A

Seminar report

On

Packet Sniffers

Submitted in partial fulfillment of the requirement for the award of degree
Of MCA

SUBMITTED TO:
www.studymafia.org

SUBMITTED BY:
www.studymafia.org

Preface

I have made this report file on the topic **Packet Sniffers**; I have tried my best to elucidate all the relevant detail to the topic to be included in the report. While in the beginning I have tried to give a general view about this topic.

My efforts and wholehearted co-corporation of each and everyone has ended on a successful note. I express my sincere gratitude to …………..who assisting me throughout the preparation of this topic. I thank him for providing me the reinforcement, confidence and most importantly the track for the topic whenever I needed it.

Acknowledgement

I would like to thank respected Mr…….. and Mr. ……..for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present a seminar report. It helped me a lot to realize of what we study for.

Secondly, I would like to thank my parents who patiently helped me as i went through my work and helped to modify and eliminate some of the irrelevant or un-necessary stuffs.

Thirdly, I would like to thank my friends who helped me to make my work more organized and well-stacked till the end.

Next, I would thank Microsoft for developing such a wonderful tool like MS Word. It helped my work a lot to remain error-free.

Last but clearly not the least, I would thank The Almighty for giving me strength to complete my report on time.

Table of Contents

1.0 Introduction..1

2.0 What is a packet sniffer? 2

3.0 Uses of a packet sniffer...3

4.0Sniffing tool…………………………………………………………………………………....4

5.0 Sniffing methods 5

5.1.1 IP-based sniffing 5

5.1.2 MAC-based sniffing 5

5.1.3 ARP-based sniffing...5

6.0 Anti sniff assumption...7

7.0 Anti-Sniff detection methods………………………………………………………………….7

7.1 Mac Detection……………………………………………………………………………..7

7.1.1 Ethernet Network Interface Cards………………………………………………………….8

7.1.2 TCP/IP on Ethernet ……………………………………………………………………….8

7.1.3 Implementation …………………………………………………………………………….8

7.1.4 Results ……………………………………………………………………………………..9

7.2 DNS Detection…………………………………………………………………………….10

7.2.1 Exploit Sniffer Behavior…………………………………………………………………..11

7.2.2 Implementation…………………………………………………………………………...12

7.2.3 Results …………………………………………………………………………………...13

8.0 Conclusion……………………………………………………………………………….....14

9.0 References……………………………………………………………………………….....15

1.0 Introduction

Packet sniffing is a technique of monitoring every packet that crosses the network. A packet sniffer is a piece of software or hardware that monitors all network traffic. This is unlike standard network hosts that only receive traffic sent specifically to them. The security threat presented by sniffers is their ability to capture all incoming and outgoing traffic, including clear-text passwords and user names or other sensitive material. In theory, it’s impossible to detect these sniffing tools because they are passive in nature, meaning that they only collect data. While they can be fully passive, some aren’t therefore they can be detected. This paper discusses the different packet sniffing methods and explains how Anti-Sniff tries to detect these sniffing programs.

2. Working of packet sniffer:

A packet sniffer works by looking at every packet sent in the network, including packets not intended for itself. This is accomplished in a variety of ways. These sniffing methods will be described below. Sniffers also work differently depending on the type of network they are in

Shared Ethernet:
In a shared Ethernet environment, all hosts are connected to the same bus and compete with one another for bandwidth. In such an environment packets meant for one machine are received by all the other machines. Thus, any machine in such an environment placed in promiscuous mode will be able to capture packets meant for other machines and can therefore listen to all the traffic on the network.

Switched Ethernet:
An Ethernet environment in which the hosts are connected to a switch instead of a hub is called a Switched Ethernet. The switch maintains a table keeping track of each computer's MAC address and delivers packets destined for a particular machine to the port on which that machine is connected. The switch is an intelligent device that sends packets to the destined computer only and does not broadcast to all the machines on the network, as in the previous case. This switched Ethernet environment was intended for better network performance, but as an added benefit, a machine in promiscuous mode will not work here. As a result of this, most network administrators assume that sniffers don't work in a Switched Environment. [2]

3. Uses of Packet Sniffers

Sniffing programs are found in two forms.

1) Commercial packet sniffers are used to help maintain networks.
2) Underground packet sniffers are used by attackers to gain unauthorized access to remote hosts.

Listed below are some common uses of sniffing programs:

- Searching for clear-text usernames and passwords from the network.
- Conversion of network traffic into human readable form.
- Network analysis to find bottlenecks.
- Network intrusion detection to monitor for attackers.

Using a sniffer in an illegitimate way is considered a passive attack. It does not directly interface or connect to any other systems on the network. However, the computer that the sniffer is installed on could have been compromised using an active attack. The passive nature of sniffers is what makes detecting them so difficult. The following list describes a few reasons why intruders are using sniffers on the network:

- Capturing clear-text usernames and passwords
- Compromising proprietary information
- Capturing and replaying Voice over IP telephone conversations
- Mapping a network
- Passive OS fingerprinting

Obviously, these are illegal uses of a sniffer, unless you are a penetration tester whose job it is to find these types of weaknesses and report them to an organization. For sniffing to occur, an intruder must first gain access to the communication cable of the systems that are of interest. This means being on the same shared network segment, or tapping into the cable somewhere between the paths of communications. If the intruder is not physically present at the target system or communications access point, there are still ways to sniff network traffic. These include:

- Breaking into a target computer and installing remotely controlled sniffing software.
- Breaking into a communications access point, such as an Internet Service Provider (ISP) and installing sniffing software.
- Locating/finding a system at the ISP that already has sniffing software installed.
- Using social engineering to gain physical access at an ISP to install a packet sniffer.

- Having an insider accomplice at the target computer organization or the ISP install the sniffer.
- Redirecting communications to take a path that includes the intruder's computer.

4. Sniffing Tools

- ***tcpdump:*** Tcpdump is a powerful tool that allows us to sniff network packets and make some statistical analysis out of those dumps. One major drawback to tcpdump is the size of the flat file containing the text output. But tcpdump allows us to precisely see all the traffic and enables us to create statistical monitoring scripts.[3]
- ***sniffit:*** Robust packet sniffer with good filtering. [3]
- ***Ethereal:*** A free network protocol analyzer for UNIX and Windows. It allows you to examine data from a live network or from a capture file on disk.[3]
- ***Hunt:*** The main goal of the HUNT project is to develop tools for exploiting well-known weaknesses in the TCP/IP protocol suite. [3]
- ***Dsniff:*** Dsniff is a collection of tools for network auditing and penetration testing. dsniff, filesnarf, mailsnarf, msgsnarf, urlsnarf, and webspy passively monitor a network for interesting data (passwords, e-mail, files, etc.). arpspoof, dnsspoof, and macof facilitate the interception of network traffic normally unavailable to an attacker (e.g, due to layer-2 switching). sshmitm and webmitm implement active monkey-in-the-middle attacks against redirected SSH and HTTPS sessions by exploiting weak bindings in ad-hoc PKI.
- IP spoofing : When the sniffing program is on a segment between two communicating end points, the intruder can impersonate one end in order to hijack the connection. This is often combined with a denial of service (DoS) attack against the forged address so they don't interfere anymore. [1]

5.1 Sniffing methods [4]

There are three types of sniffing methods. Some methods work in non-switched networks while others work in switched networks. The sniffing methods are: IP-based sniffing, MAC-based sniffing, and ARP-based sniffing.

5.1.1 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. This method only works in non-switched networks.

5.1.2 MAC-based sniffing

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

5.1.3 ARP-based sniffing

This method works a little different. It doesn't put the network card into promiscuous mode. This isn't necessary because ARP packets will be sent to us. This happens because the ARP protocol is stateless. Because of this, sniffing can be done on a switched network. To perform this kind of sniffing, you first have to poison the ARP cache1 of the two hosts that you want to sniff, identifying yourself as the other 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 us. We then log the traffic and forward it to the real intended host on the other side of the connection. This is called a man-in-the-middle attack. See Diagram 1 for a general idea of the way it works.

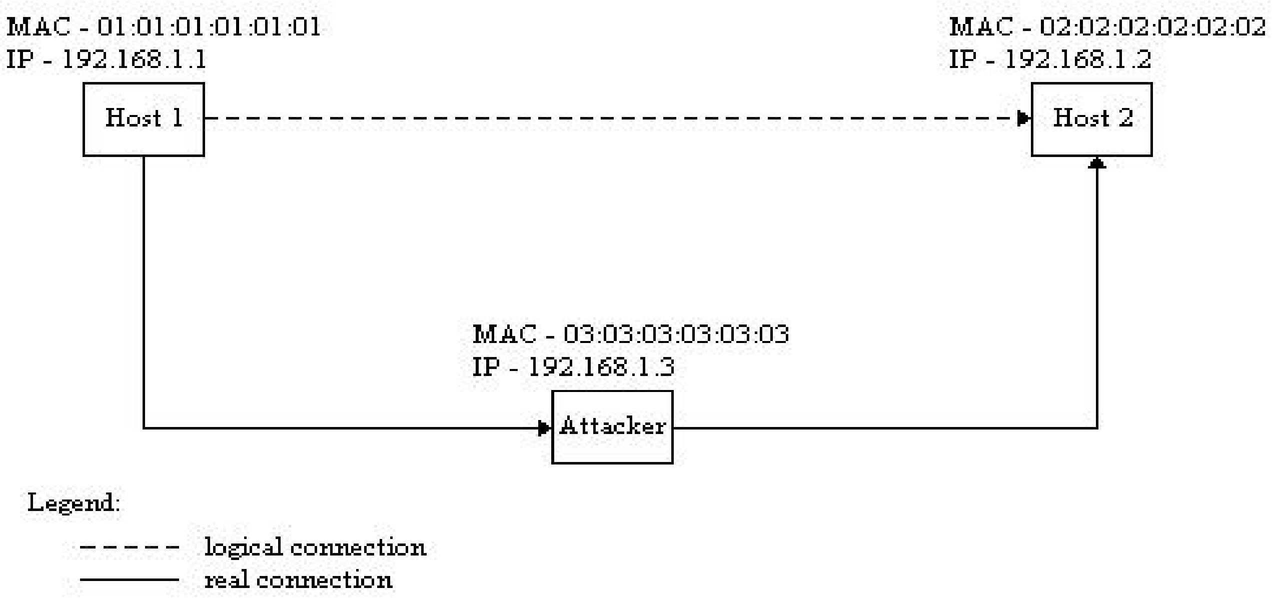


Diagram 1: ARP sniffing method

6 ANTI-SNIFF ASSUMPTIONS

We have made various assumptions when we developed our remote sniffer detector. These assumptions limit the types of sniffers that we can detect. However, we feel that our assumptions are valid and reasonable .One assumption we have made is that the sniffer is an actual sniffer program running on a host .That is, we disallow the possibility that the sniffer is a dedicated device that a hacker physically attaches to the network. This is a rather reasonable assumption since a lot of break-ins are done remotely by hackers with no physical access to the network whatsoever. Usually, a UNIX machine is broken in to , and the hacker logs on to the compromised machine and installs a sniffer with root access. Another assumption we have made is that the network segment that we are interested in, the network segment which we wish to detect whether a sniffer is running or not, is an Ethernet segment. Again, this is a reasonable assumption since a large percentage of the network segments on the Internet are Ethernet .This leads us to mention that we also assume that TCP/IP is the protocol that the network is using. Although some of our techniques can be modified to support other networking protocols, the implementation is based on TCP/IP since it is, by far, the most popular network protocol today.

7.0 Anti-Sniff detection methods :

7.1 MAC DETECTION

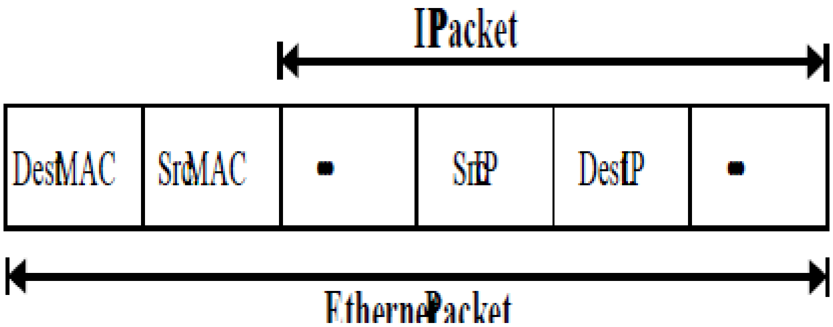
The MAC detection technique for detecting sniffers running on a Ethernet segment requires that the machine running the detector be on the same Ethernet segment as the host that is suspected of running a sniffer. Thus, this technique allows remote detection of sniffers on the same Ethernet segment, but not the remote detection of sniffers across different networks .The basic idea behind the MAC detection technique is simple and has been discussed in the past [6].

7.1.1 Ethernet Network Interface Cards:

A basic Ethernet network interface card has a unique medium access control (MAC) address assigned to it by its manufacturer. Thus, all network interface cards (NIC) can be uniquely identified by its MAC address. Since Ethernet is a shared medium network, all data packets are essentially broadcasted. Since passing all packets broadcasted on the network to the operating system is inefficient , Ethernet controller chips typically implement a filter which filters out any packet that does not contain a target MAC address for the NIC .Since sniffers are interested in all traffic on the Ethernet segment, NICs provide a promiscuous mode. In promiscuous mode, all Ethernet data packets, regardless of the target MAC address, are passed to the operating system. Thus, when a sniffer is running on a machine, the machine's NIC is set to promiscuous mode to capture all of the Ethernet traffic . Figure2 shows the flow diagram of the Ethernet data packet path to the operating system .

7.1.2 TCP/IP on Ethernet:

The Ethernet protocol standard, IEEE 802.3, specifies the Ethernet packet structure. Figure2 shows a IP packet encapsulated in a Ethernet packet. For TCP/IP, a normal IP packet destined to a particular Ethernet host has the destination host's MAC address filled in the Ethernet header and the IP address of the destination filled in the IP header. Thus, IP packets transported by Ethernet have two addresses, both of which normally correspond to a machine's MAC address and IP address [6].

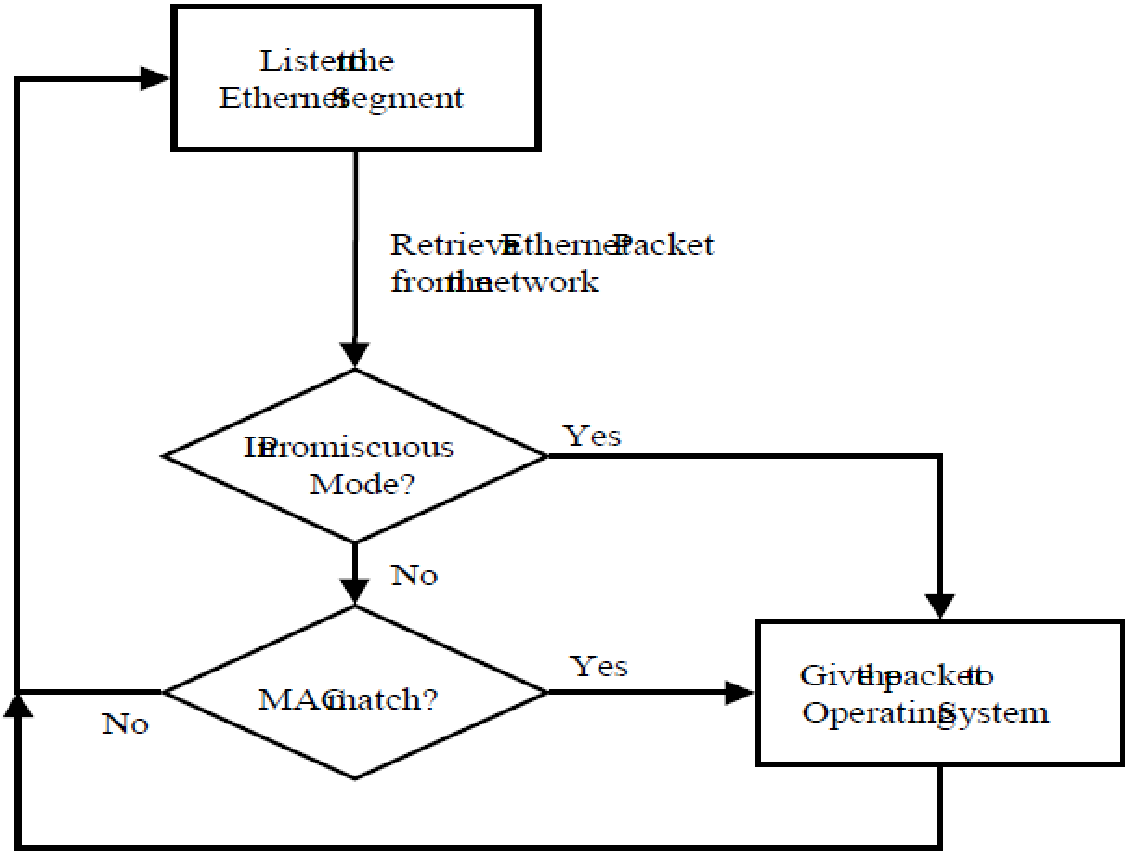


7.1.3 Implementation :

The implementation of the MAC detection technique is quite simple. The detection tool implements a ICMP Echo Request packet generator .The tool generates the full ICMP packet as well as the outer Ethernet packet that encapsulates the ICMP packet. The Ethernet packet is generate such that the target MAC address is different from the actual MAC address of the target machine. So, for any suspected host on the Ethernet segment, the tool can generate the ICMP Echo Request with incorrect MAC address and check if a ICMP Echo Reply is returned. If so, the suspected host is in promiscuous mode. Thus, a sniffer could likely be running on that host. Figure 3 shows how the MAC detection technique works as implemented.

7.1.4 Results :

The MAC detection technique works only against operating systems with a TCP/IP protocol stack that does not have the check against correct MAC addresses. We were able to confirm that Linux 2.0.35 was vulnerable to this kind of sniffer detection. We were able to detect when a Linux machine went in to promiscuous mode with 100% accuracy. However, FreeBSD 2.2.7 was not vulnerable to this kind of sniffer detection. The networking code in FreeBSD 2.2.7 correctly implements the necessary check so that incorrectly addressed Ethernet packets never reach the ICMP processing code.



Flow of Ethernet data packet with OS

8.0 DNS DETECTION:

The DNS detection technique exploits a behavior common in all password sniffers to date. This technique requires that the system administrator controls the Domain Name Server (DNS) [6]

8.1 Exploit Sniffer Behavior:

The DNS detection technique works by exploiting a behavior common to all password sniffers we have seen. The key observation is that all current password sniffers are not truly passive. In fact, password sniffers do generate network traffic, although it is usually hard to distinguish whether the generated network traffic was from the sniffer or not. It turns out that all password sniffers we have come across do a reverse DNS lookup on the traffic that it sniffed. Since this traffic is generated by the sniffer program, the trick is to detect this DNS lookup some how from normal DNS lookup requests. It is not hard to come up with the following idea. We can generate fake traffic to the Ethernet segment with a source address of some unused IP address that we provide the DNS service for. Then, since the traffic we generate should normally be ignored by the hosts on the segment, if a DNS lookup request is generated, we know that there is a sniffer on the Ethernet segment.

8.2 Implementation:

The implementation of the DNS detection technique is quite straight forward. The tool that implement this technique runs on the machine that is registered to provide the reverse DNS lookup for the trigger IP address, the invalid IP address that is used as the source address in the fake traffic. The tool generates a fake FTP [PR85] connection with the source IP address set to the trigger IP address. Then, the tool waits for a period of user definable time on the DNS service port. Within this period of time, the tool counts the number of DNS requests for the trigger IP address. When the time expires, the tool reports the number of DNS request counted. Note that the tool never returns a DNS reply. This is to avoid having the DNS entry being cached in some intermediate DNS server. The reason why DNS request needs to be counted is that the fake FTP traffic may actually be destined for a real machine on the network that provides FTP service. If so, that machine may trigger a DNS lookup. Thus, there are two cases we need to consider. If the fake FTP traffic ends up being destined to a real machine on the network, then if we count two or more DNS lookups, a sniffer is probably running on the network. Otherwise, if only one DNS lookup occurs, it is probably a legitimate lookup being performed by the host. The other case is that the fake traffic ends up being destined to no particular machine on the network. Then, if one or more DNS lookup occurs, there is most likely a sniffer on the network.

8.3 Results :

The DNS detection technique was able to detect sniffers running on a Ethernet segment with 100% accuracy regardless of operating system type. The default behavior of esniff, linsniff, sniffit and even tcpdump is to perform the reverse DNS lookup. Furthermore, it is possible to assign a trigger IP address to each network segment to perform the DNS detection technique .This is useful because even if the password sniffer does not perform a reverse DNS lookup, that is, the tool does not detect a sniffer in the required timeout period, the hacker may sometime in the future perform a reverse DNS lookup on the logged password entry. If so, then this technique can be extended to keep track of which IP address is assigned to what network and report a DNS lookup whenever it sees it in the future. request. Thus, the router will never generate the traffic on the network. However, this is possible to do if the machine running the tool is on the same network, therefore it can generate the fake traffic with invalid MAC addresses.

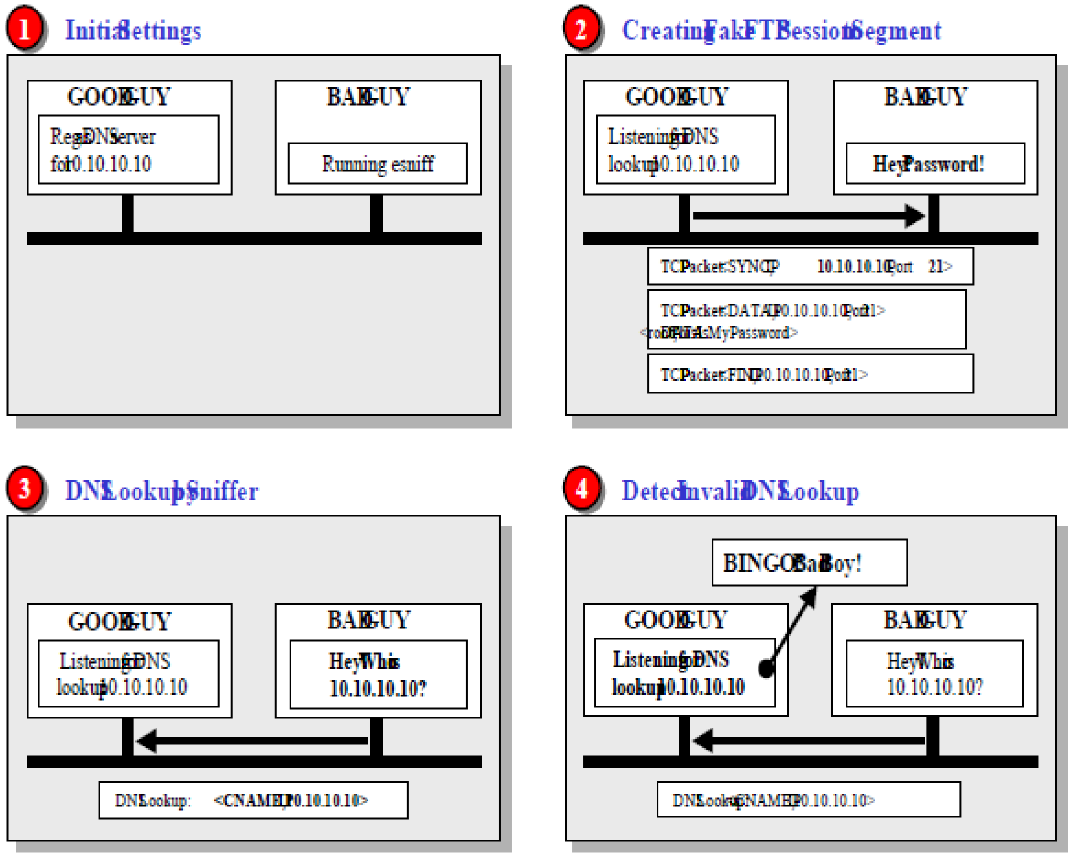

1
Initial Settings
GOOD GUY
Reg. DNS Server for 10.10.10.10
BAD GUY
Running esniff
2
Creating Fake FTP Session Segment
GOOD GUY
Listening for DNS lookup 10.10.10.10
BAD GUY
Hey Password!
TCP Packet: SYN <IP 10.10.10.10 Port 21>
TCP Packet: DATA <IP 10.10.10.10 Port 21>
<root> <PASS MyPassword>
TCP Packet: FIN <IP 10.10.10.10 Port 21>
3
DNS Lookup by Sniffer
GOOD GUY
Listening for DNS lookup 10.10.10.10
BAD GUY
Hey Who is 10.10.10.10?
DNS Lookup: <CNAME, IP 10.10.10.10>
4
Detect Invalid DNS Lookup
BINGO! Bad Boy!
GOOD GUY
Listening for DNS lookup 10.10.10.10
BAD GUY
Hey Who is 10.10.10.10?
DNS Lookup <CNAME, IP 10.10.10.10>

Diagram of DNS detection.

9.0 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 material. Because of this packet sniffers are a serious matter for network security. Fortunately, not all sniffers are fully passive. Since they aren’t tools like Anti-Sniff can detect them. Since sniffing is possible on non-switched and switched networks, it’s a good practice to encrypt your data communications.

References-

- www.google.com
- www.wikipedia.org
- www.studymafia.org
- www.projectsreports.org