# SPOOFED ARP PACKETS DETECTION IN SWITCHED LAN NETWORKS

Zouheir Trabelsi and Khaled Shuaib
*College of Information Technology*
*UAE University, Al-Ain, UAE*

Keywords: Intrusions Detection Systems, Spoofed ARP, ARP Cache Poisoning, Packet Sniffers.

Abstract: Spoofed ARP packets are used by malicious users to redirect network's traffic to their hosts. The potential damage to a network from an attack of this nature can be very important. This paper discusses first how malicious users redirect network traffic using spoofed ARP packets. Then, the paper proposes a practical and efficient mechanism for detecting malicious hosts that are performing traffic redirection attack against other hosts in switched LAN networks. The proposed mechanism consists of sending first spoofed packets to the network's hosts. Then, by collecting and analyzing the responses packets, it is shown how hosts performing traffic redirection attack can be identified efficiently and accurately. The affect of the proposed mechanism on the performance of the network is discussed and shown to be minimal. The limits of current IDSs regarding their ability to detect malicious traffic redirection attack, based on spoofed ARP packets, in switched LAN networks are discussed. Our work is concerned with the detection of malicious network traffic redirection attack, at the Data Link layer. Other works proposed protection mechanisms against this attack, but at the Application layer, using cryptographic techniques and protocols.

## 1 INTRODUCTION

Traditionally, research in the area of information and communication security focused on helping developers of systems prevent security vulnerabilities in the systems they produce, before the systems are released to customers. In addition, in most studies on network security, only external attacks were being considered. All of these areas are of the outmost importance when it comes to information and communication security, but need to be complemented with research supporting those developing detection and recovery mechanisms, and studying internal threats. *MiM* attack is used by malicious internal users to sniff the network traffic between target hosts, in switched networks. Malicious users can tap in on the network traffic for information without the knowledge of the networks legitimate owner. This information can include passwords, e-mail messages, encryption keys, sequence numbers or other proprietary data, etc. Often, some of this information can be used to penetrate further into the network, or cause other severe damage. This underlines the importance of reliable *MiM* attack detection techniques that can aid network administrators, in detecting malicious sniffing activities in switched networks.

Today, IDSs, such as Snort (www.snort.org), have become a standard part of the security solutions, used to protect computing assets from hostile attacks. IDSs are able to detect many types of attacks, such as denial of services (DoS) attacks and IP spoofing attacks. But, their ability and reliability to detect some particular attacks are still questionable, notably the *MiM* attack. IDSs must be able to detect the *MiM* attack since malicious sniffing activities in switched networks rely on this attack (Trabelsi, 2004, Trabelsi, 2005). This paper discusses the limits of the main IDSs regarding the detection of the *MiM* attack, provides an understanding of how *MiM* attack works, and proposes a novel efficient mechanism for detecting *MiM* attack-based sniffing activities in switched networks. The proposed detection mechanism is based mainly on the generation of spoofed packets which are sent to the network's hosts. By collecting and analyzing the response packets, it will be possible to identify hosts performing *MiM* attack against the other hosts in the network. Even though, the proposed mechanism injects additional spoofed



Trabelsi Z. and Shuaib K. (2006).
SPOOFED ARP PACKETS DETECTION IN SWITCHED LAN NETWORKS.
In *Proceedings of the International Conference on Security and Cryptography*, pages 40-47
DOI: 10.5220/0002102400400047

packets in the network, the network will not be flooded and its performance will not be affected, since the number of the injected packets is relatively very limited. In addition, it will be shown that the proposed mechanism does not disturb the network activities of the hosts.

The rest of the paper is organized as following: Section 2 provides an overview of the related work done in this area. Section 3 gives a brief understanding of some networking concepts to lay the groundwork for what follows. Section 4 discusses the ARP cache poisoning attack. Section 5 discusses the *MiM* attack and how it is used for malicious sniffing activities in switched network. Section 6 presents an efficient mechanism for detecting hosts sniffing switched networks. Section 7 presents a tool, called *AntiMiM*, for detecting malicious sniffing hosts in switched networks, based on the proposed detection mechanism. Section 8 discusses the affect of the proposed detection mechnaism on the network performance. Finally, conclusions are made and some directions for future research are provided in section 9.

# 2 RELATED WORK

Although few tools and some newer IDSs can detect ARP anomalies, most of them do not specially target the ARP cache poisoning attack and the *MiM* attack. Those attacks are interesting to detect because they are highly intentional. No virus or worms use it, so when such attacks are in effect, there is certainly some human controlling them. Besides, most IDSs rely on Sniffer-based sensors to detect those attacks, greatly restricting their effectiveness in switched networks.

*Arpwatch* (ftp.ee.lbl.gov) is a tool that aims to keep sysadmins informed (usually via email) about changes in the IP/MAC mappings. This catches many interesting events, such as IP addresses being changed (by authorized personnel or not), MAC addresses being changed (either by software reconfiguration or by physically replacing Ethernet card), new machines being added to the network (because of gratuitous ARPs), common misconfigurations (like IP address conflicts), etc. To do that, *Arpwatch* monitors Ethernet activity and keeps a database of Ethernet MAC address/IP address pairs. By sniffing the network, *Arpwatch* tries to detect any packet that has an Ethernet MAC address/IP address pair which does not appear in *Arpwatch's* database. However, *Arpwatch* can't tell these non-malicious events apart from intentional ARP spoofing attacks. On large busy networks with overworked or lax sysadmins, where typically hundreds of ARP anomalies are reported daily, many real serious attacks may pass unchecked. In addition, *Arpwatch's* detection technique presumes that the host running *Arpwatch* has access to a monitoring port on the Switch (usually, known under the name of *SPAN* port, or *mirroring* port). Therefore, it would be more interesting and efficient to detect any ARP anomalies without the use of any access privilege or special ports on the Switches. In addition, there is usually a substantial performance impact when port mirroring is in effect; this strategy makes ARP spoofing detection based on sniffing not quite viable on switched networks. *Snort* is an open source network intrusion prevention and detection system utilizing a rule-driven language, which combines the benefits of signature, protocol and anomaly based inspection methods. *Snort* is able to detect ARP cache poisoning attack by checking each packet contents. If the analyzed packet has an Ethernet MAC address/IP address pair which does not appear in *Snort's* database, then the system administrator is alerted. Like *Arpwatch*, *Snort* is an intrusion detection sensor, that is, it should have access to a monitoring port or be placed in a location where it can see all the incoming and outgoing network traffic.

# 3 BACKGROUND

This section is aimed at providing a brief understanding of Address Resolution Protocol (ARP) and ARP cache.

## 3.1 ARP Protocol

To map a particular IP address to a given hardware address (MAC address) so that packets can be transmitted across a local network, systems use the Address Resolution Protocol (ARP) (Plummer, 1982). ARP messages are exchanged when one host knows the IP address of a remote host and wants to discover the remote host's MAC address. For example, if host 1 wants host 2's MAC address, it sends an ARP request message (Who has?) to the broadcast MAC address (*FF:FF:FF:FF:FF:FF*) and host 2 answers with his addresses (MAC and IP). Basically, an ARP message on an Ethernet/IP network has 8 important parameters: Ethernet header source and destination MAC address, Ethernet type (=0x0806 for ARP message), ARP message header source and destination IP address, source and destination MAC

address, operation code (1 for request), (2 for reply). It is important to mention that there is nothing specifying that there must be some consistency between the ARP header and the Ethernet header. That means you can provide uncorrelated addresses between these two headers. For example, the source MAC address in the Ethernet header can be different from the source MAC address in the ARP message header.

## 3.2 ARP Cache

Each host in a network segment has a table, called ARP cache, which maps IP addresses with their correspondent MAC addresses. There are two types of entries in an ARP cache, namely: Static entries which remain in the ARP cache until the system reboots and Dynamic entries which remain in the ARP cache for few minutes (this depends on the operating system (OS)) then they are removed if they are not referenced. New ARP requests allow to add them again in the ARP cache. Static entries mechanism is used unfortunately in small local area network (LAN). However, in large networks, the deployment and updates of static entries in the ARP caches of the hosts are non practical networking tasks. New entries in the ARP cache can be created or already existing entries can be updated by ARP request or reply messages.

# 4 ARP CACHE POISONING ATTACK

ARP cache poisoning attack is the malicious act, by a host in a LAN, of introducing a spurious IP address to MAC address mapping in another host's ARP cache. This can be done by manipulating directly the ARP cache of a target host, independently of the ARP messages sent by the target host. To do that, we can either add a new fake entry in the target host's ARP cache, or update an already existing entry by fake IP and MAC addresses.

## 4.1 Static ARP Cache Update

An efficient technique to protect an ARP cache against the ARP cache poisoning attack is to use static entries in the ARP cache. That is, the entries in the ARP cache cannot be updated by ARP request and reply packets, and do not expire. But, this can provide a wrong believe of security under some OSs, such as Windows 2000 and SunOS Solaris 5.9. In fact, those OSs marks static entries in their ARP caches, but authorize their updates by ARP request and reply packets. Consequently, such entries cannot be considered as static entries, but solely permanent entries in the ARP caches.

## 4.2 Dynamic ARP Cache Update

In principle, to corrupt the entries in the ARP cache of a target host, a malicious host generates ARP request or reply messages including fake IP and MAC addresses. However, in practice, the success of this malicious activity depends on the operation system of the target host. A malicious host may attempt to send fake ARP reply messages to a target host even though the malicious host did not receive any ARP request message from the target host. If the OS of the target host accepts the fake ARP reply message from the malicious host without checking whether or not an ARP request message has been generated before, then the received ARP reply message will corrupt the ARP cache of the target host with a fake MAC/IP entry. Alternatively, the malicious host may attempt to send ARP request messages, instead of ARP reply messages. .
Table 1 shows the result of an experiment we performed on several common OSs. The objective of this experiment is to identify which OSs with dynamic entries in the ARP caches are vulnerable to the ARP cache poisoning attack. Table 1 indicates

Table 1: Update of dynamic entries in the ARP caches using ARP request and reply messages.

| | Windows XP | | Windows 2000 | | Windows 2003 Server | | Linux 2.4 | | Linux 2.6 | | Free BSD 4.11 | | SunOS Solaris 5.9 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Does the entry exist in the ARP cache? | Yes | No | Yes | No | Yes | No | Yes | No | Yes | No | Yes | No | Yes | No |
| ARP request | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| ARP reply | ✓ | X | ✓ | ✓ | ✓ | X | ✓ | X | ✓ | X | ✓ | ✓ | ✓ | ✓ |

✓ : Means that the ARP request or reply message is accepted by the system and therefore allows the update or the creation of an entry. X : Means that the ARP request or reply message is rejected by the system and therefore does not allow the update and the creation of an entry.

clearly that: All tested OSs, except Windows 2000 and Free BSD 4.11, do not allow the creation of a new entry by an ARP reply message and all tested OSs allow the creation of a new entry by an ARP request message. However, if the entry exists already in the ARP cache, all tested OSs allow its update by an ARP reply (even in the absence of an ARP request) or request message. Therefore, when using ARP reply messages, the ARP cache poisoning attack becomes difficult to realize against most OSs. However, it remains indeed possible when using ARP request messages. Most common OSs are still vulnerable to the ARP cache poisoning attack (Sanai, 2004). Malicious users can first use ARP request messages to create fake MAC/IP entries in the ARP caches of their target hosts. Then, fake ARP reply massages are used to maintain the existence of the fake entries in the ARP caches of the target hosts.

## 5 THE MIM ATTACK

In shared networks, when the Network Interface Card (NIC) is set to the promiscuous mode all the traffic can be sniffed, since each packet is broadcasted to all hosts. However, in a switched network, even by setting their hosts' NIC cards into the promiscuous mode, hackers cannot capture all the traffic in the network, because all packets sent by a host will be received only by the destination host, unless it is a broadcast packet. The *MiM* attack allows a malicious user to sniff a switched network. The attack consists into rerouting (redirecting) the network traffic between two target hosts to a malicious host. Then, the malicious host will forward the received packets to the original destination, so that the communication between the two target hosts will not be interrupted and the two hosts' users will not notice that their traffic is sniffed by a malicious host.

To do that, the malicious user should first enable his host's IP packet routing (IP packet forwarding), in order to become a router and be able to forward the redirected packets. Then, using ARP cache poisoning attack, the malicious user should corrupt the ARP caches of the two target hosts, in order to force the two hosts to forward all their packets to his host. It is important to notice that if the malicious host corrupts the ARP caches of two target hosts without enabling its IP packet routing, then the two hosts will not be able to exchange packets and it will be a DoS attack. In this case, the malicious host does not forward the received packets to their legitimate destination. This is extremely potent when we consider that not only can hosts be poisoned, but routers/gateways as well. All Internet traffic for a host could be intercepted by performing a *MiM* attack on the host and the network's router. The *MiM* attack is performed as follows (where C is the malicious host, and A and B are the two target hosts): Host C enables its IP packet routing and corrupts the ARP caches of hosts A and B, using ARP cache poisoning attack. Figure 1 (a) shows the initial entries in the ARP caches of hosts A and B, before the ARP cache poisoning attack. After the attack, host A associates host B's IP with host C's MAC, and host B associates host A's IP with host C's MAC (Figure 1 (b)). Consequently, all the packets exchanged between hosts A and B will first go to host C. Then, host C forwards them to the legitimate destination (host B or host A).

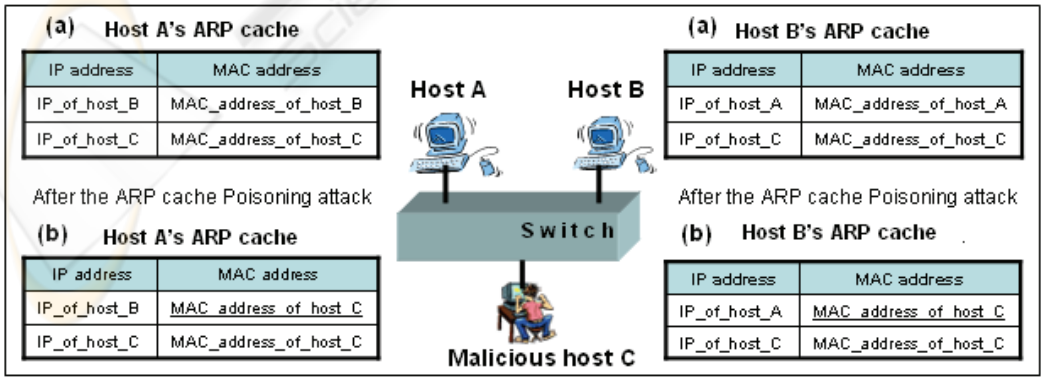


Figure 1: The entries of the ARP caches of hosts A and B before and after the ARP poisoning attack.

# 6 DETECTION OF MIM ATTACK

To perform a *MiM* attack, the malicious host should enable the IP packet routing and corrupt the ARP caches of its target hosts. Therefore, if IP packet routing is enabled in a host, then it is a suspicious host. Only routers and some servers need to enable the IP packet routing since they have to perform packets routing. But still, we need to prove that the suspicious host has performed also ARP cache poisoning attacks against target hosts. In section 6.1, we discuss an efficient technique for detecting any host in the network with enabled IP packet routing. Hosts with enabled IP packet routing are called *suspicious hosts*. Then, in section 6.2, we discuss a technique for detecting, among the *suspicious hosts,* those that have performed ARP cache poisoning attacks against other hosts in the network.

## 6.1 Detection of Hosts with Enabled IP Packet Routing

The following is a technique based on an algorithm for detecting any host with enabled IP packet routing, in a switched network. The technique consists of two phases and assumes that there is a host in the network, *Test host*, used to do all the tests needed during the two phases.

***Phase 1:*** *Generation of trap ICMP echo request packets.* In this phase, the *Test host* sends a trap ICMP Ping packet to a given target host in the network. Usually, if a host A (whose IP address is IP_A and MAC address is MAC_A) wants to Ping a host B (whose IP address is IP_B and MAC address is MAC_B) in a network, then host A should send to host B an ICMP echo request packet. A Ping packet is used to detect whether or not a host is connected to the network (Postel, 1981, Stevens, 2001). For host A, there is no reason and it is meaningless to send a Ping packet to itself, since this means that host A wants to know whether or not it is connected to the network. Usually, hosts use other mechanisms to perform any networking tests on their NIC cards, such as the use of Loop IP addresses (e.g.:127.0.0.1). When host A wants to send to itself a Ping packet, it should set the "*Destination MAC address*" field in the Ethernet header and the "*Destination IP address*" field in the IP header to the values "*MAC_ A*" and "*IP_A* ", respectively. However, the *Test host* attempts to Ping itself, using a *trap* ICMP echo request packet. In fact, the *Test host* wants to send this packet to each host in the network. Therefore, the value of the "*Destination MAC address*" field in the Ethernet header of the trap ICMP Ping packet is set to the MAC address of the target host in the network (*MAC_Target_host*). The value of the "*Destination IP address*" field in the IP header of the packet is kept as usual, that is the IP address of the *Test host* (*IP_Test_host*). The values of the main fields of the trap ICMP Ping packet are as shown in Table 2.

Table 2: Trap ICMP Ping packet.

| Ethernet header | |
|---|---|
| Source MAC address = | *MAC_ Test_host* |
| Destination MAC address = | *MAC_ Target_host* |
| Ethernet Type = | *0x0800 ( IP message)* |
| IP header | |
| Source IP address = | *IP_Test_host* |
| Destination IP address = | *IP_Test_host* |
| ICMP header | |
| Type = | 8 (echo request) |
| Code = | 0 |

The NIC card of each target host in the network will receive a trap ICMP Ping packet sent by the *Test host*. Since, the MAC address in the "*Destination MAC address*" field of the trap packet matches the MAC address of the target host, then the packet will be accepted by the target host and sent to the IP layer for processing. Since, the IP address in the "*Destination IP address*" field does not match the IP address of the target host, then the target host will either discard or forward the packet to the host whose IP address is specified in the "*Destination IP address*" field. If the target host is set to do IP packet routing, then the packet will be forwarded to the *Test host, o*therwise, the packet is discarded. Table 3 shows the results of an exercise we performed on several common OSs. This experiment confirms that when the IP packet routing is enabled all tested OSs forward the received trap ICMP Ping packet.

Table 3: Responses of the tested OSs after receiving the trap ICMP Ping packet.

| | Windows XP | | Windows 2000 | | Windows 2003 Server | | Linux 2.4 | | Linux 2.6 | | Free BSD 4.11 | | SunOS Solaris 5.9 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Enabled IP packet routing? | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* |
| The target host forwards the trap ICMP Ping packet to the original host? | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* | *Yes* | *No* |

When the target host is set to do IP packet routing, it will forward the original packet with the same IP and ICMP headers, but with a different Ethernet header, as a router did. The destination MAC address will be set to the MAC address of the *Test host*, and the source MAC address will be set to the MAC address of the target host.

***Phase 2:*** *Detection of the hosts with enabled IP packet routing.* All hosts that send back the received trap ICMP packets to the *Test host*, have enabled IP packet routing, and consequently are considered as *suspicious hosts*. In order to capture the forwarded trap ICMP packets sent by the target hosts, the *Test host* may use a Sniffer or any program that allow to capture ICMP Ping packets (echo request) whose "*Destination IP address"* field is set to the IP address of the *Test host*.

## 6.2 Detection of ARP Cache Poisoning Attack

In section 6.1, we discussed a technique for detecting the hosts with enabled IP packet routing. Those hosts are classified as *suspicious hosts.* In the following section, we discuss a technique for detecting the hosts, among the *suspicious hosts*, that have performed ARP cache poisoning attack against other hosts in the network. If a host has enabled IP packet routing (*suspicious host)* and has performed ARP cache poisoning attack against other hosts in the network, then we can conclude that the host is most likely sniffing the network traffic. This technique consists into corrupting the ARP cache of a *suspicious host* in order to force it to forward to the *Test host* the packets received from its victim hosts. It will be demonstrated that by analyzing the traffic generated by a *suspicious* host, it is possible to identify whether or not the *suspicious host* has performed ARP cache poisoning attacks against target hosts in the network.

We assume that A, B, C, D and E are the hosts of a network. Hosts A and B are exchanging network traffic. The IP packet routing in host D is enabled. In addition, host D has corrupted the ARP caches of the two hosts A and B, in order to sniff their traffic. However, the ARP cache of host C is not corrupted since it is not the target of the malicious host D. Host E is the *Test host*. The detection technique described in section 6.1 allows us to identify host D as a suspicious host, since its IP packet routing is enabled. The initial values of the entries (IP/MAC) in the ARP caches of hosts A, B, C and D, before the process of the corruption of the ARP cache of any *suspicious host*, are:
The ARP cache of host A (corrupted ARP cache) i.e. IP_B – MAC_D, the ARP cache of host B (corrupted ARP cache) i.e. IP_A – MAC_D, the ARP cache of host C i.e. IP_A – MAC_A and IP_B – MAC_B and the ARP cache of host D i.e. IP_A – MAC_A and IP_B – MAC_B.

For each suspicious host, we corrupt first its ARP cache, using ARP cache poisoning attack. To do that, the *Test host E* sends fake ARP requests to the *suspicious host* D. So that, all the entries IP/MAC in the ARP cache of host D will have the MAC address of the *Test host* E as MAC address (MAC_E). After this attack, the entries in the ARP cache of the *suspicious host* D become corrupted: The ARP cache of host D (corrupted ARP cache) i.e. IP_A – MAC_E, IP_B – MAC_E and IP_C – MAC_E

Consequently, any packet sent by host A to host B will go first to the *suspicious host* D, since the ARP caches of hosts A and B have been corrupted before by the *suspicious host D* (Figure 2: arrow (1)). Then, the *suspicious host* D forwards the received packet to the *Test host* E, since the ARP cache of the suspicious host D has been corrupted by the *Test host E (*Figure 2*:* arrow (2)*)*. Finally, the *Test host E* takes a copy of the received packet and forwards it to the legitimate destination host, which is host B (Figure 2: arrow (3)).

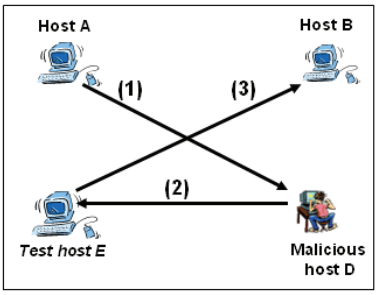


Figure 2: The network path of the packets sent by host A to host B.

By analyzing the packet sent by the *suspicious host* D to the *Test host* E, we can easily reveal that the source IP address in the IP header of the packet (2) is of host A, but the source MAC address in the Ethernet header of the packet is of the *suspicious host* D; however it should be equal to the MAC address of host A. Consequently, we have the prove that the *suspicious host* D has corrupted before the cache ARP of host A in order to sniff its traffic. If

the *Test host* receives only legitimate packets from the *suspicious host* D, that is the packets' source IP is host D's IP and the packets' destination IP is the *Test host* E's IP, then we can conclude that the *suspicious* host D has enabled IP packet routing but it is not sniffing any network traffic. Host D may have enabled IP packet routing by accident without any intention to perform malicious sniffing activities against other network hosts. The whole process of detection is then repeated for the remaining identified suspicious hosts.

# 7 THE ANTIMIM APPLICATION

Based on the proposed detection techniques, an application, called *AntiMiM*, has been developed using Visual C++6.0 and WinpCap Library. The application detects any host sniffing the switched network using the *MiM* attack. *AntiMiM* application has been evaluated against the two IDSs *Arpwatch* and *Snort*. The application does not require a monitoring port (SPAN) to run, unlike *Snort* and *Arpwatch*. In addition, *Snort* and *Arpwatch* are not able to detect the network's hosts with enabled IP packet routing. Finally, *AntiMiM* does not require a predefined database of valid IP/MAC entries, like *Snort* and *Arpwatch*. The database is used to verify whether or not a given IP/MAC pair found in a captured packet belongs to the database. Usually, such a packet is used when performing ARP cache poisoning attack. When *Snort* or *Arpwatch* are used to detect ARP cache poisoning attack, the network administrator should provide them with a database of valid IP/MAC pairs. The generation of such a database is times consuming. In addition, in large networks, the database may include erroneous entries. When a new host is connected to the network, or a host gets a new MAC address (after changing its NIC card) or IP address, the database should be updated.

# 8 NETWORK PERFORMANCE ANALYSIS

The proposed detection mechanism uses two techniques that attempt to send spoofed packets to the network's hosts and then collect the response packets for analysis. Therefore, for the efficiency of the proposed mechanism, it is important to compute the number of packets injected in the network. If the network is flooded with heavy traffic, then its performance may be affected.

We assume that there are *n* hosts in the network including the *Test host* used to perform all the tests (refer to sections 6.1 and 6.2). The proposed mechanism detects first, among the *n* hosts, the hosts with enabled IP packet routing, using the technique discussed in section 6.1. We assume that *m* hosts with enabled IP packet routing have been identified (called *suspicious hosts*). Hence, the *Test host* will send (*n-1*) trap ICMP echo request packets to the (*n-1*) hosts in the network. Only, *m* hosts will forward back the received packets, since they have enabled IP packet routing. Therefore, (*(n-1) + m)* packets are injected in the network, while detecting the hosts with enabled IP packet routing.

Then, the proposed mechanism attempts to detect among the *m* identified suspicious hosts, the hosts that have performed ARP cache poisoning attack against the other hosts. When the technique of the section 6.2 is used, the *Test host* sends fake ARP requests to the suspicious hosts in order to corrupt all the entries in their cache. Since there are *n* hosts including *m* suspicious hosts in the network, and the maximum number of IP/MAC entries in an ARP cache is (n-1), the *Test host* needs to generate ((n-1)*m) fake ARP request packets. In addition, since the ARP cache entries are supposed to expire if they are not referenced within few minutes (typically between tens of seconds to a few minutes, according to the OS), then the *Test host* should keep sending fake ARP requests periodically. As long as the *Test host* keeps doing this, the suspicious hosts will not issue ARP requests for that IP addresses, since their ARP cache entries will always be within the timeout threshold. Therefore, if the *Test host* waits a period of 10 seconds, for example, and then sends again the ((n-1) *m) fake ARP packets, and the detection process will take 1 minutes, then the *Test host* will inject (*((n-1)*m)*6)* packets in the network.

Consequently, the use of the proposed techniques does not degrade the network performance since they do not flood the network with heavy traffic. On the other hand, the techniques are independent from the Switch brand and model, since they are based on the attack of the ARP caches of the suspicious hosts.

# 9 CONCLUSION

Throughout this paper, we demonstrated that sniffing is still a big thread even in a switched network. This is against the belief that switched network are safe from sniffing activities.

Most of the works done in the area of malicious sniffing activities detection, deal with Sniffers in shared networks. Few IDSs, mainly *Snort* and *Arpwatch*, tried to detect malicious sniffing activities in switched networks. The limits of those IDSs regarding their ability to detect properly *MiM* attacks have been discussed.

The paper proposes a mechanism for detecting sniffing hosts in switched networks. The hosts use the *MiM* attack, to sniff switched network. The proposed mechanism consists into sending spoofed packets to the network's hosts. By collecting and analyzing the responses packets, it has been demonstrated how malicious sniffing hosts can be identified efficiently and accurately. The mechanism does not degrade the network performance when injecting packets into to the network, since the number of injected packets is relatively very limited.

Future works will be to make an IDS plug-in version of the proposed mechanism, such as *Snort* plug-in, and to lower the false positive/negative ratios when the hosts use personal firewalls to filter the incoming and outgoing traffic.

## REFERENCES

ftp://ftp.ee.lbl.gov/arpwatch.tar.gz

http://www.snort.org

Plummer, David C, 1982. An Ethernet Address Resolution Protocol-Converting Network Protocol to 48 bit Ethernet Address for Transmission on Ethernet Hardware, RFC-826.

Postel, J, 1981. Internet Control Message Protocol, RFC-792.

Richard Stevens, 2001. TCP/IP Illustrated: Volume 1.

Daiji Sanai, 2004. Detection of Promiscuous Nodes Using ARP Packets, http://www.securityfriday.com

Zouheir Trabelsi *et.al*., 2004. Detection of Sniffers in an Ethernet Network, in the 7th Information Security Conference, ISC 2004, Palo Alto, CA, USA.

Zouheir Trabelsi *et.al*, 2005. An Anti-Sniffer Based on ARP Cache Poisoning Attack, in the Information System Security Journal, Auerbach Publications, NY, USA, Volume 13, Issue 6.