HPY 413 Assignment: 5

Network traffic monitoring using the Packet Capture library

Dates: 26/11/2024 - 06/12/2024

In this assignment, you will create a networking monitoring tool based on the C programming language. You are required to utilize the packet capture library (libpcap), where this library allows you to to capture the packets from the network interface of the executed device.
More specifically:

1. Designed monitoring tool required to process traffic in two modes
    a. Online: monitor the traffic live from a network interface (pcap_open_live)
    b. Offline: read a pcap file (pcap_open_offline).
2. In the Online mode you should capture network traffic and process the TCP and UDP packets.
    a. Do not use pcap_compile or pcap_setfilter.
    b. You will need sudo permission in order to capture the real time traffic.
3. For the Offline mode you will download the pcap file of real IoT attack also known as mirai (https://github.com/ymirsky/Kitsune-py/raw/refs/heads/master/mirai.zip).
4. You should execute your tool for online and offline mode and provide the terminal output of your execution as **online_output.txt** and **offline_output.txt**.
5. You can find more information about the libpcap: https://linux.die.net/man/3/pcap and https://www.tcpdump.org

Implementation

During this assignment you are required to implement the following:

1. Select one interface that you wish to monitor or select the pcap file name.
2. Start capturing/reading packets.
3. Apply any user provided filters.
4. Decode each received packet (i.e., is it a TCP or UDP packet?)[1]
5. Skip any packet that is not TCP or UDP.
6. Print the packet's source and destination
    a. IP addresses.
    b. port numbers.
7. Print the packet's protocol.
8. Print the packet's TCP/UDP header length and TCP/UDP payload length in bytes.
9. Find where the payload is in memory.
10. In your program (when possible), mark each retransmitted packet as "Retransmitted".
11. On exit, your program must print the following statistics:

[1] Support both IPv4 and IPv6 packets.

a. Total number of network flows captured.[2]
b. Number of TCP network flows captured.
c. Number of UDP network flows captured.
d. Total number of packets received (include the packets you skipped, that weren't TCP or UDP packets.).
e. Total number of TCP packets received.
f. Total number of UDP packets received.
g. Total bytes of TCP packets received.
h. Total bytes of UDP packets received.

Theoretical questions

1. Can you tell if an incoming TCP packet is a retransmission? If yes, how? If not, why?
2. Can you tell if an incoming UDP packet is a retransmission? If yes, how? If not, why?

Command Line Parameters

The developed tool should also support command line arguments and switch the execution functionality based on the parsed arguments. The options that should be supported are:

-i	Select the network interface name (e.g., eth0)
-r	Packet capture file name (e.g., test.pcap)
-f	Filter expression in string format (e.g., port 8080)
-h	Help message, which show the usage of each parameter

Examples of the execution:

- ./pcap_ex -i eth0 (save the packets in log.txt)
- ./pcap_ex -r mirai.pcap (print the outputs in terminal)
- ./pcap_ex -i eth0 -f "port 8080"

Additional information

1. The options defined in the "Tool specification" section must remain as-is.
2. If no appropriate option was given, your program has to print the appropriate error message.
3. You need to create a **Makefile** to compile your library and programs (you must submit it with your source code).
4. You are provided with a sample packet capture to test your program. Its duration is 5 minutes.

[2] A network flow is defined by the 5-tuple {source IP address, source port, destination IP address, destination port, protocol}.

5. You need to create a **README** with your name, your AM and a short description of your implementation with answers to theory questions.
6. You must submit the following files: **README**, **Makefile**, **pcap_ex.c**.
7. You should write the outputs of the execution (with -i) in a log file and the outputs of the execution (with -r) appear in the terminal (**online_output.txt** and **offline_output.txt**).
8. You should place all these files in a folder named <AM>_assign5 and then compress it as a .zip file. For example, if your login is 2020123456 the folder should be named 2020123456_assign5 you should submit 2020123456_assign5.zip.
9. **GPT generated code is banned, and submission with GPT created code would be rejected.**