Shane McDonough

Kadejha Jones

Shawn Ching

Lab 5

Traceroute

1. **Start a new wireshark capture**
2. **Run a traceroute to scrivnor.cikeys.com**
3. **How many hops did it take to get to the web server?**

   13 hops had been completed.
4. **Did any hops not respond? (it will appear as a *) If so, which ones?**

   5 hops did not respond. Hops 8 through 12 did not respond.
5. **Design a Wireshark filter to find the traceroute messages leaving your machine**

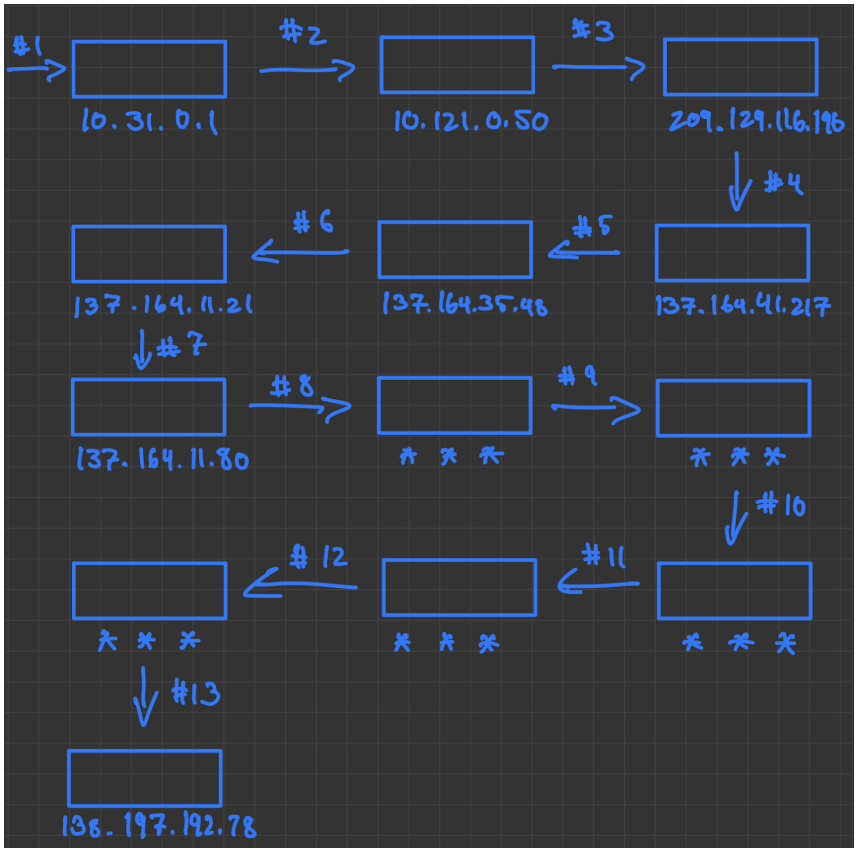
   ip.src == 10.31.13.43 && ip.dst == 138.197.192.78 && !icmp
6. **Can you write a filter to find the traceroute message responses?**

   It might not be possible to make a generic one to find ALL responses

   i. ip.ttl.too_small
7. **What type of message is traceroute sending to the server?**

   The message type is UDP.
8. **What protocol do the routers use for the responses? What is the message type?**

   The protocol they use is ICMP. The message type is Time-to-live-exceeded.
9. **Draw the route from your machine to the server. Label the routers with IP addresses and hop #.**



Syscalls()

1. **Socket() returns an integer, how is that number used in the calls to the other networking functions?**
    a. Socket returns a file descriptor. That integer is then used by “connect”, “setsockopt”, “bind”, “sendto”, “poll”, “ioctl”, etc as a file descriptor.
2. **What does SOCK_DGRAM, IPPROTO_UDP mean? How is it used in the syscalls?**
    a. SOCK_DGRAM is a flag that tells the socket to support the datagrams.
    b. IPPROTO_UDP is a flag that indicates the UDP protocol.
    c. Both are used as arguments to socket.
3. **What IP address and port number is the socket being bound to?**
    a. IP address: 0.0.0.0
    b. Port number: 0

4. **We know from the previous qanda section, that the TTL is increasing by 1 every time. How/where does this show up in the syscalls?**
    a. This shows up in the setsockopt function call. IP_TTL is passed in as the optname and optval gets a value one bigger than the last.
5. **A connection is then established on the socket, where is the traffic always going?**
    a. The traffic is going to the socket address also known as the destination.
        i. In our case the specific destination: 138.197.192.78 at varying ports starting at 33434 and increasing by one each hop.
    b. The destination is the value being passed to the second argument of “connect”, “addr”.
6. **Finally, sendto() is called.**
    a. **What data is sent on the socket?**
        i. The socket is sending “@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_”
    b. **What is the size of the data?**
        i. The size of the data is 32.
    c. **What does the function return? Why?**
        i. When it succeeds it returns the number of bytes sent.
        ii. When it fails it returns -1
        iii. In our case it returns 32 bytes.

NAT

1. **Identify the IP addresses of the server: ipchicken.com**
    a. 104.26.8.109
2. **Capture wireshark packets leaving your computer to https://www.ipchicken.com, use a filter to find them**
    a. **HTTPS uses port 443**

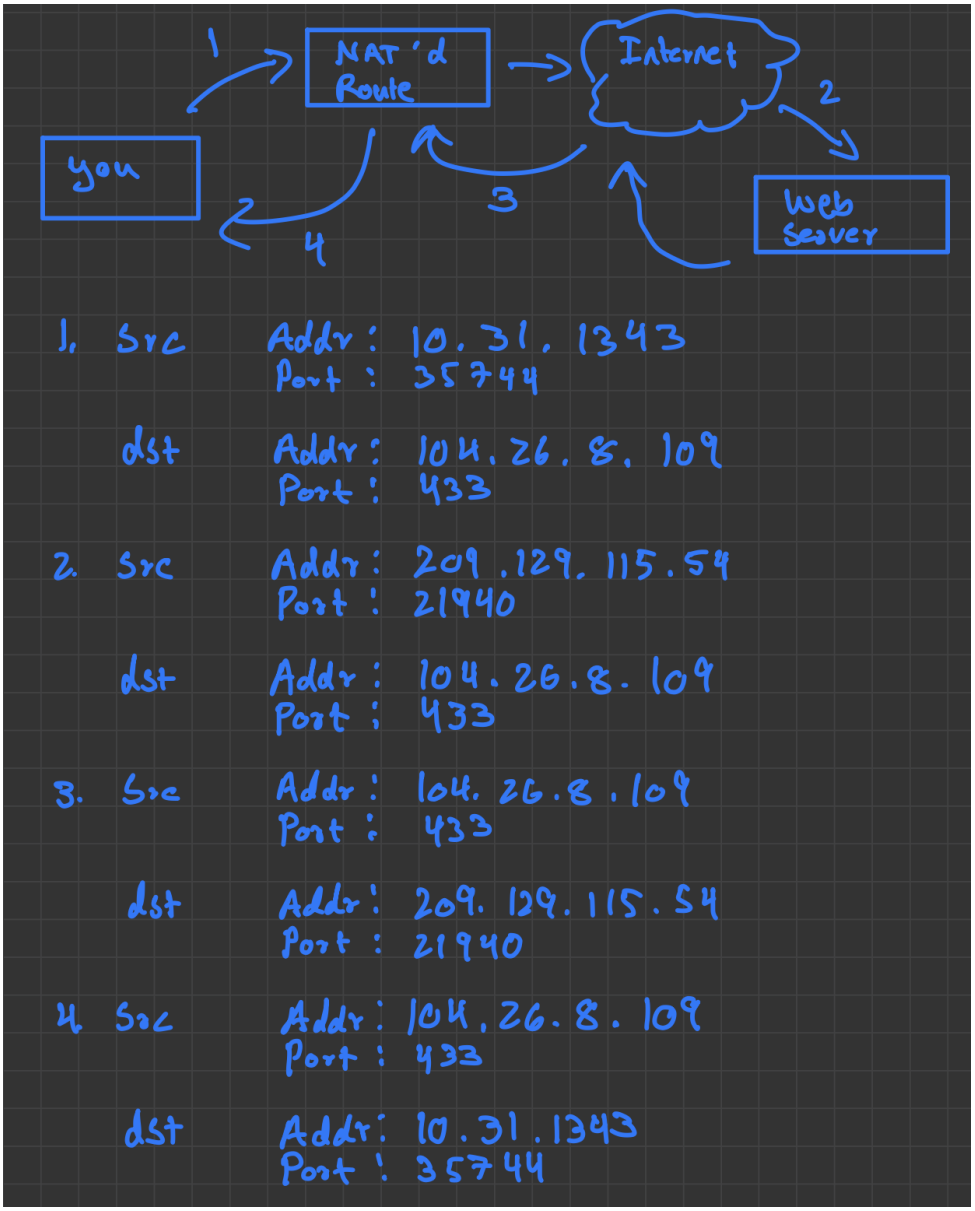
b. **The IP address will be one of the three from answer number 1**

c. **Stop your capture once you have traffic**

d. **Save/submit your capture file with your answers**

3. **At this point we have enough information to determine the source IP, destination IP, source port, and destination port of each socket for arrows 1-6 in the diagram on the board**

    a. Draw your diagram



Ping & Syscalls()

1. **Find the syscall in ping that is not permitted as a user**
    a. Socket was not permitted as a user.
2. **Let the user use SOCK_RAW**
    a. **Copy /usr/bin/ping into your home folder as myping**
    b. **Ensure the program myping is owned by student with ls -l**
    c. **Ensure the operation is not permitted with: ./myping -N ipv4 10.0.0.1**
    d. **Use man 7 capabilities to find the linux capability that will allow for raw sockets**
    e. **Set the capability to permissive using the setcap command**
        i. **sudo setcap 'CAPABILITY+p' ./myping, where CAPABILITY is the thing you found in the man page**
    f. **Verify the capability was set using getcap ./myping**
    g. **Now, ./myping -N ipv4 10.0.0.1 should work!**
        i. DONE!