ICMP Blind Connection-Reset + Blind throughput reduction attack against TCP

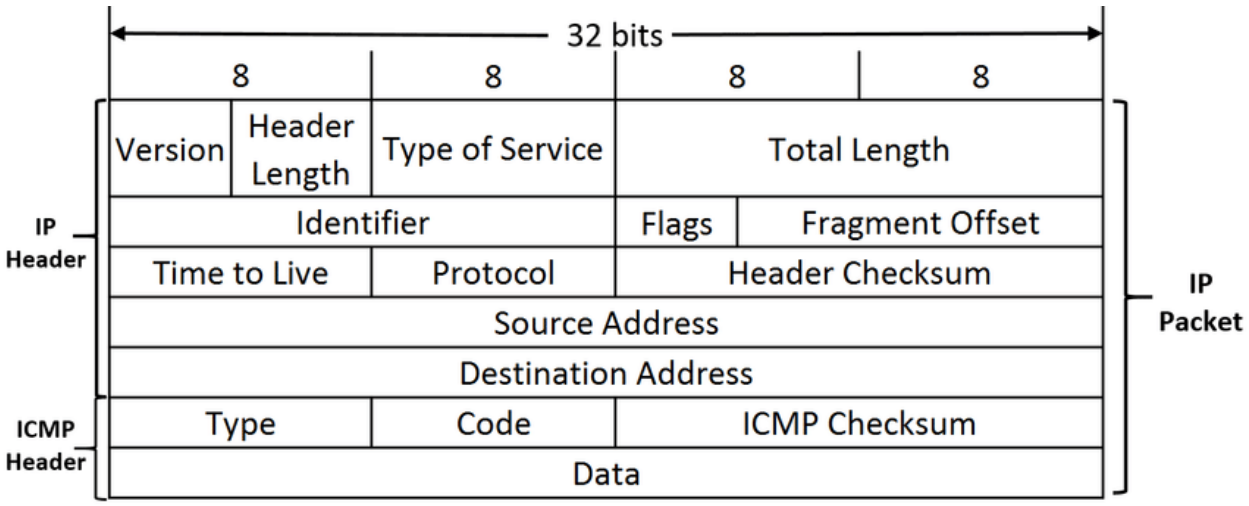
Md. Tahmidur Rafid
Student ID: 1605046

Steps of attack:

1. **Setting up three VMs:** Three virtual machines were set up for this attack. The machines we used are:
    a. Attacker Machine (IP: 192.168.0.110)
    b. Server Machine (IP: 192.168.0.106)
    c. Client Machine (IP: 192.168.0.109).

2. **Setting up Server and Client:** A Server was set up on the server machine and a client was set up on the client machine. Code for both server and client were written in java. The Server was operating on port 1105. The server was continuously printing the messages received from the client side. Our goal was to reset the established TCP connection between client and server.

3. **Creating raw ICMP packet:** For this attack we have to send ICMP hard error message, so we had to design our own ICMP packet.



we created raw packet following the packet diagram. In the source address field, we set the IP address of the client and IP address of the server was put in the destination address field. Our first attack was ICMP blind connection reset and for this we had to send ICMP hard error message. We know that protocol unreachable, port unreachable and fragmentation needed messaged are considered as ICMP hard error messages. In our second task, we will send ICMP source quench message to reduce the throughput of the data transmission. The codes for the corresponding error messages are given below:

a. Task 1:
   - Protocol unreachable (Type = 3, code = 2)
   - Port Unreachable (Type = 3, code = 3)
   - Fragmentation needed (Type = 3, code = 4)

b. Task 2:
   - Source quench message (Type = 4, code = 0)

Then we filled up the type and code field with proper value of the type of attack we were going to perform. The checksum was calculated before sending the packet.

4. The code for creating and sending the packets were written in C++. We used basic C++ libraries in Linux to create and send the packet

5. The ICMP packets were sent from attacker machine to the server. The packets were visible in Wireshark. It was confirmed that the packets reached the destination successfully. But unfortunately, the attack was not successful. Several attempts were made to close the TCP connection but nothing happened.

Compiling the Code:

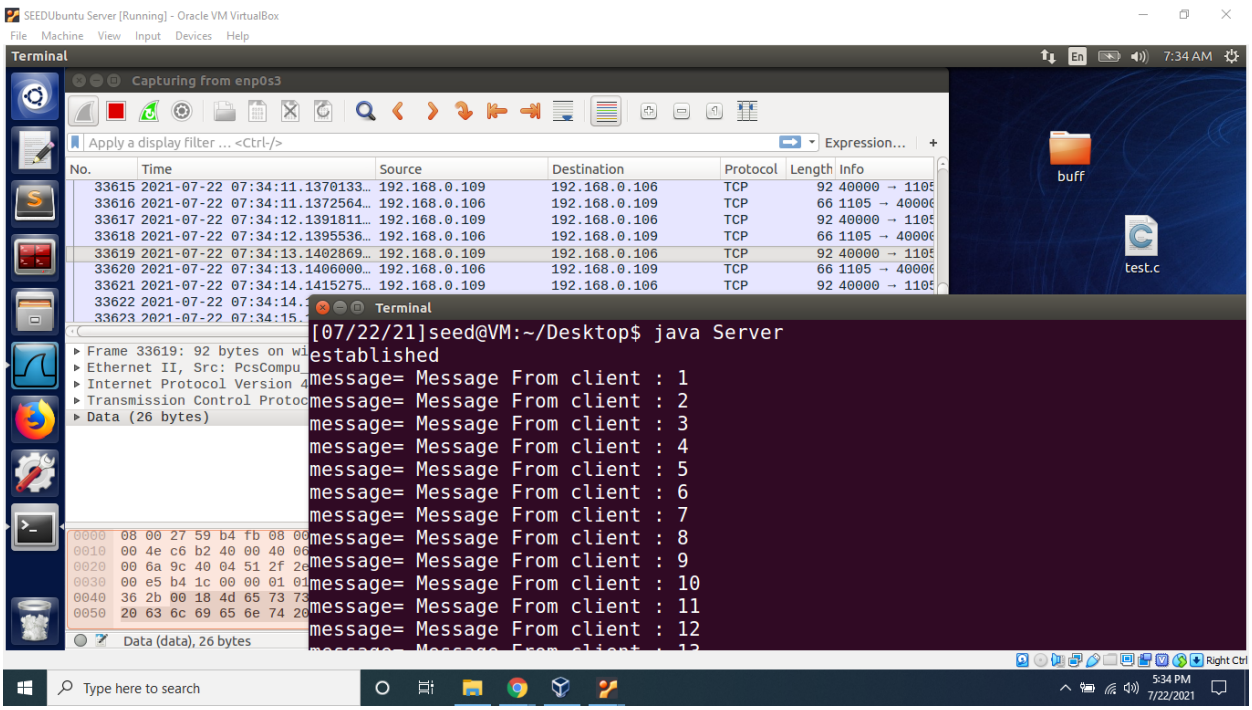
g++ -std=c++11 icmpAttack.cpp

Running the Code:

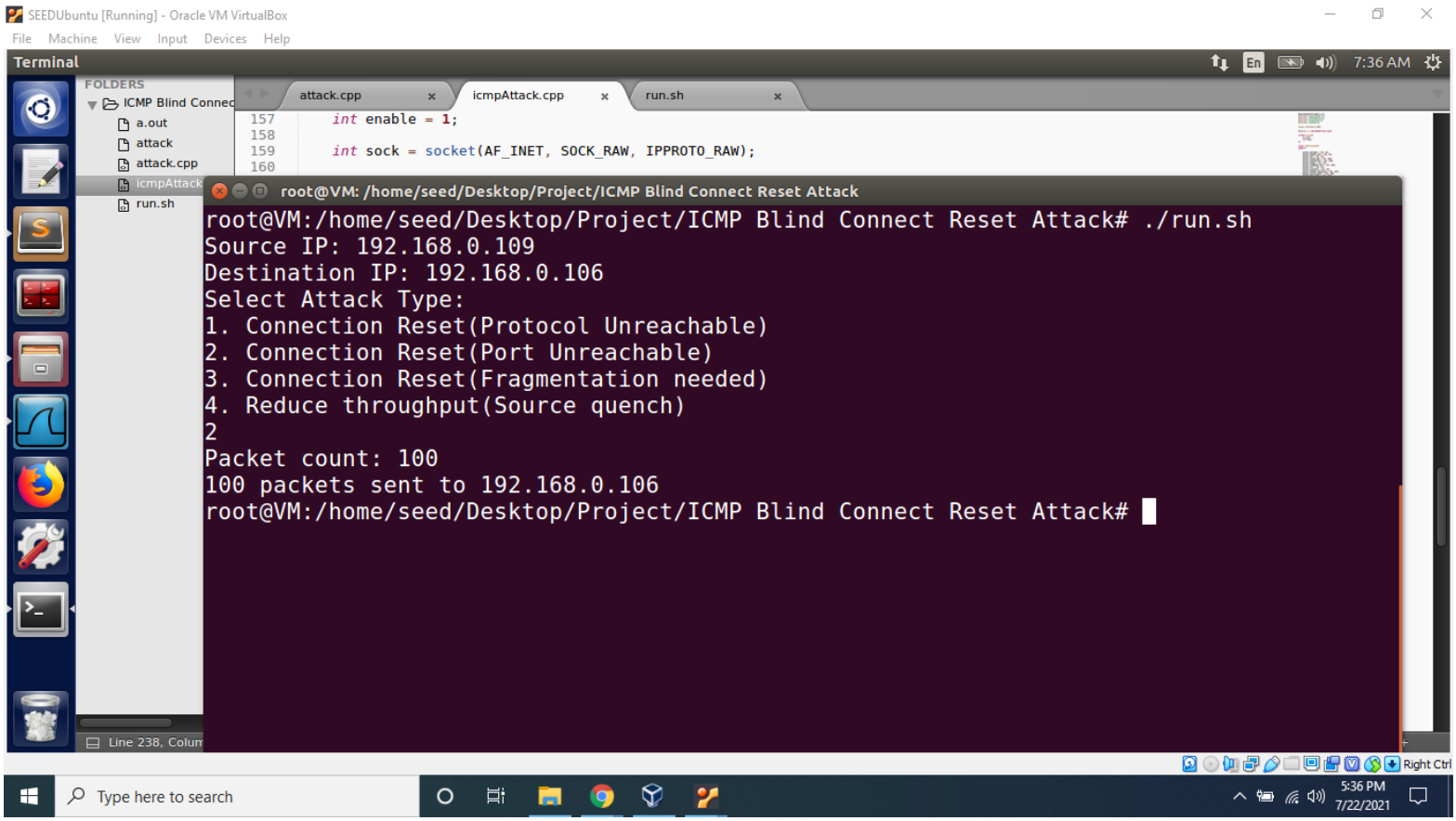
sudo ./a.out

Our Procedures:

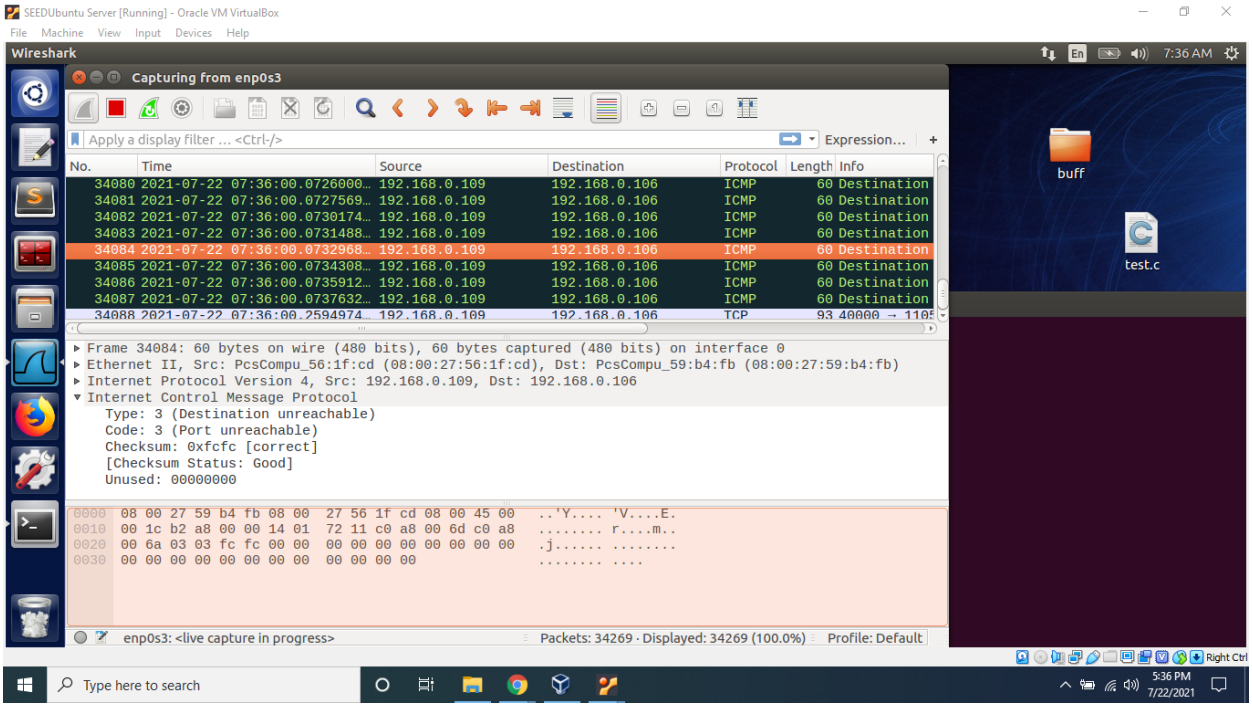
First, the TCP connection was established. Messages were being received from the client to the server side. The TCP packets were also visible in Wireshark.



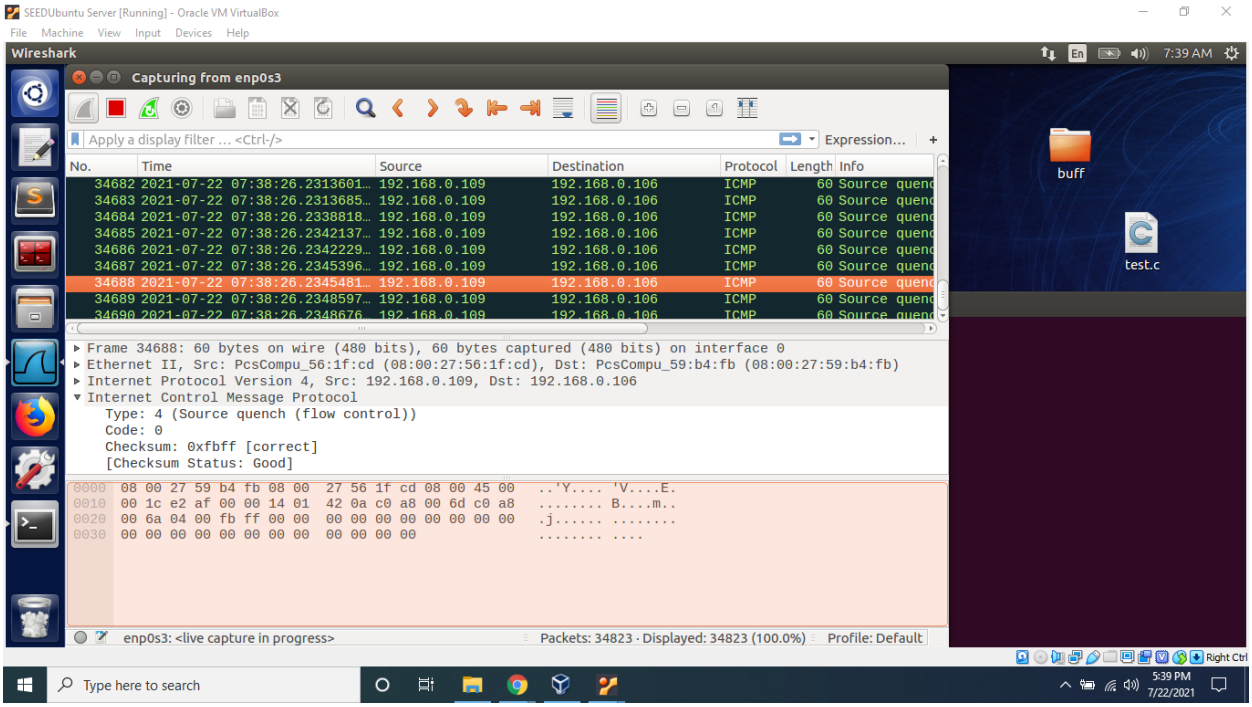
Then, we run our program. The program take input of the source-destination IP, attack type and packet counts.



The ICMP packets are visible on the server side but nothing happened to the TCP connection.



The same happened for the source quench message.



The attack was not successful. The reason behind the failure of attack is that current operating systems have implemented preventive measure for ICMP attacks.

"Based on this analysis, most popular TCP implementations treat all ICMP "hard errors" received for connections in any of the synchronized states (ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, or TIME-WAIT) as "soft errors". That is, they do not abort the corresponding connection upon receipt of them." [1]

"This counter-measure has been implemented in BSD-derived TCP/IP implementations (e.g., [FreeBSD], [NetBSD], and [OpenBSD]) for more than ten years [Wright][McKusick]. The Linux kernel has also implemented this policy for more than ten years [Linux]." [1]

"As discussed in the "Requirements for IP Version 4 Routers" RFC [RFC1812], research seems to suggest that ICMPv4 Source Quench messages are an ineffective (and unfair) antidote for congestion. [RFC1812] further states that routers SHOULD NOT send ICMPv4 Source Quench messages in response to congestion. Furthermore, TCP implements its own congestion control mechanisms ([RFC5681] [RFC3168]) that do not depend on ICMPv4 Source Quench messages. Based on this reasoning, a large number of implementations completely ignore ICMPv4 Source Quench messages meant for TCP connections." [1]

So according to the reference, those ICMP attacks are no longer possible in current systems. This is why our attack was not successful.

References:

[1] (http://www.faqs.org/rfcs/rfc5927.html)