

# Source Address Validation Support for Network Forensics

Khamphao Sisaat (sisaat-k@is.naist.jp)

Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, Japan

Daisuke Miyamoto (daisu-mi@is.naist.jp)

Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, Japan

## Abstract

This paper presents a new algorithm for supporting network forensics. The key idea of our approach is that it uses a probing technique to verify whether or not a sender node sent a packet with its real source addresses based on a probing technique. Our approach also establishes where the real sender node of the packet is located by identifying the port number of the Layer 2 Switch (L2SW). In addition, our proposed system stores specific information about network users in order to deal with security incidents after they have taken place. Our approach provides extensive evidential information about network users, which always indicates the true location of their source nodes. It helps network operators to identify and trace attacks back to their origin source nodes for most cases of network attack, even if much time has elapsed since the occurrence of the security incident. Consequently, by using our approach, an attacker can no longer hide its node location, whatever the patterns of the attack.

**Key words** Source Address Validation, Network Forensics, Source Address Spoofing, ARP, FDB, Layer 2 Switch

## 1 Introduction

In this paper, we describe a novel algorithm for source address validation to support network forensics.

Network forensics describes after-the-fact steps that a network operator takes to capture, collect, record and analyze network events in order to retain effective evidential information about all network users [12]. Digital forensic science and network forensics have been defined [8], and IETF has published guidelines for evidence collecting and archiving [3] in which many kinds of evidence are identified, such as memory, temporary file systems, physical configuration and logged data.

Increasing levels of unauthorized access generate an increasing demand for forensic analysis, namely, electronic evidence discovery, computer analysis and data recovery. By performing forensic analysis, network operators can analyze security incidents such as unauthorized access and trace the incident back to its original source. When a security incident is detected, a network operator investigates the cause by gathering information, i.e., system and application logs, captured packets, volatile data and configuration files. These data con-

tain a great deal of useful information for network operators, including IP addresses, link-layer (MAC) addresses, network interfaces, logged times and details of events. Many existing forensic tools therefore try to capture packets on a network in order to identify and analyze a variety of trails. Accordingly, collected electronic evidence is the core of this growing research field.

In the Internet, however, many malicious users apply various types of attack techniques to break into systems, steal information, change data or destroy the systems. Moreover, malicious users often access the network by employing spoofed techniques to hide their source nodes from network operators. In this case, the information in the log files that support network operators in their analysis of network incidents is not always reliable. In other words, source address information about the end users, such as IP address and MAC address, does not always indicate the true source node, confounding the efforts of network operators to trace and specify the attacker's node based on information in the log files.

This research focuses on Source Address Validation (SAV) to make evidential information about network users, such as source IP address, source MAC address and switch port number, more reliable for network operators to analyze and handle when responding to security incidents. SAV is very important for network operators wishing to identify an attacker's nodes after an attack has been completed. Unfortunately, the source address of a packet is not always trustworthy, because the attacker can forge a packet and send it to the target or victim's node. The reason that networks are vulnerable to spoofed source addresses is that the Internet protocol lacks security; IP allows attackers to employ source IP address spoofing [19], [21] and [14].

Spoofing the source IP address of packets on the Internet is one of the major techniques used by attackers to mount Denial of Service (DoS) attacks. In DoS, an attacker forges the source IP address of the packets that are used in the attack. Instead of carrying the source IP of the node it came from, the packet contains an arbitrary IP address, which is selected either randomly or intentionally. According to one study [7], there are at least four thousands such attacks every week on the Internet. Aside from being very effective in generating DoS for the victim, spoofed attacks give attackers two additional advantages: first, they weaken the ability to mitigate the attack, since malicious traffic cannot be categorized by source and hence is much harder to filter out. Second, they make it

difficult to trace the attack back to the source of attack.

Although there are some countermeasures to this problem, they do not provide effective mechanisms that network operators may use today to detect and filter out spoofed packets. Existing countermeasures are reviewed briefly in Section 2, but they cannot eliminate completely the problem of source address spoofing.

In this paper, we present a new algorithm for validating the source address of each IP packet, based on a confirmation technique. The essential idea of our approach is to verify whether or not the source node of a packet sent the packet with its own source addresses. This is done with a confirmation packet, which is based on an ARP (Address Resolution Protocol) [5] request about the source IP address of the packet to the node whose MAC address is the same as the source MAC address of the packet. The system also determines the consistency between the result of ARP probing and the source address pair of the issued packet: any inconsistency shows that spoofing has occurred. Moreover, the system ensures the traceability of all end nodes by recording in a log file specific information about the source node, such as source IP address, source MAC address, and switch port number including logged time. By keeping evidential information in such a manner, network operators can analyze and trace security incidents for any cases of attack, even after the security incidents have been completed.

The rest of this paper is organized as follows. Section 2 discusses existing approaches to the problem. The technical details of our approach are described in Section 3, followed by a discussion in Section 4. Section 5 summarizes our contributions and concludes the paper.

## 2 Related Work

### 2.1 Authentication method

A typical method that can eliminate malicious traffic in computer networks is authentication. IEEE802.1X [24] provides a framework for allowing a user to be authenticated by a certificate authority before the user accesses the Ethernet or a Wireless LAN. IEEE802.1X uses three components for authentication conversation, i.e., the user that wants to be authenticated (called the supplicant), Ethernet switch or wireless access point (the authenticator), and a RADIUS server called (the authentication server). IEEE802.1X itself does not provide an actual authentication mechanism, but uses the Extensible Authentication Protocol (EAP) [13] for message exchange during the authentication process.

The mechanism of IEEE802.1X has two disadvantages: first, it is burdensome for network operators to manage user accounts and certificates and to undertake a registration process for each new user before they can use the network; second, it is applicable only to closed networks, as unspecified users cannot access the network freely.

S-ARP (Secure Address Resolution Protocol) [4] provides protection against ARP poisoning [20] based on an extension of the ARP protocol with an authentication scheme for ARP replies. ARP poisoning attacks employ source address spoofing and perform Man-in-the-Middle (MitM) attacks [1]. In S-ARP, each host has a public/private key pair certified by a

local trusted party on the LAN, which acts as a certification authority. All ARP reply messages are digitally signed by the sender with the corresponding private key. At the receiving side, the signature is verified by the node public key. Thus an attacker can no longer send spoofed ARP replies to redirect traffic through its adapter.

An attacker, however, could still announce a false MAC address for its adapter, whether such an address be some other node's or a non-existent one. Another disadvantage of S-ARP is that, using the S-ARP in a mixed LAN, the part running traditional ARP is still vulnerable to ARP poisoning. In addition, the list of nodes that does not run S-ARP must be given to every secure node, which has to communicate with an unsecured node because nodes that run S-ARP will not accept a non-authenticated message unless specified in the list of knowing nodes. Furthermore, S-ARP, like IEEE802.1X, incurs high management costs for registration processes of new network users as well as their certificate management. Therefore, it can be used only in a closed network.

### 2.2 Filtering method

To detect and filter out spoofed packets, the most popular method is ingress filtering [17], in which an ISP prohibits receiving packets from its local network whose source address does not belong to the corresponding local network address space. Thus, the mechanism of ingress filtering ensures that traffic leaving a local network may carry only spoofed IP addresses that belong to the same subnet, a process that is called subnet spoofing [11].

Ingress filtering is good-will preventive, but not a self-defensive method. Cooperative network operators deploy the method to avoid being the source of such attacks, but the method cannot provide any remedies to victims who are being attacked. If all networks employed ingress filtering, it could significantly reduce packet spoofing on the Internet. However, the deployment of ingress filtering inflicts significant cost on the implementing ISP, both in equipment and administration, without assuring significant benefit or protection. The incentive for an ISP to deploy these mechanisms is thus relatively low.

### 2.3 Traceback and forensics methods

Traffic pattern: If traffic pattern data is stored during a Distributed Denial of Service (DDoS) attack, it can be analyzed post-attack to look for specific characteristics within the attacking traffic. This characteristic data can be used to update load balancing and throttling [6] countermeasures to increase their efficiency and protection ability. DDoS attack traffic patterns can also help network operators develop new filtering techniques to prevent DDoS attack traffic from entering or leaving their network. This method, however, cannot find the source of the attacker's node if DDoS attacks employ source address spoofing.

Packet traceback: To help identify attackers, packet traceback techniques have been proposed [23]. The concept involves tracing Internet traffic back to its source, rather than that of a spoofed source IP address. This technique helps to identify the attacker. Additionally, when the attacker sends different types

of attacking traffic, this method provides the victim system with information that might help develop filters to block the attack. In a large network that consists of many Autonomous Systems (ASs), a malicious packet sent to the victim's node will move through many ASs before reaching the victim's node. In this case, if one of the ASs located between the attacker's AS and the victim's AS does not support packet traceback, the method cannot trace the malicious packet to its origin source node.

**Network traffic tracking system:** A model for developing a network traffic tracking system that would identify and trace user traffic throughout a network has been proposed [22]. This model can be very successful within a closed network environment where internal client systems can be fully managed by a central network operator who can track individual end user actions, but it begins to break down over widely distributed networks.

**Event logs:** Network operators can also keep event logs of the attack information, such as a DoS attack, in order to perform forensic analysis and to assist law enforcement in the event that the attacker does severe damage. Using Honeypots [15] and other network equipment such as firewalls, packet sniffers, and server logs, the provider can store all the events that occurred during the setup or execution of the attack. This allows network operators to discover what type of DoS attack or combination of attacks was used. If DoS employs source address spoofing, however, network operators are still faced with the trace problem, because the information in the log file contains the spoofed address and thus cannot indicate the true source node of the attacker.

## 3 Proposal

In this paper, we present a new algorithm for validating the source address of each IP packet. The key idea of our approach is that it involves two verifications: first, it verifies the source address pair of the packet, whether it is spoofed or not. Second, it checks to see where the source node of the packet is located. The proposed system can therefore ensure that the source address of each packet always indicates the true sender node.

### 3.1 Assumptions

We assume the following behaviors of the network environment and network operators:

- Any end node can send a packet with or without a spoofed source address.
- An end node may often change an IP address but must not change a MAC address.
- Network operators need to know when a security incident occurred as well as trace the attack back to its source.
- An ARP probing packet, which is sent by the system, may not reach the source node.
- Forwarding Data Base (FDB) information is always trustworthy for network operators to locate the true sender node.

- A network switch provides a port mirroring for the end node to copy each incoming and outgoing packet from one port.
- If all end nodes are directly connecting to the L2SW, the proposed system would effectively specify the location of the sender node.

The first assumption reveals that anyone can use source address spoofing to hide the identity of the attacker node. The reason that source address spoofing is possible is that TCP/IP lacks security features, allowing the attacker to employ source IP address spoofing [14].

The next assumption is simply a characteristic of networks which employ a DHCP (Dynamic Host Configuration Protocol) server [18] to dynamically assign IP addresses to the end nodes. In such a network, an end node may be assigned a new IP address when it connects to the network.

The third assumption addresses the goal of network forensics, as a response to security incidents after they have been completed.

The fourth assumption reveals that if a source node is involved in a DoS attack, the source node for confirmation in a link local network is attacked by a large number of meaningless packets. Thus, the source node cannot receive the ARP probing packet from the proposed system.

The last three assumptions concern the capabilities and features of the L2SW. First, FDB information can always be relied on by network operators, since FDB always learns about the switch port number, where the source node is connected whenever the end node is connected to the switch network. Second, most modern switches support a port-monitoring feature, which allows the end node to copy all network traffic from one port.

### 3.2 Requirements

According to the above assumptions, the followings are required:

- The proposed system must effectively detect the location of the source node for any patterns of attack.
- The proposed system must effectively detect any cases of source address spoofing.
- The proposed system must retain the ability to trace the attack back to its originator.
- The proposed system must send a confirmation packet to only the sender node when the source address pair and switch port number of the packet are not verified.
- The proposed system should not send a confirmation packet to any node that has no relation to the issued packet.
- The system must record the network event with logged time.
- The system can receive all Ethernet flames.

We consider that the first three requirements are the most important part of this research, because we believe that if we can satisfy these requirements, our approach can effectively support network forensics. The contributions from these three requirements will be made network forensics store effective evidential information about the network users. In other words, information about network users can always identify the true source nodes. Therefore, attackers can no longer hide their source nodes for any patterns of attack.

The next two requirements ensure that the confirmation technique of our approach does not disturb other nodes that have no relation to the issued packet. For example, it does not consume network bandwidth while sending a confirmation packet, since the confirmation packet is sent only to the source node of the issued packet.

The next requirement is important for network operators because they may want to know when the security incident has occurred.

The last requirement is needed because the proposed system must capture all packets and check whether or not they are spoofed source addresses before logging them into the log file.

### 3.3 Algorithm

The main algorithm of this research is a probing algorithm to verify the source address of the packet. This algorithm process is indicated in Figure 1, which shows that the probing packet is sent only to the sender node. Therefore, this research confirms that the probing packet does not consume much network traffic and that it has no effect on other nodes that have no relation to the issued packets.

In computer networks there are many kinds of messages that can be used as a probing message to find some network device. These messages are ICMP (Internet Control Message Protocol) echo requests [10] and ARP requests. The ICMP echo request, known as the ping message, does not guarantee that the receiving node of the ICMP request packet will send a reply packet back to the sender node, even if it is online, because an ICMP echo request can be blocked by a personal firewall system. Therefore, if the receiving node is configured to block ICMP packets, it will never send an ICMP reply packet to an ICMP request. On the other hand, an ARP request packet is required for an Ethernet network to function properly, so it is not typically blocked by any security system; any node in the network can thus reply to the ARP request packet. Consequently, we decided to use the ARP request packet as a probing packet in this research.

With broadcast ARP request, the probing packet may consume network bandwidth, since the ARP request packet will be sent to each node in the same subnet. In addition, the probing packet may cause occurrences of spoofing packets, because some nodes may fake an ARP reply packet to the ARP probing packet. To avoid such problems, we send a unicast ARP request as a probing packet only to the node that sent the issued packet. The probing packet is containing IP/MAC address pair of the proposed system as the source addresses, and it is also containing IP/MAC address pair of the sender node as the destination addresses. Thus, the other nodes in the same subnet that have no relation with the issued packet will never receive ARP probing packets from our proposed system.

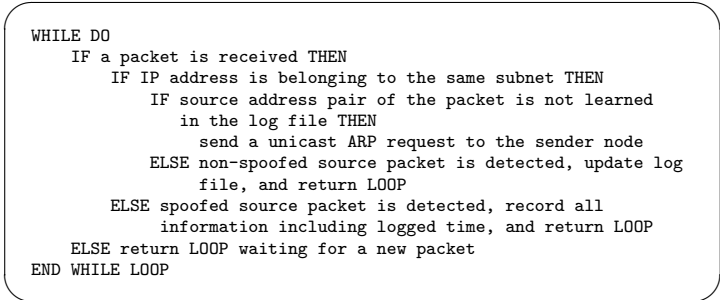


Figure 1: ARP probing algorithm

Table 1: Information record

Item	Explanation	Memory (byte)
Time	logged time	8
IP address	Sender's IP address	4
MAC address	Sender's MAC address	6
Port number	Switch port number	2
Status	Spoofed or non-spoofed	2

In today's Internet, any nodes in the network can forge packets; therefore, by using an ARP probing message to verify the source address of the packet, it is still possible to have spoofed source packets that have been sent by an attacker. Consequently, to enhance the reliability of source IP address and MAC address of the packet, we considered using the FDB of the L2SW, since the FDB always learns about the switch port number where the sender node is connected. Consequently, based on the information about the FDB, together with the source address pair result of ARP probing, network operators can easily find the location of the sender node in many cases of an attack.

Sending an ARP probing packet introduces some delays to normal packets. To avoid such delays, we employ the log file to store specific information about each end node after it has sent the packet. This information contains logged time, source IP address, source MAC address, switch port number and result of packet checking, as indicated in Table 1. If the received packet has the pair of source IP address and MAC address corresponding to the switch port number matches with the pair in the log file, then the system will not send an ARP probing packet again. An ARP probing packet will be sent when the system receives a packet from an end node that has never sent a packet before, or when an entry in the log file has expired. Using log files therefore reduces the delays that might be incurred by normal packets in the system. In addition, the evidential information about network users in the log file is very useful for network operators to analyze and trace users after security incidents have occurred, because it always indicates the true source node of the packet.

### 3.4 Source Address Validation

The proposed system maintains a log file recording of the source IP address, MAC address, switch port number, logged time and results of checks of previously received packets. When the system receives a packet, it checks the pair of IP

address and MAC address, and the switch port number, and then compares these to the pairs in the log file. If there is a matching pair, the packet is considered as coming from a node that has been checked before; in this case, the system will update the logged time of the entry. If there is no matching pair, however, the system sends an ARP probe to the sender node to ask for its IP address. If the system does not receive an ARP reply packet from the sender node within 5 ms<sup>1</sup>, it designates the packet as a spoofed packet. The result of checking source IP address, MAC address, and switch port number of the packet (including logged time) are then recorded in the log file. On the other hand, the system does a consistency check by comparing the source IP/MAC address pair of the ARP reply packet with the pair of issued packets. If there is a match, the packet is determined as a non-spoofed packet, but if there is no match, the packet is deemed a spoofed packet and the results of checking and information about the packet, including logged time, are recorded in the log file.

### 3.5 System Design

Our approach can be implemented as the following network models:

#### Passive monitoring device

To implement with this model, the proposed system can be incorporated into any end node which connects directly to the switch, as indicated in Figure 2. The system will act as a monitoring device and capture all Ethernet frames and examine them before recording the information about the sender node into the log file. In this model, the system can employ port mirroring [2], a feature of the switch to capture all packets that pass through the switch. The system can also use the Simple Network Management Protocol (SNMP) [9] to fetch all FDB entries from the switch. It is easy to implement our proposed algorithm on this implementation model because our approach concept is simple and there is no need to modify any network protocols such as ARP. However, the passive monitoring device model has two disadvantages. First, it requires a high overhead to fetch information such as Ethernet frames and FDB information from the switch. Second, since the system employs SNMP to fetch the FDB information from the switch, FDB in the system and FDB in the switch may not synchronize if the system is not fast enough to get a new entries update from the switch. This will affect the detection of source node location in our proposed system, for example if an end node was moved to a new switch port number while the log file of the proposed system still kept the previous switch port number corresponding to the IP/MAC address pair of the end node. In this case, when the end node sends a packet, the proposed system will use the previous information about the end node to verify the packet. As a result, the evidential information in the log file cannot be relied on by network operators wishing to find the location of the source node of the packet.

#### Build-in L2SW

<sup>1</sup>According to the result of checking the ARP request/reply packet's Round Trip Time (RTT) in the LAN, the approximate RTT is 1.2 ms to 4.8 ms. Therefore, we designated 5 ms as the waiting time for an ARP reply packet from the sender node in our proposed system.

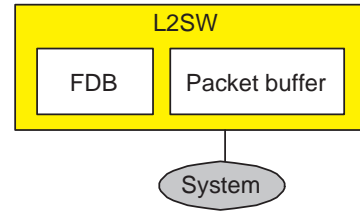


Figure 2: Passive monitoring device

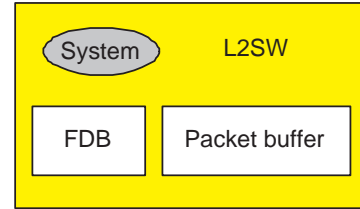


Figure 3: Build-in L2SW

To implement with this model, our proposed system is incorporated into the inside of the L2SW as illustrated in Figure 3. In this model, our proposed system can get a new entries update of FDB from the switch faster than the previous model, thereby making the FDB information in the system more effective in locating the sender node. This model also requires a low overhead to fetch all information from the switch, since the only information needed, Ethernet frames and FDB, must be held in the switch. This network model, however, is difficult to implement if the proposed system algorithm differs substantially from the traditional L2SW.

### 3.6 System Verification

We have verified our proposed algorithm in a passive monitoring device with the network testbed configuration illustrated in Figure 4. Node A is operated by an attacker, nodes X and Y are operated by normal users and node M is operated by our proposed system. The end nodes used for the testing environment run FreeBSD 4.11 and every end node belongs to the same local network, with a L2SW (FOUNDRY NETWORKS FastIron WorkGroup) among them.

Our proposed system is tested to determine whether or not it can detect the location of the sender node in an actual attack. The test cases and results are illustrated in Table 2, which shows that our proposed system effectively detects the location of the sender node for all patterns of source address spoofing.

Based on the verification results, this research reveals that the source IP/MAC address pair corresponding to switch port number of each packet is always reliable for network operators wishing to investigate or to locate the source node of the attacker. Therefore, if we record this information in the log file, it will be available and reliable for network operators to analyze and trace security incidents back to their origin source nodes, even if the incidents have been completed. Consequently, this research should be a valuable tool for network forensics to cope with malicious events after they have been completed.

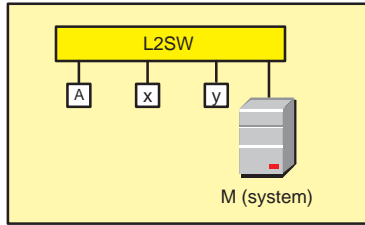


Figure 4: Testbed environment

Table 2: Test results of node location verification

MAC \ IP		Non-Spoofed	Spoofed	
			(a)	(b)
Non-Spoofed		Detectable	Detectable	Detectable
Spoofed	(x)	Detectable	Detectable	Detectable
	(y)	Detectable	Detectable	Detectable

**Note:**

- (a) spoofed IP address is not used by another node in the same LAN.
- (b) spoofed IP address is used by another node in the same LAN.
- (x) spoofed MAC address is not used by another node in the same local link.
- (y) spoofed MAC address is used by another node in the same local link.

## 4 Discussion

The main purpose of our research is to build a system that can support network forensics by ensuring that the source address of each IP packet always indicates the true sender node. Our approach is simple and easily configurable, and it does not require ARP modification.

In our verification testbed, we successfully created a new algorithm to detect the location of a sender node in many types of attack. By recording investigative evidential information about each network user, our system can provide support for network operators to deal with security incidents on their networks after they have been completed.

### 4.1 Stand Alone Network Model

In this work, our approach is concerned with security incidents in a link local environment. In this model, our approach therefore acts as a monitoring device and captures and examines all network traffic from each end node on the LAN. For the incident response of this model, our approach can report evidence about the incident, including the time and place at which it occurred, and the location of the sender node. Thus, network operators can easily find the source of the incident. In the stand alone network model, if the system goes down as a result of an attack, i.e., a DoS attack, it cannot do anything with the packets while it is down. To avoid DoS attacks, network operators should employ a device such as IDS. Another problem of the stand alone network model is disappearance of the

log files. For example, if the disk storing log files is broken, the proposed system is useless, since there is no evidential information about the network users to deal with security incidents. Additionally, with the stand alone network model, even if the system can detect spoofed source packets, it cannot determine whether the packet is a malicious or a normal one. To avoid the last two problems, the proposed system can be deployed in collaboration with other security devices, as explained in the following subsection.

### 4.2 Network Component Collaborations

To further increase security of computer networks, the proposed system should collaborate with other network components as illustrated in Figure 5. To avoid disappearance of the log file, the proposed system can store the log file in the syslog. Therefore, even if the disk storage of the system is broken, the information about the network users in the syslog is always available for network operators to cope with any incidents. When the system detects a spoofed source packet, it can ask for IDS to analyze the packet and determine whether it is a malicious or a normal packet. Furthermore, by collaborating with Quarantine Network [16], the system can isolate a malicious node from the network when it has been detected. However, making the module of this network model communicate between the proposed system and other components is complex.

- Operator: the person who manages the system investigates whether an incident has occurred, and finds the source of the attacker’s node based on the information in the log file.
- Syslog: the protocol used for capturing the log file from the proposed system to the syslog server.
- Spoofed packet analyzing: when the proposed system detects a spoofed source packet, it copies the spoofed source packet for analysis by IDS, whether the packet is malicious or not.
- Malicious node isolation: if the network operator finds incidents that are hard to deal with, the proposed system may ask Quarantine Network to isolate the malicious node from the network until the problem has been properly fixed and disinfected.

### 4.3 Method Comparisons

In this section, we compare our approach with IEEE802.1X and S-ARP methods, described in Section 2.1, in terms of management cost and network usage. We believe that our approach has a lower management cost, because it does not require a registration process before each network user connects to the network. Furthermore, our approach can be used in open networks, where any unspecified end nodes can be connected to the network freely without any registration processes or any configurations on the end node or on the network server.

Our approach has certain drawbacks, as the ARP probing packet introduces a delay to each normal packet. However,

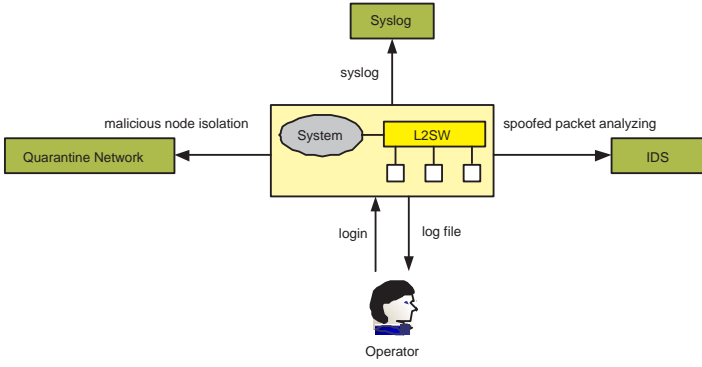


Figure 5: Network component collaborations

this happens only the first time a packet is received; when the same end node sends further packets, the system will not send an ARP probing packet again.

#### 4.4 Performance Analysis

The performance of our approach in many possible cases is analyzed as follows:

##### The bottleneck and worst cases

*The bottleneck case* : The bottleneck part of our approach is ARP probing. Since our approach sends an ARP probe to verify the source address of each new coming packet, it introduces delays to each normal packet of 1.2 ms to 5 ms as the RTT of an ARP request/reply packet. This happens only the first time for each received packet, because the log file helps the system to avoid sending unnecessary ARP probes. When the system receives the first packet and verifies that the source IP/MAC address pair and switch port number of the packet is in the log file, the system will never sent an ARP probe again.

*The worst case* : This is illustrated in the network topology shown in Figure 6. The attacker, i.e., from node A, spoofs the source IP and MAC address of node B, which is connected to the same hub and communicating. In this attack scenario, when the attack is found, it is difficult for network operators to distinguish which of node A or node B is the malicious node, since the hub port cannot tell anything about the end nodes that are connected to it. Our system can support network operators in tracing the attacker's node into its location at the port number of the switch. In this case, a possible way for network operators to deal with such an attack is to separate both nodes from the network by moving them to a new VLAN. Alternatively, the operator may search for the attacker's node by manually isolating end nodes one by one until the real attacker's node is found.

According to the last assumption, as explained in Section 3.1, if all end nodes in the LAN are directly connected to the L2SW, the above problem would never occur. In other words, the proposed system would effective specify the location of the source nodes of the packets for any patterns of attack.

##### Computation Overhead

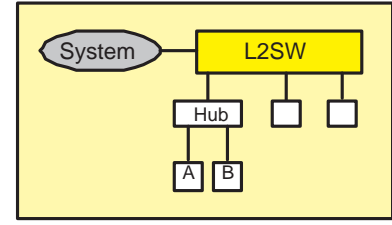


Figure 6: The worst case

*Time checking* : the time checking of the packet can be described by the following equation:

$$\text{Total Time} = O(t(F)) + O(t(M)) + O(t(N))$$

- $O(t(F))$  - time to fetch an Ethernet frame and FDB entries from the switch.
- $O(t(M))$  - time for searching the information in the log file.
- $O(t(N))$  - RTT of ARP request/reply packet.

From this equation we can consider that if we implement our approach as build-in L2SW, the time to fetch the information  $O(t(F))$  and time for searching the information from the log file  $O(t(M))$  must not be high, because all the information is in the switch. RTT of ARP request/reply packet  $O(t(N))$  will be 0 if the received packet is coming from a source node that has been checked by the system before. Thus, this research confirms that our approach takes very little time to check the packet.

*Memory consumption* : Each entry in the log file requires 22 bytes of memory space, as illustrated in Table 1. The table shows that each entry consumes little memory space, and therefore that memory consumption should cause no concern.

To determine the capacity of the log file of the proposed system, we considered whether the network's size is large or small. In a large network, the log file will need to store numerous entries because many client nodes will be communicating on the network. In the current version of the proposed system, we did not precisely estimate the storage capacity of the log file. Here, we give an example of the memory space required to support a log file. As indicated in Table 1, one entry requires 22 bytes; therefore, if the proposed system is incorporated into a device that has a free memory space of 128 kilobytes, the log file can store almost 6,000 entries.

The timeout of each entry can be considered for two cases. First, in a short timeout entry, the entry will be deleted early by the proposed system. This helps to reduce memory space, but has the drawback that the entry may delete too early, before the attack has been detected, thus making the log file useless. Second, in a long timeout entry, the entry will be deleted late, and may still remain even after the attack has been detected. Keeping the entry for a long time will consume memory space. Considering the advantage and disadvantage of both short and long timeout entries, deciding the timeout for each entry depends on the network security policy. Some

network security policies may require a short timeout for each entry, while others may require a long timeout.

### ARP packet consideration

If an ARP reply packet is not returned when an ARP probing packet is sent by the proposed system, we assume that the target node may be involved in a DoS attack. If so, the target node for confirmation in a link local network may be undergoing attack by a large number of useless packets and may therefore not receive the ARP probe packet from our system. In this case, the packet will be designated as a spoofed packet, because the system will not have received an ARP reply packet from the sender node within the specified period of time. To avoid such cases of DoS attack, our system should employ a device such as IDS to detect meaningless DoS attacks.

## 5 Conclusion

In this paper, we have proposed source address validation support for network forensics. Our proposed system retains effective evidential information about the network users. Therefore, it should prove extremely useful for network operators in dealing with malicious events, even if much time has elapsed since their occurrence.

Our testbed results show that our proposed system effectively detects the location of the source node for many attack conditions. Furthermore, by recording the results of packet checking, and of the source IP/MAC address pair of the packet corresponding to the switch port number including logged time, our system will retain effective evidence about the network users which will always indicate their real source nodes. Accordingly, our proposed algorithm can be applied as a network forensics system against various security incidents.

## References

- [1] Alberto Ornaghi and Marco Valleri. *Man in the middle attacks Demos*. Blackhat, Conference - USA, 2003.
- [2] Cisco Systems, Inc. *Configuring the Catalyst Switched Port Analyzer (SPAN) Feature*. Document ID: 10570, August 2006.
- [3] D. Brezinski and T. Killalea. *Guideline for Evidence collection and Archiving*. Internet RFC 3227, February 2002.
- [4] D. Bruschi, A. Ornaghi, and E. Rosti. *S-ARP: a Secure Address Resolution Protocol*. ICSAC IEEE, 2003.
- [5] David C. Plummer. *An Ethernet Address Resolution Protocol*. Internet RFC 826, November 1982.
- [6] David K. Y. Yau, John C. S. Lui, Feng Liang, and Yeung Yam. *Defending Against Distributed Denial of Service Attacks with Max-min Fair Server-centric Router Throttles*. IEEE/ACM Transactions on Networking, Vol. 13, No. 1, 2005.
- [7] David Moore, Geoffrey. M. Voelker, and Stefan Savage. *Inferring Internet Denial-of-Service Activity*. 2001.
- [8] Gary Palmer. *A Road Map for Digital Forensic Research*. Technical Report DTR-T001-01, the First Digital Forensic Research Workshop (DFRWS), November 2001.
- [9] J. Case, M. Fedor, M.Schoffsatll, and J.Davin. *A Simple Network Management Protocol (SNMP)*. Internet RFC 1157, May 1990.
- [10] J. Postel. *Internet Control Message Protocol*. Internet RFC 792, September 1981.
- [11] Jelena Mirkovic and Peter Reiher. *A Taxonomy of DDoS Attack and DDoS Defense Mechanisms*. ACM SIGCOMM Computer Communications Review, Volume 34, Number, April 2004.
- [12] John R. Vacca. *Computer Forensics: Computer Crime Scene Investigation, Second Edition*. Charles River Media, July 2005.
- [13] L. Blunk and J. Vollbrecht. *PPP Extensible Authentication Protocol (EAP)*. Internet RFC 2284, March 1998.
- [14] L. Todd Heberlein and Matt Bishop. *Attack Class: Address Spoofing*. Proceeding of the 19th National Information Systems Security Conference, pp. 371-377, October 1996.
- [15] Nathalie Weiler. *Honeypots for Distributed Denial of Service*. Proceeding of the Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02), 2002.
- [16] Nobuyoshi TANAKA, Kuji FUKUDA, Hiroaki NAKADA, and Hiroki SHIMOKAWA. *Development of PC Quarantine System*. NEC J. of Adv. Tech., Winter, 2005.
- [17] P. Ferguson and D. Senie. *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*. Internet RFC 2267, January 1998.
- [18] R. Droms. *Dynamic Host Configuration Protocol*. Internet RFC 2131, March 1997.
- [19] Robert T. Morris. *A Weakness in the 4.2BSD Unix TCP/IP Software*. Technical Report Computer Science No.117, AT and T Bell Labs, February 1985.
- [20] Sean Whalen. *An Introduction to ARP Spoofing*. Revision 1.82, April 2001.
- [21] S.M. Bellovin. *Security Problems in the TCP/IP Protocol Suite*. ACM Computer Communications Review, April 1989.
- [22] Thomas E. Daniels and Eugene H. Spafford. *Network Traffic Tracking System: Folly in the Large?* Proceedings of the 2000 Workshop on New Security Paradigms, Feb 2001.
- [23] Vern Paxson. *An analysis of Using Reflectors for Distributed Denial of Service Attacks*. ACM Computer Communication Review (CCR), July 2001.
- [24] William J. Meador. *Port-based authentication with IEEE Standard 802.1x*. 2004.