Project 1

Samuel Bhushan, Partner: Hailey Parkin

September 25, 2015

1 Introduction

Project one exposed us to the wireshark tool and some C++ networking libraries. Wireshark allows us to see network traffic and filters are used to capture only some packets. In this lab we made a basic program in C++, using given libraries, that read in packets and printed them out. Wireshark was used to show that our program was functioning properly.

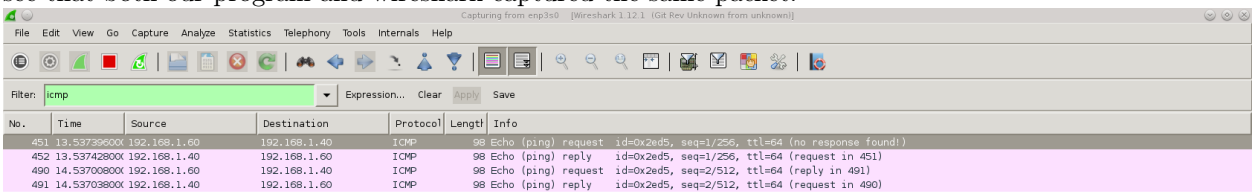
2 The code

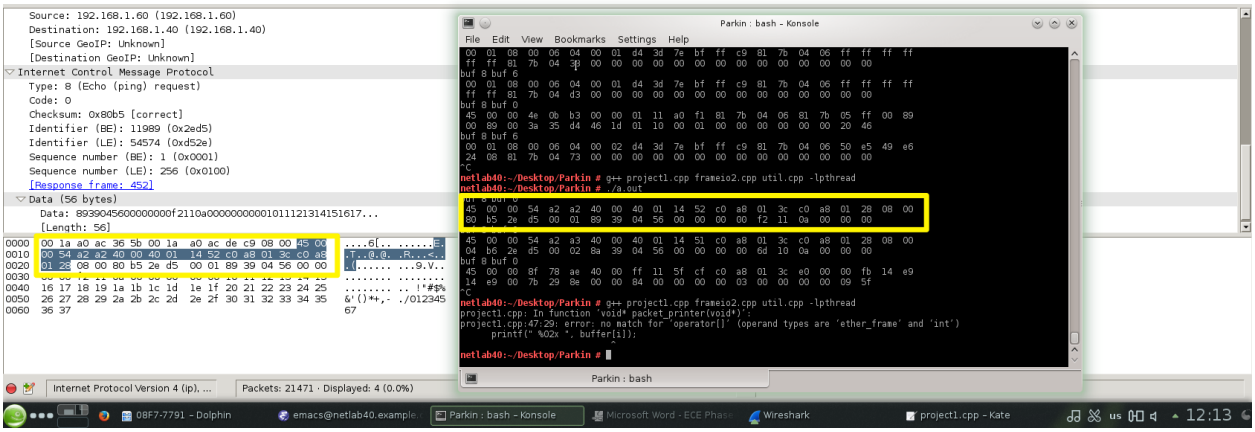
Gathering ideas from the example code, our group made a program with a thread that continually receives packets from the network. It limits which packets to receive by checking the source mac address and the protocol type. We only wanted icmp packets from a specific computer. Our code is listed below:

```
#include "frameio.h"
#include "util.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>

frameio net;              // gives us access to the raw network
message_queue ip_queue;   // message queue for the IP protocol stack
message_queue arp_queue; // message queue for the ARP protocol stack

struct ether_frame        // handy template for 802.3/DIX frames
{
   octet dst_mac[6];      // destination MAC address
   octet src_mac[6];      // source MAC address
   octet prot[2];         // protocol (or length)
   octet data[1500];      // payload
};

void* packet_printer(void*);

pthread_t packet;

int main()
{
        net.open_net("enp3s0");
        pthread_create(&packet,NULL,packet_printer,NULL);
        for( ; ;) sleep(1);
}

void* packet_printer(void *arg)
{
        ether_frame buffer;

        while(1)
        {
           int n = net.recv_frame(&buffer,sizeof(buffer));
           if ( n < 42 ) continue; // bad frame!
           if((buffer.src_mac[5] == 201) && (buffer.prot[1] == 0))
             {
               printf("buf %d ",buffer.prot[0]);
               printf("buf %d \n",buffer.prot[1]);
               for(int i=0;i<42;i++)
                 {
                   //printf("i%d ",i);
                   printf(" %02x ", buffer.data[i]);
                   if(i== 21 || i==41)
                     printf(" \n");
                 }
             }
        }
}
```

3 Comparing Received Packets with wireshark

We sent ping requests from a computer with a mac address ending in 0xC9 (0d201). In the screenshot below, we see that both our program and wireshark captured the same packet:





4 Conclusions

Wireshark is quite a useful program, especially for troubleshooting network problems. It helped us realize that our packets were failing to send because a lan cable was unplugged. I have also used wireshark in a security class where we were able to hijack a connection between two computers. The provided libraries worked flawlessley and were easy to implement.