Project 1: Parsing, filtering and generating network traffic

CS 7473 Network Security

Noah M. Jorgenson

This project was probably the most involved project I have done related to computer networks. The amount of information is overwhelming initially. In order to read the different varieties of packets I employed trial and error techniques and also only documentation from RFCs and Wikipedia pages. With that information I used a very simple class which made a new PacketHandler class which extended Runnable and read in a packet and filtered information about it. With some simple hardcoded values and if statements I was able to filter and get some basic information for all the packet types, Ethernet, IP, ARP, TCP, UDP and ICMP. My packet class hierarchy starts with a GenericPacket class with two variables which is extended by an EthernetPacket class. The EthernetPacket class contains the MAC addresses, EtherType, etc. The EthernetPacket is then extended by either an IPPacket or ARPPacket with their respective values, and finally the IPPacket extended by an ICMPPacket, TCPPacket or UDPPacket, again with the appropriate values for the given packet type.

After I had the general architecture set up I then built my command line argument parsing switch in the main method. This is where I think my code may have gotten sloppy, but not wanting to prematurely optimizing I just pressed on, I will go back and look for opportunities to improve my code for the second project, which I assume will build upon this code. Anyway, the count and input/output arguments were straightforward to program. The filtering however, based on packet type, addresses and ports, was where I had to stop and consider the avenues I could take. I ultimately chose to implement methods in the packet classes which filtered based on passed parameters. So in the PacketParser I determine if the values are not null, then I am filtering based on them, so I will call the respective filter method on the packet type. With this setup I think it will be easy to extend the functionality while maintaining the readability of my code.

While my parser does output the essential information, it is in a line format and not in the suggested readable blocks in the project handout. I can easily update this portion of my code, but I chose to put it off since it is something aesthetic, I wanted to focus on the functionality of my parser after the output was readable.

My packet generator lives in a class named PacketGenerator, it’s very simple, based on the PacketParser with just a single argument pointing to the filename of the input.

Network Questions

1. Telnet Session to 192.168.1.22
   Client IP: 192.168.1.66
   Username: group15
   Password: 192.168.1.66

2. FTP Logon Session to 192.168.1.66
   Client IP: 192.168.1.62

3. File Transfer to 192.168.1.42
   Client IP: 192.168.1.62
   Username: group14
   Password: 192.168.1.62
   Filename: FTP-GROUP14.NFO

4. HTML File Transfer by 192.168.1.10
   Client IP: 192.168.1.22
   Filename: /cs7493/
   Content type: text/html

5. IP Addresses Returned by DNS Servers (192.168.1.14 and 192.168.1.46)
   Iodine: 192.168.1.62
   Hydrogen: 192.168.1.10

6. ARP Request to 192.168.1.200 Reply
   MAC Address: 00:c0:4f:58:d9:94