ECE 5600 Project (Phase 1)

Project Due Date: Sep 25, 2017

OBJECTIVE

1. Be familiar with Wireshark and Linux OS.

2. Know how to capture traffic and C programming under Linux OS environment.
3. Know how to customize Wireshark filter to capture and analyze the data frames.

BACKGROUND

During the course of this semester, we will implement some of the TCP/IP protocol stack using Linux. Linux allows us to configure a socket in packet mode so that we can send and receive raw packets over the Ethernet. We will be able to build up our own network stack without interfering with the normal operation of the network connection. In order to avoid chaos, each student needs an IP address, while has the following format:

192.168.1. xxx

You can use the command "ifconfig" to get your IP address.

PRE-LAB READING

Chapter 4 P 281-286, P 465-469

PROJECT PROCEDURE

The following steps outline the first phase of this project:

1. Get access to the Networking Lab (EL103). Go see Heidi at the ECE Student Store and ask for the access card. Once you've filled out the necessary paperwork, she'll give you the access and you will need to pick up the card from the key office.

2. Log on to one of the computers in the lab. You may log in as root and use **ece5600** as the password. Be careful, as super-user on these machines you can really mess them up. Create a new folder under /home/students/, the folder's name can be your A number.
3. Please make sure you put all your documents in this folder and take all your codes and data with you after each lab session, since they are public computers and may be shared by other students from other classes.

4. Get the C++ code for Frame I/O from the course website. Read carefully frameio.h and frameio.cpp. Understand them thoroughly because they are very useful.
5. Now, read the example code (example1.cpp). This example basically gives you a hint on how to use existing codes. Write a program to read an incoming packet (either IP packet or ARP packet) and print the first 42 bytes, each in "%02x" format, to the standard output. For clarity, put a space between each byte and add a new-line character ('\n') after the 22nd and 42nd bytes. 4. Run your program and verify that it monitors network traffic (Network traffic in an isolated network will be light, so if you don't see any traffic, go to a different machine and ping something.)
6. Run the program Wireshark (**Start_Applications_System_Monitor_Network Analyzer**). This is a sophisticated program that monitors network traffic. You can get familiar with this software by reading the user's guide carefully.
7. Start capturing traffic (the device to monitor is probably "eth0", you should check it before the experiment). From another machine, ping your IP address (given above). You will observe the packets transmission over the network. There are many unrelated frames captured in Wireshark, you can define the capture filter in **Capture Options**.

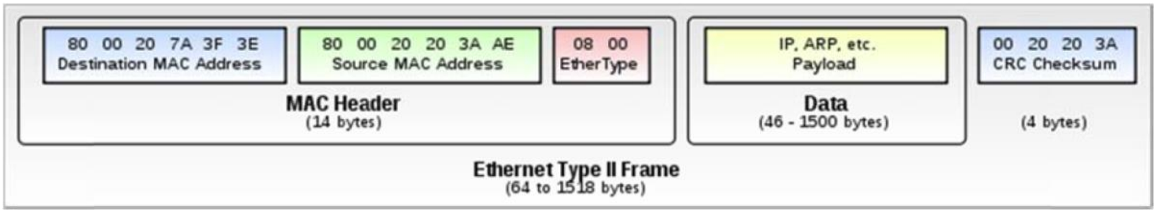
8. Stop capturing traffic and use CTRL+C to terminate your program. Use Wireshark to find the ARP packets containing your IP address. Select that packet, select the raw data in the bottom window and paste it into a file (usually by pressing the middle mouse button or wheel, or by pressing both left and right buttons at the same time for a 2-button mouse).
9. 8 Locate the corresponding packet in your program's output. Select both lines and paste them into your report as well.
10. Take all your programs and data with you on your memory stick. Leave no trace behind.

REPORT REQUIREMENTS

1. Read carefully about the lab report requirements from the syllabus.
2. Screenshot the Wireshark window (with detailed output text in the bottom window) and your command window's output after you successfully capture the frames and compare the two frames.
3. Attach your codes as an appendix in the end of your report.
4. Be sure to add your partner's name in the report.
5. Online submissions only, please archive all your files in .zip or .rar format.

Hints for Project 1

In order to understand project 1, you need to have some knowledge on Ethernet frame. It looks like the following (Preamble is omitted since it's used for synchronization):



The MAC header contains three parts: Destination MAC address, Source MAC address and frame type, with a total length of 14 bytes. Here, frame type (Ether Type) indicates the payload data type. Two common types are 0x0800 (IP) and 0x0806 (ARP).

You need to output MAC header (14 bytes) and the first 28 bytes from the payload in project

A good practice is to run the example1.cpp first and here is the brief procedure:

1) Find the network interface name, by using command "ifconfig" in your computer.
2) Replace the name in example1.cpp, the old name is "eth1" in the main function.
3) Compile the code. Run "g++ example1.cpp frameio2.cpp util.cpp -lpthread -o out" from the terminal.
4) Run the code. Using "./out" in the terminal.

You should see that this example code captures IP and ARP frames.

(If there's no traffic information, make sure you changed the interface name, if still no output, ping your machine from other computers.)

Now, for your own program, you can borrow some ideas from the example code.

1. Define a frameio structure.
2. Open the interface.
3. Using a while loop to receive frames. (if the frame length is less than 42, discard it)

4. Print the first 42 bytes, with a space between each byte and add a new line after 22 bytes for extra clarity.