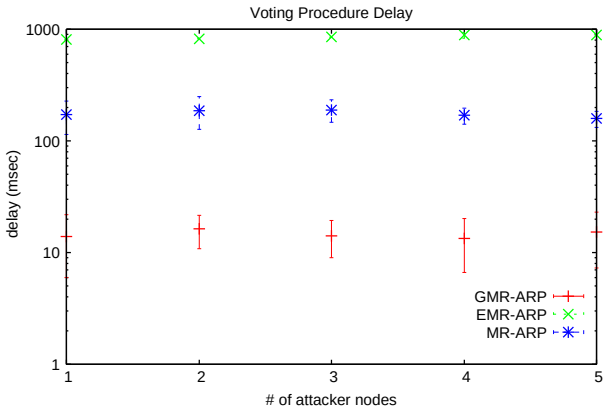


Figure 15: Comparison of the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 2

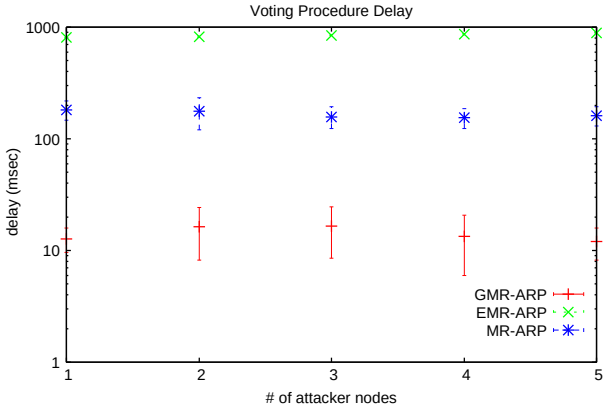


Figure 16: Comparison of the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 3

the voting procedure delay has increased significantly for MR-ARP, and the delay increased slightly for GMR-ARP. On the other hand, the voting delay for EMR-ARP did not change significantly, since the delay was determined dominantly by the puzzle computation time at the neighbor node with a high performance CPU. The increase in delay for MR-ARP can be explained as follows. In Scenario 1, the voting request node was a wired node, and made a decision after collecting 120 reply messages from the wired neighbor nodes. On the other hand, in Scenario 2, all the nodes were wireless nodes with the exception of the gateway router. Therefore, it took much longer time for the voting request node to collect 120 reply messages due to the MAC layer access delay on the wireless link [21].

In the case of GMR-ARP, the voting request node did not need to collect 120 reply messages. In Scenario 2, there was only one wired node, the gateway router, and the voting replies from the gateway router arrived much earlier than the voting replies from the other wireless nodes for the reason given in the previous subsection. Because the voting request node can make a decision based only on the 50 replies from the gateway router, the voting procedure delay is small for GMR-ARP compared to the other voting-based schemes.

Fig. 16 shows the comparison results for Scenario 3. The trend was similar to the case of Scenario 2 in Fig. 15 for all the schemes because the environment was similar, i.e. the voting request node and all the neighbor nodes are wireless nodes except for the gateway router.

Fig. 17 compares the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 4. We find that the trend is very similar to the case of Scenario 1, i.e. Fig. 14. All the nodes are wired nodes in both scenarios, and the numbers of the neighbor nodes participating in voting are close to each other. Thus, the voting procedure delays are measured similar in those scenarios for each of the three voting-based schemes.

6. Conclusions

This paper proposed, a new version of Address Resolution Protocol (ARP), GMR-ARP, to prevent ARP cache poisoning-based MITM attacks. GMR-ARP attempts to protect the ARP cache of the upgraded machines based on the collaboration among them. The conflict on the mapping between IP and MAC addresses was resolved using a voting method. There are some other voting-based schemes, i.e. EMR-ARP and MR-ARP. However, the voting scheme was refined further to overcome the limitation of the other voting-based schemes. Especially, the number of voting reply mes-

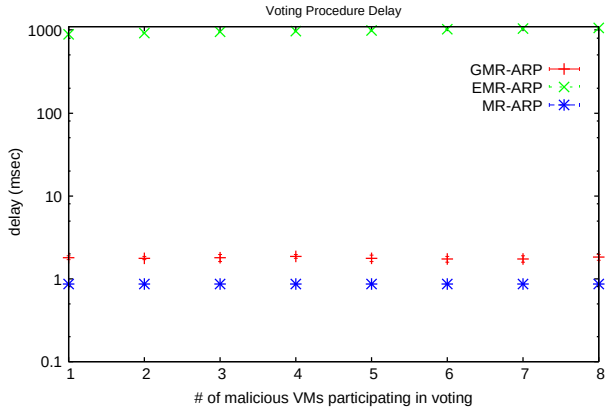


Figure 17: Comparison of the voting procedure delays of GMR-ARP, EMR-ARP, and MR-ARP for Scenario 4

sages required for each neighbor node was determined analytically considering the fairness among the different nodes.

EMR-ARP might not work reliably when the computation power of the neighbor machines is significantly different from each other. On the other hand, GMR-ARP circumvents this limitation by avoiding the use of a computational puzzle. In contrast to MR-ARP, GMR-ARP can protect upgraded machines when the wired nodes and wireless nodes coexist in the same subnet. GMR-ARP improves the fairness in voting compared to MR-ARP by partially dropping the voting reply messages from the too-early replying nodes. The analysis and experimental results show that GMR-ARP can protect the ARP cache of the upgraded machines from the ARP cache poisoning attack effectively with less traffic overhead compared to EMR-ARP or MR-ARP provided the number of the good machines is larger than the number of the malicious machines in the wired network. Therefore, if this condition is satisfied, GMR-ARP can protect the ARP cache of the upgraded machines, even though the attackers outnumber the good nodes in the wireless network.

The proposed mechanism is not based on public-key cryptography, and the manual configuration, such as distribution of the public key and the MAC address of the centralized key management server, is not required. Therefore, the proposed scheme can efficiently mitigate ARP poisoning-based MITM attacks, even in public Wi-Fi hot-spots.

Appendix A. Derivation of (3)

To derive (3), we first investigate the probability $\Pr(K_1 = t|K_Z = l)$. Assuming that the event of $Z = i$ is independent of the event $K_z = l$, we can obtain from (2)

$$\Pr(Z = i|K_Z = l) = \Pr(Z = i) = 1/M.$$

The following can be obtained from the above relation:

$$\begin{aligned}\Pr(K_1 = t|K_Z = l) &= \sum_{i=1}^{M} \Pr(K_1 = t, Z = i|K_Z = l) \\ &= \sum_{i=1}^{M} \Pr(K_1 = t|Z = i, K_Z = l) \Pr(Z = i|K_Z = l) \\ &= \frac{1}{M} \sum_{i=1}^{M} \Pr(K_1 = t|Z = i, K_Z = l).\end{aligned} \tag{A.1}$$

We first consider the case where $t = l$. In this case, $\Pr(K_1 = t|Z = i, K_Z = l) = 0$, for $i \neq 1$, and $\Pr(K_1 = t|Z = 1, K_Z = l) = 1$. Therefore, the following can be obtained from (A.1)

$$\Pr(K_1 = t|K_Z = l) = 1/M, \quad \text{for } t = l. \tag{A.2}$$

We now consider the case where $t < l$. We can easily know that $\Pr(K_1 = t|Z = i, K_Z = l) = 0$ for $i = 1$, and $\Pr(K_1 = t|Z = i, K_Z = l) = \Pr(K_1 = t|Z = j, K_Z = l)$ for $i \neq j (i > 1, j > 1)$. Thus, we can obtain the following relation from (A.1):

$$\begin{aligned}\Pr(K_1 = t|K_Z = l) &= \frac{M-1}{M} \Pr(K_1 = t|Z = M, K_Z = l) \\ &= \frac{M-1}{M} \frac{\Pr(K_1 = t, Z = M, K_Z = l)}{\Pr(Z = M, K_Z = l)}, \\ &\qquad \text{for } t < l.\end{aligned} \tag{A.3}$$

When $K_Z = l$, $\sum_{i=1}^{M} K_i$ can have any value in the range of $[l, (M-1)(l-1)+l]$. Thus, we have

$$\begin{aligned}&\Pr(Z = M, K_Z = l) = \\ &\qquad \sum_{m=l}^{(M-1)(l-1)+l} \Pr(Z = M, K_Z = l, \sum_{i=1}^{M} K_i = m).\end{aligned} \tag{A.4}$$

The probability $\Pr(Z = M, K_Z = l, \sum_{i=1}^{M} K_i = m)$ can be calculated by considering all the possible combinations of voting reply messages from each neighbor node. When we calculate $\Pr(Z = M, K_Z = l, \sum_{i=1}^{M} K_i = m)$, the number of the voting reply messages from Node M is l, and we let x_i denote the number of the voting reply messages from Node i $(1 \leq i \leq M-1)$. Then, x_i should

be less than l because Node M replies the earliest in this case, and we can obtain the following relation:

$$\begin{aligned}\Pr(Z=M, K_Z=l, \sum_{i=1}^{M} K_i = m) = \\ \sum_{\substack{x_1+\ldots+x_{M-1}=m-l,\\ 0\le x_i\le l-1}} \binom{m-1}{l-1}\left(\frac{1}{M}\right)^{l-1}\binom{m-l}{x_1}\left(\frac{1}{M}\right)^{x_1} \\ \times\binom{m-l-x_1}{x_2}\left(\frac{1}{M}\right)^{x_2}\times\cdots \\ \times\binom{m-l-x_1-\cdots-x_{M-2}}{x_{M-1}}\left(\frac{1}{M}\right)^{x_{M-1}}\cdot\left(\frac{1}{M}\right) \\ = \sum_{\substack{x_1+\ldots+x_{M-1}=m-l,\\ 0\le x_i\le l-1}} \left(\frac{1}{M}\right)^{m}\binom{m-1}{l-1}\binom{m-l}{x_1} \\ \times\binom{m-l-x_1}{x_2}\times\cdots\times\binom{m-l-x_1-\cdots-x_{M-2}}{x_{M-1}}.\end{aligned} \tag{A.5}$$

Combining (A.4) and (A.5) yields

$$\begin{aligned}\Pr(Z=M, K_Z=l) = \sum_{m=l}^{(M-1)*(l-1)+l}\left(\frac{1}{M}\right)^{m}\binom{m-1}{l-1}\times \\ \sum_{\substack{x_1+\ldots+x_{M-1}=m-l,\\ 0\le x_i\le l-1}} \prod_{1\le j\le M-1}\binom{m-l-\sum_{1\le k<j}x_k}{x_j}.\end{aligned} \tag{A.6}$$

$\Pr(K_1 = t, Z = M, K_Z = l)$ can be derived in a similar manner as follows:

$$\begin{aligned}\Pr(K_1=t, Z=M, K_Z=l) = \sum_{m=l+t}^{(M-2)*(l-1)+l+t}\left(\frac{1}{M}\right)^{m}\binom{m-1}{l-1} \\ \times\binom{m-l}{t}\sum_{\substack{x_2+\ldots+x_{M-1}\\ =m-l-t,\\ 0\le x_i\le l-1}} \prod_{2\le j\le M-1}\binom{m-l-t-\sum_{2\le k<j}x_k}{x_j}.\end{aligned} \tag{A.7}$$

By (A.2), (1) can be expressed as

$$\begin{aligned}P_{vv}(l,\zeta) &= Pr(K_1\ge\zeta l|K_Z=l) \\ &= \sum_{i=\lceil\zeta l\rceil}^{l-1}\Pr(K_1=i|K_Z=l)+\frac{1}{M}.\end{aligned} \tag{A.8}$$

Combining (A.3), (A.6), (A.7), and (A.8) yields (3). □

Appendix B. Derivation of (5)

Z was defined to be a random variable denoting the node that sends the required number (l) of voting reply messages the earliest among the neighbor nodes in Section 3. If we put $D(l) = \min_{1\le i\le M} D_i(l)$, the probability $\Pr(D_1(t) \le D(l))$ for $t \le l$ can be expressed as

$$\begin{aligned}\Pr(D_1(t)\le D(l)) &= \Pr(D_1(t)\le D(l), Z=1) \\ &\quad+\Pr(D_1(t)\le D(l), Z\ne 1) \\ &= \Pr(Z=1)\Pr(D_1(t)\le D(l)|Z=1) \\ &\quad+\Pr(Z\ne 1)\Pr(D_1(t)\le D(l)|Z\ne 1).\end{aligned} \tag{B.1}$$

$\Pr(Z = 1) = 1/M$ and $\Pr(Z \ne 1) = (M-1)/M$ by (2). If $Z = 1$, then $D_1(l) = \min_{1\le i\le M} D_i(l) = D(l)$. $D_1(t) \le D_1(l)$ for $t \le l$. Therefore, we can find $\Pr(D_1(t) \le D(l)|Z = 1) = 1$. Using these relations, (B.1) can be changed to

$$\Pr(D_1(t)\le D(l)) = \frac{1}{M}+\frac{M-1}{M}\Pr(D_1(t)\le D(l)|Z\ne 1). \tag{B.2}$$

$\Pr(D_1(t) \le D(l)|Z \ne 1)$ can be expressed as

$$\begin{aligned}&\Pr(D_1(t)\le D(l)|Z\ne 1) \\ &\qquad=\sum_{j=l}^{\infty}\Pr(D_1(t)\le D(l), D(l)=j|Z\ne 1).\end{aligned}$$

$Z \ne 1$ if and only if $D_1(l) > \min_{2\le i\le M} D_i(l)$. Thus, the above relation can be changed into

$$\begin{aligned}&\Pr(D_1(t)\le D(l)|Z\ne 1) \\ &=\sum_{j=l}^{\infty}\Pr(D_1(t)\le j, \min_{2\le i\le M}D_i(l)=j|D_1(l)>\min_{2\le i\le M}D_i(l)).\end{aligned} \tag{B.3}$$

We can obtain

$$\begin{aligned}&\Pr(D_1(t)\le j, \min_{2\le i\le M}D_i(l)=j|D_1(l)>\min_{2\le i\le M}D_i(l)) \\ &=\Pr(\min_{2\le i\le M}D_i(l)=j|D_1(l)>\min_{2\le i\le M}D_i(l)) \\ &\quad\times\Pr(D_1(t)\le j\,|\min_{2\le i\le M}D_i(l)=j, D_1(l)>\min_{2\le i\le M}D_i(l)).\end{aligned} \tag{B.4}$$

If we put $\hat{D}(l) = \min_{2\le i\le M} D_i(l)$, the second term of the above equation can be expressed as

$$\begin{aligned}&\Pr(D_1(t)\le j\,|\hat{D}(l)=j, D_1(l)>\hat{D}(l)) \\ &\quad=\frac{\Pr(D_1(t)\le j, \hat{D}(l)=j, D_1(l)>\hat{D}(l))}{\Pr(\hat{D}(l)=j, D_1(l)>\hat{D}(l))} \\ &\quad=\frac{\Pr(D_1(t)\le j, D_1(l)>j|\hat{D}(l)=j)}{\Pr(D_1(l)>j|\hat{D}(l)=j)}.\end{aligned}$$

Because we already assumed independence among $D_i(\cdot)$ and $D_j(\cdot)$'s for $i \ne j$, $D_1(l)$ and $\hat{D}(l)$ are also indepen-

dent, and the above relation can be simplified to

$$\begin{aligned}&\Pr(D_1(t) \le j \,|\hat{D}(l) = j, D_1(l) > \hat{D}(l))\\ &= \frac{\Pr(D_1(t) \le j, D_1(l) > j)}{\Pr(D_1(l) > j)} = \Pr(D_1(t) \le j | D_1(l) > j).\end{aligned} \tag{B.5}$$

We can also obtain the following relation for the first term in the final product of (B.4):

$$\begin{aligned}&\Pr(\min_{2\le i\le M} D_i(l) = j | D_1(l) > \min_{2\le i\le M} D_i(l))\\ &= \frac{\Pr(D_1(l) > \hat{D}(l), \hat{D}(l) = j)}{\Pr(D_1(l) > \hat{D}(l))}\\ &= \frac{\Pr(\hat{D}(l) = j)\Pr(D_1(l) > j | \hat{D}(l) = j)}{\Pr(D_1(l) > \hat{D}(l))}\\ &= \frac{\Pr(\hat{D}(l) = j)\Pr(D_1(l) > j)}{\Pr(D_1(l) > \hat{D}(l))},\end{aligned} \tag{B.6}$$

where the last equality is valid by the independence between $D_1(l)$ and $\hat{D}(l)$. By the discussion before (B.3), we have $\Pr(D_1(l) > \hat{D}(l)) = \Pr(Z \ne 1) = (M-1)/M$. Combining this relation with (B.3), (B.4), (B.5), and (B.6) yields

$$\begin{aligned}&\Pr(D_1(t) \le D(l) | Z \ne 1)\\ &= \frac{M}{M-1}\sum_{i=l}^{\infty} \Pr(\min_{2\le j\le M} D_j(l) = i)\Pr(D_1(l) > i\,, D_1(t) \le i\,).\end{aligned} \tag{B.7}$$

We now investigate $\Pr(\min_{2\le j\le M} D_j(l) = i) = \Pr(\hat{D}(l) = i)$. Because we assumed that $D_j(l)$'s are mutually independent among the different j's, we can obtain

$$\begin{aligned}\Pr(\min_{2\le j\le M} D_j(l) \le x) &= 1 - Pr(\min_{2\le j\le M} D_j(l) > x)\\ &= 1 - \Pr(D_2(l) > x, \cdots, D_M(l) > x)\\ &= 1 - \Pr(D_2(l) > x) \times \cdots \times \Pr(D_M(l) > x).\end{aligned} \tag{B.8}$$

The distribution of $D_j(l)$, for $j = 1, 2, \ldots, M$, can be expressed as

$$\Pr(D_j(l) = i) = \binom{i-1}{l-1}\left(\frac{1}{M}\right)^{l-1}\left(1 - \frac{1}{M}\right)^{i-l}\left(\frac{1}{M}\right), \; i \ge l. \tag{B.9}$$

Therefore, $\Pr(D_j(l) > x)$, for an integer x, can be expressed as

$$\begin{aligned}\Pr(D_j(l) > x) &= 1 - \Pr(D_j(l) \le x)\\ &= 1 - \sum_{i=l}^{x}\binom{i-1}{l-1}\left(\frac{1}{M}\right)^{l}\left(1 - \frac{1}{M}\right)^{i-l}, \; 1 \le j \le M.\end{aligned} \tag{B.10}$$

Combining (B.8) and (B.10) yields

$$\Pr(\min_{2\le k\le M} D_k(l) \le x) = 1 - \left\{1 - \sum_{j=l}^{x}\binom{j-1}{l-1}\left(\frac{1}{M}\right)^{l}\left(1 - \frac{1}{M}\right)^{j-l}\right\}^{M-1}.$$

Thus, we can obtain from the above relation

$$\begin{aligned}\Pr(\min_{2\le j\le M} D_j(l) = i) &= \Pr(\min_{2\le j\le M} D_j(l) \le i)\\ &\quad - \Pr(\min_{2\le j\le M} D_j(l) \le i-1)\\ &= \left\{1 - \sum_{j=l}^{i-1}\binom{j-1}{l-1}\left(\frac{1}{M}\right)^{l}\left(1 - \frac{1}{M}\right)^{j-l}\right\}^{M-1}\\ &\quad - \left\{1 - \sum_{j=l}^{i}\binom{j-1}{l-1}\left(\frac{1}{M}\right)^{l}\left(1 - \frac{1}{M}\right)^{j-l}\right\}^{M-1}.\end{aligned} \tag{B.11}$$

We next investigate $\Pr(D_1(l) > i\,, D_1(t) \le i)$ of (B.7) in more detail. If we put $\tilde{D}_{l-t} = D_1(l) - D_1(t)$, then $\tilde{D}_{l-t}$ means the delay of Node 1 to deliver $l-t$ voting reply packets, i.e. from $(t+1)$-th to l-th reply messages, successfully. Since we model the medium access attempt, especially for the medium to the voting request node, of each neighbor node as a Bernoulli trial with the success probability of $1/M$, we can easily know that $\tilde{D}_{l-t} = D_1(l) - D_1(t)$ is independent from $D_1(t)$, and $\tilde{D}_{l-t}$ has the same distribution as $D_1(l-t)$. Thus, we can obtain

$$\begin{aligned}&\Pr(D_1(t) \le i, D_1(l) > i)\\ &= \sum_{k=t}^{i}\Pr(D_1(t) = k)\Pr(\tilde{D}_{l-t} > i - k).\end{aligned} \tag{B.12}$$

We obtain from (B.9)

$$\Pr(D_1(t) = k) = \binom{k-1}{t-1}\left(\frac{1}{M}\right)^{t}\left(1 - \frac{1}{M}\right)^{k-t}. \tag{B.13}$$

$\Pr(\tilde{D}_{l-t} > i-k)$ can also be obtained from the distribution of $D_1(l-t)$ as

$$\begin{aligned}&\Pr(\tilde{D}_{l-t} > i-k)\\ &= 1 - \sum_{j=l-t}^{i-k}\binom{j-1}{l-t-1}\left(\frac{1}{M}\right)^{l-t}\left(1 - \frac{1}{M}\right)^{j-(l-t)}.\end{aligned} \tag{B.14}$$

Combining (B.12), (B.13), and (B.14) yields

$$\Pr(D_1(t) \le i, D_1(l) > i) = \sum_{k=t}^{i} \binom{k-1}{t-1} \left(\frac{1}{M}\right)^{t} \left(1-\frac{1}{M}\right)^{k-t} \times \left\{1 - \sum_{j=l-t}^{i-k} \binom{j-1}{l-t-1} \left(\frac{1}{M}\right)^{l-t} \left(1-\frac{1}{M}\right)^{j-(l-t)}\right\}. \quad \text{(B.15)}$$

Combining (4), (B.2), (B.7), (B.11), and (B.15) gives (5). □

References

[1] D. C. Plummer, An ethernet address resolution protocol, RFC 826, 1982.

[2] R. W. Stevens, TCP/IP Illustrated, vol. 1, Addison Wesley, 2001.

[3] C. Benvenuti, Understanding linux network internals, O'Reilly, 2006.

[4] Hacking UNIX 2003, A tutorial for performing various attacks including ARP poisoning attack on UNIX systems, http://duho.cjb.net

[5] S. Whalen, An introduction to arp spoofing, http://packetstormsecurity.nl/papers/protocols/intro_to_arp_spoofing.pdf

[6] B. Zdrnja, "Malicious Javascript Insertion through ARP Poisoning Attacks," IEEE Security & Privacy, vol. 7, no. 3, pp. 72-74, May-June 2009.

[7] Y. Bhaiji, Network Security Technologies and Solutions, Cisco Press, 2008.

[8] I. Teterin, Antidote, http://online.securityfocus.com/archive/1/299929

[9] D. Bruschi, A. Ornaghi, and E. Rosti, S-ARP: a Secure Address Resolution Protocol, *in Proc. of Annual Computer Security Applications Conference (ACSAC)*, 2003.

[10] V. Goyal, and R. Tripathy, An efficient solution to the ARP cache poisoning problem, *in Proc. of Information Security and Privacy*, July 2005.

[11] W. Lootah, W. Enck, and P. McDaniel, TARP: Ticket-based address resolution protocol, *Computer Networks*, vol. 51, no. 15, pp. 4322-4337, Oct. 2007.

[12] R. Philip, Securing Wireless Networks from ARP Cache Poisoning, *Master's Thesis*, San Jose State University, 2007.

[13] S. Y. Nam, D. Kim, and J. Kim, Enhanced ARP: Preventing ARP Poisoning-based Man-in-the-Middle Attacks, *IEEE Communications Letters*, vol. 14, no. 2, pp. 187-189, Feb. 2010.

[14] M. Lacage, M. H. Manshaei, and T. Turletti, IEEE 802.11 Rate Adaptation: A Practical Approach, *in Proc. of ACM International Symposium on Modeling, analysis and simulation of wireless and mobile systems (MSWiM'04)*, Oct. 2004.

[15] S. Y. Nam, S. Jurayev, S. Kim, K. Choi, and G. S. Choi, "Mitigating ARP Poisoning-based Man-in-the-Middle Attacks in Wired or Wireless LAN," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012:89, March 2012.

[16] C. Dwork, and M. Naor, Pricing via processing or combatting junk mail, *in Proc. of CRYPTO*, 1992.

[17] N. Borisov, Computational Puzzles as Sybil defenses, *in Proc. of IEEE International Conference on Peer-to-Peer Computing*, 2006.

[18] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y-C. Hu, Portcullis: protecting connection setup from denial-of-capability attacks, *in Proc. of SIGCOMM*, Kyoto, Japan, Aug. 2007.

[19] Gratuitous ARP - The Wireshark Wiki, http://wiki.wireshark.org/Gratuitous_ARP

[20] T. Sakurai, H. L. Vu, "MAC Access Delay of IEEE 802.11 DCF," *IEEE Trans. on Wireless Communications*, vol. 6, no. 5, pp. 1702-1710, May 2007.

[21] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas, "IEEE 802.11 Packet Delay - A Finite Retry Limit Analysis," *in Prof. of IEEE Globecom*, 2003.

[22] C. M. Koksal, H. Kassab, and H. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," *in Prof. of ACM SIGMETRICS*, 2000.

[23] Z. Fang, B. Bensaou, and Y. Wang, "Performance evaluation of a fair backoff algorithm for IEEE 802.11 DFWMAC," *in Prof. of ACM MOBIHOC*, 2002.

[24] Z. Li, S. Nandi, and A. K. Gupta, "Modeling the short-term unfairness of IEEE 802.11 in presence of hidden terminals," *Performance Evaluation*, vol. 63, no. 4, pp. 441-462, May 2006.