Programming Project

Offline Network Protocol Analyzer

Overview

The objective of this project is to write an offline network protocol analyzer. Your program can either run on the command line or be supported by a graphical interface. The list of protocols that your analyzer will need to process is the following:

- Layer 2: Ethernet
- Layer 3: IP
- Layer 4: TCP, UDP
- Layer 7: HTTP (header only of requests and replies)

The output should be saved in a text file with basic formatting for better readability.

Grading

- This project is to be conducted **individually**.
- Submission **deadline** is **August 07, 2020 11:59PM (Shanghai time)**.
- Submission **material** consists of:
    1. An **archive** to be submitted in the Assignment section of **NYU Classes** including:
        a. all your **source code**,
        b. a **binary file** or **makefile** to launch your analyzer,
        c. a **readme** file describing your code and the design decisions you made,
        d. a **howto** file explaining how to install and use your analyzer.
    2. A **10-minute presentation** on **VoiceThread**.The VoiceThread links will be posted on NYU Classes. In the VoiceThread, you will present:
        1. an **overview** of your project,
        2. a description of **what you have achieved**,
        3. a **demonstration** of the software you have written.
- You are free to use any programming language or environment.

High-level Guidelines

1/ Input

The input to your program is a text file containing the hex dumb of multiples Ethernet frames (without preamble and FCS fields):

- Each byte is encoded as two hex characters.
- Each byte is surrounded by a space.
- Each line begins with an offset encoded on at least 1 byte (2 hex characters) describing the position of the first byte on the line in the frame.
- Each new frame starts with an offset of 0 and there is a space separating the offset from the following bytes.
- The offset is a hex number of at least two hex characters.
- All hex characters can be uppercase or lowercase.
- There is no limit on the length or the number of bytes per line.
- Any text dump at the end of the line should be ignored. Any hex numbers in this text should also be ignored.
- Lines of text between the bytestring lines should be ignored.
- Any line without a leading offset should be ignored.
- Any incomplete line should raise an error message specifying the corrupted line.

2/ Output

The output of your program should be similar to Wireshark's 'Packet Details Pane'. The output should consist of the protocol headers listed in the same order as in each frame hexdump contained the input text file.

- For each protocol header, the output will list the protocol fields and the value of these fields in hexadecimal and converted in decimal when needed,

  Example: IP Total Length field: 0x05C8 (1500 bytes).

- For protocol fields containing a code, your program will provide the meaning of the code,

  Example: Ethernet Type field: 0x0806 (ARP)

- The protocols and fields of the packet can be shown in a tree which can be expanded and collapsed or at least using indentations to distinguish the fields belonging to different protocol headers.
- The output should be saved in a text file with basic formatting for better readability.