Packet Sniffing: Network Wiretapping

Article · March 2009

CITATIONS
3

READS
22,395

3 authors, including:

Nimisha Patel
Sankalchand Patel College of Engineering
21 PUBLICATIONS **190** CITATIONS
SEE PROFILE

Rajan Patel
Gandhinagar Institute of Technology
37 PUBLICATIONS **219** CITATIONS
SEE PROFILE

2009 IEEE International Advance Computing Conference (IACC 2009)
Patiala, India, 6–7 March 2009

Packet Sniffing: Network Wiretapping

Nimisha P. Patel, Rajan G. Patel, Dr. Dhiren R. Patel

Computer Engineering Department, Sardar Vallabhbhai National Institute of Technology, Surat-395007, India.

***Abstract*-Packet sniffing is a method of tapping each packet as it flows across the network. It is a technique in which a user sniffs data belonging to other users of the network. It is effective on both switched network and non-switched network. Packet sniffers can operate as an administrative tool or for malicious purposes. It depends on the user's intent. This paper discusses how sniffing can be done in case of hub and switched network, various packet sniffing methods, different methods that AntiSniff uses to detect these sniffing programs are also discussed.**

1. INTRODUCTION

Packet sniffing is a technique of monitoring network traffic [1]. In LANs, packet sniffing and remote network monitoring (RMON) are well-known techniques used by network administrators to monitor LAN behavior and diagnose troubles [2]. It is effective on both switched and non switched networks. In a non-switched network environment packet sniffing is an easy thing to do. This is because network traffic is sent to a hub which broadcasts it to everyone. Switched networks are completely different in the way they operate. Switches work by sending traffic to the destination host only. This happens because switches have CAM (Content Addressable Memory) tables. The CAM table is a system memory construct used by ethernet switch logic to dereference Media Access Control (MAC) addresses of stations to the ports on which they connect to the switch itself. Before sending traffic from one host to another on the same local area network, the host's ARP cache is first checked. The ARP cache is a table that stores both layer 2 (MAC) addresses and layer 3 (IP) addresses of hosts on the local network. If the destination host isn't in the ARP cache, the source host sends a broadcast ARP request looking for the host. When the host replies, the traffic can be sent to it. The traffic goes from the source host to the switch, and then directly to the destination host [1]. This description shows that traffic isn't broadcast out to every host, but only to the destination host, therefore it's harder to sniff traffic.

In its simple form a packet sniffer simply captures all of the packets of data that pass through a given network interface. Typically, the packet sniffer would only capture packets that were intended for the machine in question. However, if placed into promiscuous mode, the packet sniffer is also capable of capturing all packets traversing the network regardless of destination. Promiscuous mode is a configuration of a network card that makes the card pass all traffic it receives to the central processing unit rather than just packets addressed to it. By placing a packet sniffer on a network in promiscuous mode, a malicious intruder can capture and analyze all of the network traffic. Within a given network, username and password information is generally transmitted in clear text which means that the information would be viewable by analyzing the packets being transmitted. A packet sniffer, sometimes referred to as a network monitor or network analyzer, can be used legitimately by a network or system administrator to monitor and troubleshoot network traffic. Pcap is a network capture file format often used in packet sniffer such as Ethereal [3]. There are many open source packet sniffer tools available such as tcpdump, wireshark, ettercap, snort etc.

The rest of paper organized as follows: section 2 describes packet sniffing and use of packet sniffing programs, section 3 discusses how sniffing can be performed around hub and in a switched network, section 4 describes various packet sniffing methods, section 5 describes sniffer detection and finally conclusion in section 6.

2. PACKET SNIFFING AND USE OF PACKET SNIFFING PROGRAMS

Packet sniffing is the act of capturing packets of data flowing across a computer network. Packet sniffing is to computer networks what wire tapping is to a telephone network. This is done through the use of packet sniffers, which are devices that can be plugged into a network and used to eavesdrop on the network traffic [4]. Using the information captured by the packet sniffer an administrator can identify erroneous packets and use the data to pinpoint bottlenecks and help to maintain efficient network data transmission. However, it is also widely used by hackers and crackers to gather information illegally about networks they intend to break into. Using a packet sniffer it is possible to capture data like passwords, IP addresses, protocols being used on the network and other information that will help the attacker infiltrate the network. Packet sniffing is primarily used in intrusion detection, network management, wiretapping and

hacking [5]. Password sniffing programs are most popular packet sniffing program.

Today's networks may already contain built-in sniffing modules. Most hubs support the RMON standard, which allow the intruder to sniff remotely using SNMP, which has weak authentication. Many corporations employ Network Associates "Distributed Sniffer Servers", which are set up with easy to guess passwords. Windows NT machines often have a "Network Monitoring Agent" installed, which again allows for remote sniffing. A packet sniffing is difficult to detect, but it can be done. But the difficulty of the solution means that in practice, it is rarely done. The popularity of packet sniffing stems from the fact that it sees everything. Use of packet sniffing program includes [6]:

- Logging network traffic.
- Solving communication problems such as: finding out why computer A cannot communicate with computer B. (e.g. the communication may not be possible because of various reasons, such as a problem in either the system or the transmission medium.)
- Analyzing network performance. This way the bottlenecks present in the network can be discovered, or the part of the network where data is lost (due to network congestion) can be found.
- Retrieving user-names and passwords of people logging onto the network.
- Detecting network intruders.

3. Sniffing Around The Hub And Sniffing In A Switched Network

3.1. Sniffing around the Hub

Sniffing on a network that has hubs installed is a dream for any packet analyst. Traffic sent through a hub is sent to every port connected to that hub. Therefore, to analyze a computer on a hub, just plug in a packet sniffer to an empty port on the hub, and it will allow seeing all communication to and from all computers connected to that hub. As illustrated in figure 1 [7], visibility window is limitless when sniffer is connected to a hub network.

3.2. Sniffing in a switched network

A switched environment is the most common type of network. Switches provide an efficient means of transporting data via broadcast, unicast, and multicast traffic. As a bonus, switches allow full-duplex communication, meaning that machines can send and receive data simultaneously through a switch [7]. Unfortunately for packet analysts, switches add a whole new level of complexity to a packet analyst's job. In a switched network environment, packets are only sent to the port they are destined to, according to their destination MAC addresses.

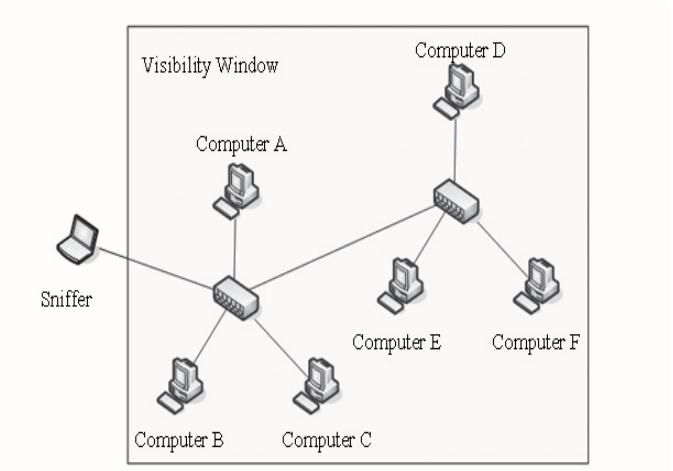

Fig. 1. Sniffing on a hub network provides a limitless visibility window.

The advantage of a switched environment is that devices are only sent packets that are meant for them, meaning that promiscuous devices aren't able to sniff any additional packets. When sniffer is plugged to a port on a switch, it will allow to see only broadcast traffic and the traffic transmitted and received by that machine, as shown in figure 2 [7].There are three primary ways to capture traffic from a target device on a switched network: port mirroring, hubbing out and ARP cache poisoning.

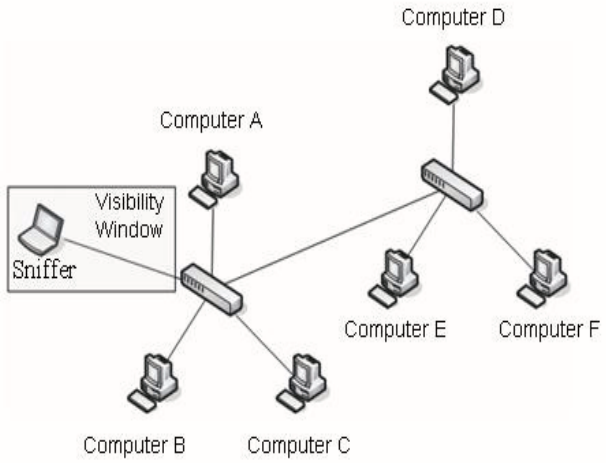

Fig. 2. The visibility window on a switched network is limited to the port on which sniffer is plugged.

A. Port Mirroring

A Port Monitor is a program that is functioning in the background while the other programs are executed [8]. Port mirroring, or port spanning as it is often called, is perhaps the easiest way to capture the traffic from a target device on a switched network. In this type of setup, access to the command-line interface of the switch is required on which the target computer is

located [7]. Also, the switch must support port mirroring and have an empty port into which an analyzer can be plugged.

When port mirroring, log into the command-line interface for switch and enter a command that forces the switch to copy all traffic on a certain port to another port (see figure 3[7]). The exact command to set up port mirroring will vary depending on the manufacturer of the switch. For instance, to capture the traffic from a device on port three of a switch, simply plug the analyzer into port four and mirror port three to port four. This would allow seeing all traffic transmitted and received by target device.

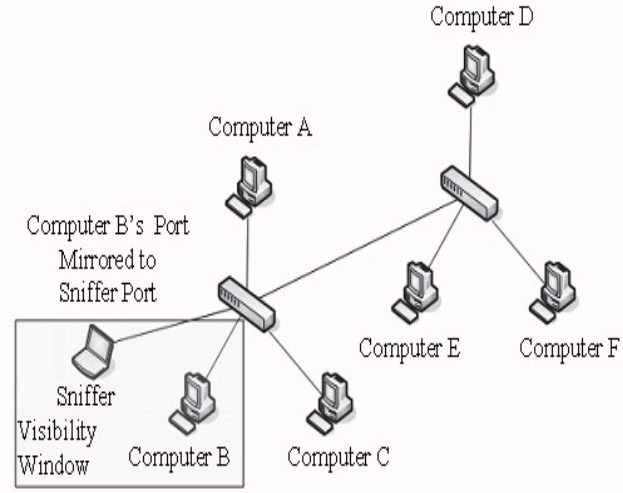

Fig. 3. Port mirroring allows to expand visibility window on a switched network.

When port mirroring, be aware of the throughput of the ports which are mirrored. Some switch manufacturers allow to mirror multiple ports to one individual port, which may be very useful when analyzing the communication between two or more devices on a single switch.

B. Hubbing Out

Another very simple way of capturing the traffic through a target device on a switched network is by hubbing out. Hubbing out is a technique in which localize the target device and analyzer system on the same network segment by plugging them directly into a hub [7]. Many people think of hubbing out as cheating, but it's really a perfect solution in situations where we can't perform port mirroring but still have physical access to the switch the target device is plugged into.

In order to hub out, we need a hub and a few network cables. Once hardware is available, go to the switch the target device resides on and unplug the target from the network. Then plug the target's network cable into hub, and plug in another cable connecting analyzer. Next, connect the hub to the network by plugging in a network cable from it to the network switch. Now we have basically put the target device and analyzer into the same broadcast domain, and all traffic from the target device will be broadcast so that the analyzer can capture those packets (see figure 4 [7]).

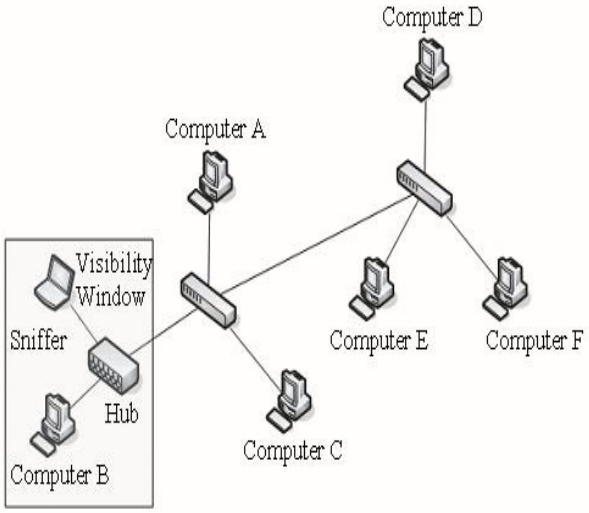

Fig. 4. Hubbing out isolates target device and analyzer on their own broadcast domain.

When hubbing out, be sure that you're using a true hub and not a falsely labeled switch. Several networking hardware vendors have a bad habit of marketing and selling a device as a hub when it actually functions as a low-level switch. If we aren't working with a proven, tested hub, we will only see our own traffic, not that of the target device. The best way to determine whether or not the device we are using is a true hub is to hook a pair of computers up to it and see if one can sniff the other's traffic. If so, we have a true hub.

ARP cache poisoning is explained in section 4.

4. PACKET SNIFFING METHODS

There are three types of sniffing methods. Some methods work in non-switched networks while others work in switched networks [9]. The sniffing methods are:

4.1. IP-based sniffing

This is the original way of packet sniffing. It works by putting the network card into promiscuous mode and sniffing all packets matching the IP address filter. Normally, the IP address filter isn't set so it can capture all the packets [10]. This method only works in non-switched networks.

4.2. MAC-based sniffing

This method works by putting the network card into promiscuous mode and sniffing all packets matching the MAC address filter [10].

4.3. ARP-based sniffing

This method works a little different. It doesn't put the network card into promiscuous mode. This happens because the ARP protocol is stateless. Because of this, sniffing can be done on a switched network [9]. To perform this kind of sniffing, first poison the ARP cache of the two hosts that we want to sniff, identifying yourself as the attacker host in the connection. Once the ARP caches are poisoned, the two hosts start their connection, but instead of sending the traffic directly to the other host it gets sent to attacker host. Attacker then logs the traffic and forwards it to the real intended host on the other side of the connection. This is called a man-in-the-middle attack [10]. A man-in-the-middle attack is, in the scope of a LAN, a technique where an attacker is able to redirect all traffic between two hosts of that same LAN for packet sniffing or data manipulation, without the end hosts being aware of it [11]. See figure 5 [10] for a general idea of the way it works.

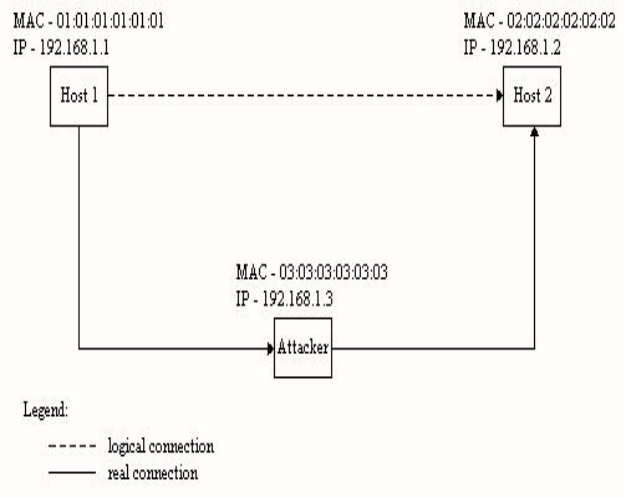

Fig. 5. ARP sniffing method

A. ARP Cache Poisoning

ARP cache poisoning is the process of falsifying the source MAC addresses of packets being sent on an Ethernet network [12]. In other words, when a host adds an incorrect <IP, MAC> mapping to its ARP cache, this is known as ARP cache poisoning (or simply ARP poisoning) or ARP spoofing [13]. It is a MAC layer attack that can only be carried out when an attacker is connected to the same local network as the target machines, limiting its effectiveness only to networks connected with switches, hubs, and bridges; not routers [12]. By performing ARP cache poisoning, an intruder can impersonate another host and gain access to sensitive information [13].

The principle of ARP spoofing is to send fake, or "spoofed", ARP messages to an Ethernet LAN. Generally, the aim is to associate the attacker's MAC address with the IP address of another node. Any traffic meant for that IP address would be mistakenly sent to the attacker instead. The attacker could then choose to forward the traffic to the actual default gateway (passive sniffing) or modify the data before forwarding it (man-in-the-middle attack) [12]. The attacker could also launch a denial-of-service attack against a victim by associating a nonexistent MAC address to the IP address of the victim's default gateway.

The attacker: 10.0.0.1
MAC address: 00-AA-BB-CC-DD-00
The victim 1: 10.0.0.2
MAC address: 00-AA-BB-CC-DD-E1
The victim 2: 10.0.0.3
MAC address: 00-AA-BB-CC-DD-E2

All the attacker actually does is sending a crafted packet to 10.0.0.2 with spoofed IP of 10.0.0.3 and his own MAC address and then it sends a crafted packet to 10.0.0.3 with spoofed IP of 10.0.0.2 with his own MAC. Table 1 shows ARP table of victim 10.0.0.2 and table 2 shows ARP table of victim 10.0.0.3.

Table 1. ARP cache of victim 10.0.0.2

IP Address	MAC Address
10.0.0.3	00-AA-BB-CC-DD-00

Table 2. ARP cache of victim 10.0.0.3

IP Address	MAC Address
10.0.0.2	00-AA-BB-CC-DD-00

Now all the traffic between those two hosts will go through the attacker first. So this means that the attack will need to reroute the packets to the real destination else we get a DOS on the network and there will be no traffic possible. Also remember that the ARP tables get updated so if during a long period of time there is no ARP poisoning the entries will be deleted and we won't be able to sniff until start poisoning again [12]. The idea was to sniff the traffic between machine 10.0.0.2 and 10.0.0.3 with machine 10.0.0.1.

5. SNIFFER DETECTION

There are quite a few methods to detect sniffers on the network.

• The user can generate packets that contain an invalid address. If a machine (on the network) accepts the packets, it can be concluded that the machine is running a sniffer. Here is one method to do it: The user can change the MAC address of his or her machine temporarily. Sending packets to the old address of the machine should not result in the acceptance of that

packet. If a machine does accept the packet, then it is running a sniffer [6].

• Software programs such as AntiSniff can be used to detect sniffers. According to a review by Dave Kearns (Network world on NT), "AntiSniff gives the ability to remotely detect computers that are packet sniffing. By running a number of non-intrusive tests in a variety of ways network administrators and information security professionals can determine whether or not a remote computer is listening in on all network communications" [6]. AntiSniff is a tool made by L0pht Heavy Industries designed to detect hosts on an Ethernet/IP network segment that are promiscuously gathering data [9]. The tests are broken down into the following three classes: DNS tests, operating system specific tests, and network and machine latency tests [14].

5.1. DNS test

This test is here because many packet sniffing tools perform IP address to name lookups to provide DNS names in place of IP addresses. This information is useful to attackers because most of the time hosts are named for what they provide. An example would be a mail server being named mail.abc.com. Hosts not watching traffic destined to them will not attempt to resolve the IP addresses in the packets. To test this, AntiSniff places the network card into promiscuous mode and sends packets out onto the network aimed to bogus hosts [10] [15]. If any name lookups from the bogus hosts are seen, a sniffer might be in action on the host performing the lookups.

5.2. Operating system specific tests

This class of tests is aimed at certain operating systems. There is the ARP test that is designed for Microsoft Windows 95, 98, and NT. The second test is known as the Ether Ping test which is designed for Linux and NetBSD kernels [10]. Each test will be described below.

A. ARP test

This test is to exploit the flaw found in the way Microsoft operating systems analyze broadcast ARP packets. This is found in Microsoft Windows 95, 98, and NT. This method uses ARP request packets that are sent to a target with an incorrect MAC address that has the group bit set. The group bit indicates if a MAC address is an individual address or a group address. When in promiscuous mode the driver for the network card checks for the MAC address being that of the network card for unicast packets, but only checks the first octet of the MAC address against the value 0xff to determine if the packet is broadcast or not. Note that the address for a broadcast packet is ff:ff:ff:ff:ff:ff [10]. To test for this flaw, AntiSniff sends a packet with a MAC address of ff:00:00:00:00:00 and the correct destination IP address of the host. After receiving a packet, the Microsoft OS using the flawed driver will respond while in promiscuous mode. It should be noted that this flaw is based on the default Microsoft driver shipped with the OS.

B. Ether Ping test

In older Linux kernels there is a specific condition that allows users to determine whether a host is in promiscuous mode or not. The Ether ping test exploits the flaw in how many operating systems analyze ICMP packets [15]. When a network card is placed in promiscuous mode every packet is passed on to the OS. Some Linux kernels looked only at the IP address in the packets to determine whether they should be processed or not. To test for this flaw, AntiSniff sends a packet with a bogus MAC address and a valid IP address. Vulnerable Linux kernels with their network cards in promiscuous mode only look at the valid IP address [10]. To get a response, an ICMP echo request message is sent within the bogus packet leading to vulnerable hosts in promiscuous mode to respond.

5.3. Network and machine latency tests

These last sets of tests are here because hosts in promiscuous mode don't have low level hardware filtering. This dramatically increases network traffic not meant for the host leading to the OS kernel to do the filtering. The increased filtering done by the kernel causes more latency. ICMP Time Delta test, Echo test, and the Ping Drop test are the Network and machine latency tests [10] [15].

A. ICMP Time Delta test

This test uses baseline results to determine network and machine latency. AntiSniff probes the host by sending ICMP echo request messages with microsecond timers to determine the average response time. After the baseline has been created, it floods the local network with non-legitimate traffic. During the flood of traffic; it sends another round of ICMP echo request probes to determine the average response time [10]. Hosts in promiscuous mode have a much higher latency time.

B. Echo test

This test is actually an option for the ICMP Time Delta test. The user has the option to use the ECHO service for time deltas, if it's available on the remote host [10].

C. Ping Drop test

This test is also run during the flood of traffic. It involves sending a large amount of ICMP echo request messages to the host. It keeps track of the number of

dropped ping responses [10]. When a host is in promiscuous mode it will have a much higher level of network traffic to process leading to network latency which causes the host to drop packets because it can't keep up.

6. CONCLUSION

When computers communicate over networks, they normally just listen to the traffic specifically for them. However, network cards have the ability to enter promiscuous mode, which allows them to listen to all network traffic regardless of if it's directed to them. Packet sniffers can capture things like clear-text passwords and usernames or other sensitive information. Because of this packet sniffers are a serious matter for network security. Since sniffing is possible on non-switched and switched networks, it's a good practice to encrypt your data communications. User can employ a number of techniques to detect sniffers on the network and protect the data from sniffers.

REFERENCES

[1] Ryan Spangler, "Packet Sniffing on Layer 2 Switched Local Area Networks", Packetwatch Research, December 2003.

[2] Thomas M. Chen, Lucia Hu, "Internet Performance Monitoring", Proceedings of the IEEE, pp. 1592-1603, VOL. 90, NO. 9, September 2002.

[3] Aaron Lanoy and Gordon W. Romney, "A Virtual Honey Net as a Teaching Resource", 7th International Conference on Information Technology Based Higher Education and Training, (ITHET '06), IEEE, July 2006.

[4] Greg Barnett, Daniel Lopez, Shana Sult, Michael Vanderford, "Packet Sniffing: Network Wiretapping", Group project, INFO 3229-001, 2002.

[5] A. Meehan, G. Manes, L. Davis, J. Hale, S. Shenoi, "Packet Sniffing for Automated Chat Room Monitoring and Evidence Preservation", Proceedings of the Second annual IEEE Systems, Man and Cybernetics Information Assurance Workshop, New York, pp. 285-288, June 2001.

[6] Sabeel Ansari, Rajeev S.G., Chandrashekar H.S, "Packet Sniffing: A Brief Introduction", VOL. 21, pp. 17-19, IEEE, December 2002.

[7] Chris Senders, Practical Packet Analysis, using Wireshark to solve real-world network problems, No Starch Press Inc, San Francisco, 2007.

[8] Dick Hazeleger, "Packet Sniffing: A Crash Course", Netherlands, 2001.

[9] Raed Alomoudi, Long Trinh, Darleen Spivey, "Protecting Vulnerabilities or Online Intrusion: The Efficacy of Packet Sniffing in the Workplace", Florida Atlantic University ISM 4320, 2004.

[10] Ryan Spangler, "Packet Sniffer Detection with AntiSniff". University of Wisconsin, Department of Computer and Network Administration, May 2003.

[11] Jorge Belenguer, Carlos T. Calafate, "A low-cost embedded IDS to monitor and prevent Man-in-the-Middle attacks on wired LAN environments", International Conference on Emerging Security Information, Systems and Technologies, IEEE, pp. 122-127, October 2007.

[12] DiabloHorn, Kimatrix, "ARP Poisoning In Practice"., 2008.

[13] Cristina L. Abad, Rafael I. Bonilla, "An Analysis on the Schemes for Detecting and Preventing ARP Cache Poisoning Attacks", 27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07), IEEE,June 2007.

[14] H. AbdelallahElhadj, H. M. Khelalfa, H. M. Kortebi, "An Experimental Sniffer Detector: Sniffer Wall", S´Ecurit´e des Communications sur Internet (SECI02), September 2002.

[15] Daniel Susid, "An evaluation of network based sniffer detection; Sentinel", School of Economics and Commercial Law, GÖTEBORG UNIVERSITY Department of Informatics, 2004.