CITS 3004
Cybersecurity

Lab 6: Packet Sniffing and Spoofing

Packet sniffing and spoofing are the two important concepts in network security; they are two major threats in network communication. Being able to understand these two threats is essential for understanding security measures in networking. There are many packet sniffing and spoofing tools, such as `Wireshark`, `Tcpdump`, `Netwox`, etc. Some of these tools are widely used by security experts, as well as by attackers. Being able to use these tools is important for students, but what is more important for students in a network security course is to understand how these tools work, i.e., how packet sniffing and spoofing are implemented in software.

The objective of this lab is for students to master the technologies underlying most of the sniffing and spoofing tools. Students will play with some simple sniffer and spoofing programs, read their source code, modify them, and eventually gain an in-depth understanding on the technical aspects of these programs. At the end of this lab, students should be able to write their own sniffing and spoofing programs.

Task 1: Sniffing

Sniffer programs can be easily written using the `pcap` library. With `pcap`, the task of sniffers becomes invoking a simple sequence of procedures in the `pcap` library. At the end of the sequence, packets will be put in buffer for further processing as soon as they are captured. All the details of packet capturing are handled by the `pcap` library. Tim Carstens has written a tutorial on how to use `pcap` library to write a sniffer program. The tutorial is available at [1]. In this task, you need to read the tutorial, play with the program `sniffex` included in the tutorial, read the source code `sniffex.c`, and carry out the following subtasks.

To compile the `sniffex.c`, use the following command.

```
# gcc -o sniffex sniffex.c -lpcap
```

CITS 3004 Cybersecurity

THE UNIVERSITY OF WESTERN AUSTRALIA

1A) SNIFFING PACKETS

Write filter expressions to capture each of the followings.

1. Capture the ICMP packets between two specific hosts. (your IP address can be found by using a command `ifconfig`, and you can test the target host with google). You can sniff ICMP packets by using ping, and you can try telnet (these are TCP packets) to google to see if they can be sniffed.
2. Capture the TCP packets that have a destination port range from to port 10 - 100. Again, you can test this using telnet and ping.

1B) SNIFFING PASSWORDS

We will now use `sniffex` to capture the password when somebody is using `telnet` on the network that you are monitoring. You may need to modify the `sniffex.c` a little bit if needed.

If you are using our pre-built VM, the `telnetd` server is already installed; just type the command `telnet [address]` to start it.

If you are not using the provided VM, then you may also need to start the `telnetd` server on your VM.

```
# sudo service openbsd-inetd start
```

You can try connecting to the telnet server `scn.org` to demonstrate this exercise.

Answer Q1 ~ Q5 on LMS

Task 2: Spoofing

When a normal user sends out a packet, operating systems usually do not allow the user to set all the fields in the protocol headers (such as TCP, UDP, and IP headers). OSes will set most of the fields, while only allowing users to set a few fields, such as the destination IP address, and the destination port number, etc. However, if the user has the root privilege, he/she can set any arbitrary field in the packet headers. This is essentially packet spoofing, and it can be done through *raw sockets*.

Raw sockets give programmers the absolute control over the packet construction, allowing programmers to construct any arbitrary packet, including setting the header fields and the payload. Using raw sockets is quite straightforward; it involves four steps: (1) create a raw socket, (2) set socket option, (3) construct the packet, and (4) send out the packet through the raw socket. As an example, rawtcp.c and rawudp.c programs are provided, which generates raw TCP and UDP packets. For example, you can generate TCP packets as follows.

```
# rawtcp [source IP] [source port] [dest IP] [dest port]
```

You have to provide the source IP, source port, destination IP and destination port.

There are many online tutorials that can teach you how to use raw sockets in C programming. In this task, we will not focus on any specific tutorial. Your task is to read some of these tutorials. You can start reading from [2] (if you want explained tutorial try section 4 of [3]).

2A) SNIFFING AND SPOOFING

The `sniffspoof.c` code is provided for you, which sniffs the ICMP packets, and generate a spoofed ICMP reply packets even if the target host is not valid. You can try pinging an invalid target host (e.g., 1.2.3.4) to see what replies you get when the `sniffspoof` is running and when it is not running.

2B) SPOOFING ICMP PACKETS

Complete the partially written `icmp.c` code which is provided for you. Once done, test that you can generated spoofed ICMP packets.

Answer Q6 ~ Q9 on LMS

References

1. http://www.tcpdump.org/pcap.htm
2. http://www.tenouk.com/Module43a.html
3. http://www.enderunix.org/docs/en/rawipspoof/