CMPSC335: Fundamentals of Communication Networks

Lab 11 (8 points)

Objectives

The following concepts and skills are demonstrated through this process:

1. Analyzing network packets with Wireshark.
2. Practicing C/C++ programming by utilizing static library to analyzing binary files.
3. Understand TCP Packet format.

Lab Activities

Please complete all lab activities (1 – 2) and submit your Lab report to Canvas per the submission instructions given at the end of this document.

Lab Activity 1 (3 points)

In this activity, you will use **Wireshark** to analyze network packets. These packets are captured and stored in a pcapng file (*tcp.pcapng*). Please analyze this file and answer the following questions.

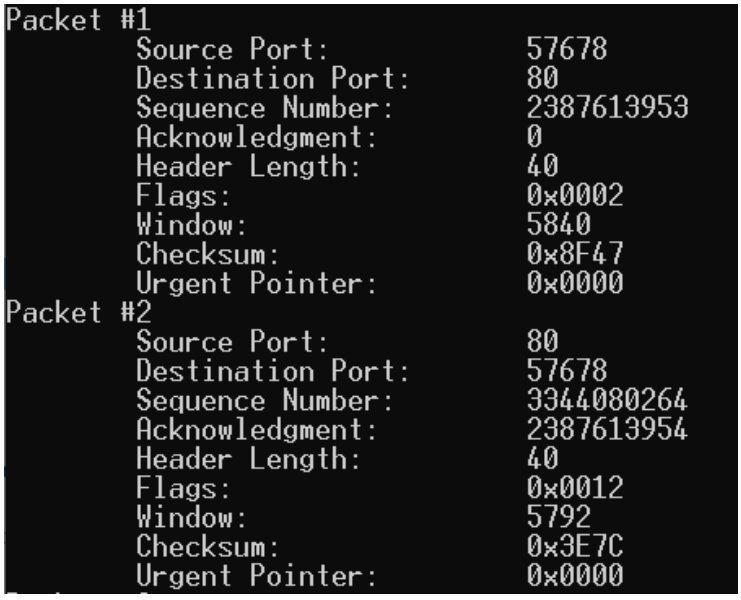
Lab Activity 1 – Questions

Please include answers to the following questions in your Lab report:

1. How many packets are stored in this file?
2. Let's assume the client's IP address is 192.168.1.140 and the server's IP address is 174.143.213.184. What are the ports used by the client and the server?
3. What is the *initial* sequence number used by the client and the server (Use the *raw* number)?
4. When this TCP connection is terminated, what is the last sequence number used by the client and the server (Use the *raw* number)?

Lab Activity 2 (5 points)

In this activity, you will complete one function (*ParseTCPPackets*), which parses the pcapng file (*tcp.pcapng*) and displays the packet information like the format shown in the figure below.



Notice that, you only need to output the details of TCP packet header, such as what is shown for Packet #1. Except the Flags, the Checksum, and the Urgent Pointer, the others use the decimal format. Notice the header length, which uses the byte as the unit. So, you need to multiply four with the number read from the packet.

Please use the cpp file (***Lab11.cpp***) as the start point. It is posted on Canvas. This program uses a static library, which can facilitate our analysis.

- Download this file (***linux.zip***) and unzip it. Copy all files into the same directory in Kali Linux. Read the "*ReadMe.txt*" file for how to use these files.

****Hints** - When using the function *light_get_next_packet*, please notice two variables with the types *light_packet_header* and *uint8_t*.

This function *light_get_next_packet* has the following signature:

```
int light_get_next_packet (light_pcapng_t *pcapng,
                           light_packet_header *packet_header,
                           const uint8_t **packet_data);
```

The type *light_packet_header* has the following detinition:

```
typedef struct _light_packet_header {
        uint32_t interface_id;
        struct timeval timestamp;
        uint32_t captured_length;
        uint32_t original_length;
        uint16_t data_link;
        char* comment;
        uint16_t comment_length;
} light_packet_header;
```

Lab Activity 2 – Questions

Please include answers to the following questions in your Lab report.

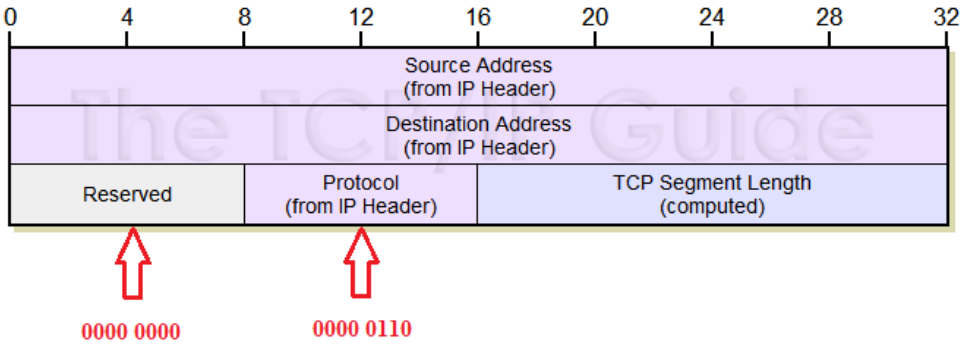
1. Include a screenshot of the console output result of Lab11.cpp, which outputs the information like the figure above. Because there are many packets in this pcapng file, the screenshot only needs to include the **last three packets**.
2. Include the completed ***Lab11.cpp*** file.

Lab Activity 3 (2 points) (**Bonus Activity**)

In this activity, you will continue the previous activity and **verify** the checksum in the **TCP** packet header. The sample output is: (The added text is highlighted with yellow color.)

```
Packet #1
        Source Port:            57678
        Destination Port:       80
        Sequence Number:        2387613953
        Acknowledgment:         0
        Header Length:          40
        Flags:                  0x0002
        Window:                 5840
        Checksum:               0x8F47
        Urgent Pointer:         0x0000
        Checksum verification succeeds!
Packet #2
        Source Port:            80
        Destination Port:       57678
        Sequence Number:        3344080264
        Acknowledgment:         2387613954
        Header Length:          40
        Flags:                  0x0012
        Window:                 5792
        Checksum:               0x3E7C
        Urgent Pointer:         0x0000
        Checksum verification succeeds!
```

Notice that, when verifying the checksum in the TCP packet header, recap how the checksum is computed. The pseudo header is:



The TCP segment length may be odd. So, the last octet is padded on the right with zeros to form a 16-bit for checksum purposes.

The previous class video and labs, discussing checksum verification, can be helpful and some code can be reused.

Lab Activity 3 - Questions

Please include answers to the following questions in your Lab report.

1. Include a screenshot of the console output result, which contains the checksum verification result. Because there are many packets in this pcapng file, the screenshot only needs to include Packet #1 - #5 and Packet #35 - #40.
2. Rename the ***Lab11.cpp*** to ***Lab11Bonus.cpp*** and upload this file.

Submission

There is a MS-Word Lab report template on Canvas that you can download as a starting point for your Lab submission. There are two sections for you to fill in. Each section corresponds to the two Lab Activities for this Lab. For each section, please give a brief summary of what you did – feel free to include any thoughts / concerns / problems / etc. you encountered during the activities. Also, include your answers to the questions asked in each Lab Activity. Save your report as a PDF and submit it to Canvas before the deadline.