International Journal Of Engineering Inventions
E-ISSN: 2278-7461, P-ISSN: 2319-6491
Volume 5, Issue 1 [Oct. 2015] PP: 57-60

Security Issues for Better Connection to Internet

[1]C Morris, [2]A Rama, [3]A srinivasulu
[1,2,3]Department of ECE,SreeDattha Institute of Engineering & Science

Abstract:- 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. The security threat presented by sniffers is their ability to capture all incoming and outgoing traffic, including clear-text passwords and usernames or other sensitive material. While packet sniffers can be fully passive, some aren't, therefore they can be detected. This paper discusses the different methods that AntiSniff uses to detect these sniffing programs.

I. 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 usernames 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 AntiSniff tries to detect these sniffing programs.

1.1 Packet Sniffer

A packet sniffer is a tool that plugs into a computer network and monitors all network traffic. It monitors traffic destined to itself as well as to all other hosts on the network. Packet sniffers can be run on both non-switched and switched networks.Uses of a packet sniffer Sniffing programs are found in two forms. Commercial packet sniffers are used to help maintain networks,whileunderground 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.

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. Here is a good set of definitions I found on the two types of Ethernet environments. This information was taken from an article on LinuxJournal.com by Sumit Dhar.

Shared Ethernet: In a shared Ethernet environment, all hosts are connected to the same busand compete with one another for bandwidth. In such an environment packets meant forone machine are received by all the other machines. Thus, any machine in such anenvironment placed in promiscuous mode will be able to capture packets meant for othermachines 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 sendspackets 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.

Sniffing methods
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.

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.

MAC-based sniffing

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

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.

II. ANTI SNIFF DETECTION METHODS

Anti Sniff is a tool made by L0pht Heavy Industries designed to detect hosts on an Ethernet/IP network segment that are promiscuously gathering data. The most current version (1.02.1) is designed to work on a non-switched network. AntiSniff performs different types of tests to determine whether a host is in promiscuous mode. The tests are broken down into the following three classes: DNS tests, operating system specific tests, and network and machine latency tests. These tests are described below.

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. If any name lookups from the bogus hosts are seen, a sniffer might be in action on the host performing the lookups.

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 Net BSD kernels. Each test will be described below.

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. 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. 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.

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. 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. To get a response, an ICMP echo request message is sent within the bogus packet leading to vulnerable

hosts in promiscuous mode to respond.

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. The following tests will be explained: ICMP Time Delta test, Echo test, and the Ping Drop test.

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. Hosts in promiscuous mode have a much higher latency time.

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.

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. 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.

III. 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 AntiSniff can detect them. Since sniffing is possible on non-switched and switched networks, it's a good practice to encrypt your data communications.

REFERENCES

[1] D. Geer, "Is it time for clockless chips? [Asynchronous processor chips]," IEEE Comput., vol. 38, no. 3, pp. 18–19, Mar. 2005.

[2] J. Sparsø and S. Furber, Principles of Asynchronous Circuit Design.

[3] Boston, MA, USA: Kluwer Academic, 2001.

[4] P. Choudhury, S. Sahoo, and M. Chakraborty, "Implementation of basic arithmetic operations using cellular automaton," in Proc. ICIT, 2008, pp. 79–80.

[5] M. Z. Rahman and L. Kleeman, "A delay matched approach for the design of asynchronous sequential circuits," Dept. Comput. Syst. Technol., Univ. Malaya, Kuala Lumpur, Malaysia, Tech. Rep. 05042013, 2013.

[6] M. D. Riedel, "Cyclic combinational circuits," Ph.D. dissertation, Dept. Comput. Sci., California Inst. Technol., Pasadena, CA, USA, May 2004.

[7] R. F. Tinder, Asynchronous Sequential Machine Design and Analy- sis: A Comprehensive Development of the Design and Analysis of Clock-Independent State Machines and Systems. San Mateo, CA, USA: Morgan, 2009.

[8] W. Liu, C. T. Gray, D. Fan, and W. J. Farlow, "A 250-MHz wave pipelined adder in 2-μm CMOS," IEEE J. Solid-State Circuits, vol. 29, no. 9, pp. 1117–1128, Sep. 1994.

[9] F.-C. Cheng, S. H. Unger, and M. Theobald, "Self-timed carry-lookahead adders," IEEE Trans. Comput., vol. 49, no. 7, pp. 659–672, Jul. 2000.

[10] S. Nowick, "Design of a low-latency asynchronous adder using spec- ulative completion," IEE Proc. Comput. Digital Tech., vol. 143, no. 5, pp. 301–307, Sep. 1996.

[11] N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective. Reading, MA, USA: Addison-Wesley, 2005.

[12] C. Cornelius, S. Koppe, and D. Timmermann, "Dynamic circuit tech- niques in deep submicron technologies: Domino logic reconsidered," in Proc. IEEE ICICDT, Feb. 2006, pp. 1–4.

[13] M. Anis, S. Member, M. Allam, and M. Elmasry, "Impact of technology scaling on CMOS logic styles," IEEE Trans. Circuits Syst., Analog Digital Signal Process., vol. 49, no. 8, pp. 577–588, Aug. 2002.