tcpdump Exercises

1. Run tcpdump and read the input file **concepts.pcap**. How many records were displayed? What command did you use?
2. Now, run tcpdump with the -n option which disables hostname resolution. Does this help the command run faster?
3. Limit the output to the first two records in **concepts.pcap**. What option did you use?
4. What is the destination IP address of the third record?
5. Suppress the output of timestamps when reading the input file **concepts.pcap**. What option did you use?
6. Use tcpdump to display the first record from **concepts.pcap** and display it in hexadecimal with the timestamp suppressed. What is the full command that you used?
    a. What is the value of the first byte in the pcap? Can you find the hexadecimal value for the TTL in the IP header?
7. Using the **-v** option can you verify the TTL value from 6.a? Does the hexadecimal value from 6.a match the decimal output?
8. Display the MAC addresses in the first record of **concepts.pcap**. What command option did you use? What is the source MAC address?
9. Display the hexadecimal output of the last record in **concepts.pcap** with name resolution and timestamp suppressed. What is the hexadecimal value of the protocol field? What protocol does this indicate?
10. Based on the ports used in the last two records in **concepts.pcap**, what UDP protocol is being used?

Use **tcpdump.pcap** for the following questions:

1. Write a tcpdump filter to display the packets with a source host address of 127.0.0.1 and only the ACK flag set and no other flags set. Examine the packets that were displayed by the filter to verify that only the ACK flag is set in each packet.
2. Write a tcpdump filter to display the packets with a destination address of 127.0.0.1 and either the RST or ACK flags set and may have any other flags set. Examine the packets that were displayed to verify your filter.
3. Write a tcpdump filter to display records with port 80 and RST, SYN, and FIN flags set and any other flags may be set.
4. Write a tcpdump filter that displays the packets with a destination network address of 10.10.10/24 and a value in the final octet from 208-223 and 240-255. What filter did you use?
5. Write a tcpdump filter that displays the single packet that has the word GET in the first

You may use Wireshark or tcpdump for the following questions.

1. Use **TCP.pcap**, find the packet with destination host address 192.168.2.109 and source port of 2056. What is wrong with the packet? What tool and filter did you use?
2. Examine the two packets with source port 4545. What is odd about these packets? Why is payload on these packets unusual? Could this be a valid retransmission?
3. Compare two sets of packets, one with source port 10.154.1.8 and the other with destination port 143. One connection is a series of retries to connect to a non-responding host/network. The other is a series of successful SYN connections. Which set is which? Explain why you think your answer is correct.
4. Use **udp-icmp.pcap**, why should there be no response to the first packet?
5. Use **udp-icmp.pcap**, why should there be no response to the fifth packet?
6. Use **udp-icmp.pcap**, packets 37-46 are related. Can you determine what the activity is in these packets?

tcpdump Answers

1. tcpdump -r concepts.pcap, 6 records
2. Yes
3. -c 2
4. 192.168.11.13
5. -t
6. tcpdump -r concepts.pcap -ntxc 1
    a. 0x4500 , 0x40
7. 0x40 = 64
8. -e , aa:00:04:00:0a:04
9. 0x11 , UDP
10. DNS

tcpdump.pcap Questions

1. tcpdump -r tcpdump.pcap -nt 'src host 127.0.0.1 and tcp[13] = 0x10'
2. tcpdump -r tcpdump.pcap -nt 'dst host 127.0.0.1 and tcp[13] & 0x14 != 0'
3. tcpdump -r tcpdump.pcap -nt 'port 80 and tcp[13] & 0x07 =0x07'
4. tcpdump -r tcpdump.pcap -nt ' dst net 10.10.10 and ip[19] & 0xd0 = 0xd0'

Wireshark or tcpdump Questions

1. The checksum has an invalid value of all 0's. The packet would be dropped by the receiving host.

    tcpdump filter is: 'dst host 192.168.2.109 and src port 2056'

    Wireshark filter is: 'ip.dst == 192.168.1.09 and tcp.srcport == 2056'

2. These are two sequencial packets with the same source and desitnation values. The sequence numbers and payload lengths are the same. This could be a retransmission, however there are several oddities. The first packet is a SYN packet with payload, which is unsual. Next, if we examine the payloads we see that they are different. If this were a retransmission, the payloads would be the same. This could be used for IDS/IPS evasion.

    tcpdump -r TCP.pcap -nX 'src port 4545'

3. The first set, with source host 10.254.1.8, is the retries and the second set, with destination port 143, are the succesfull connection. The first set of connections has the same host, source port (3655) and TCP sequence number which is indicative of a series of retries. The second set of connections has the same host IP, however the source ports and TCP sequence numbers change, which is indicative of several different connection attempts.

    tcpdump filter: 'src host 10.254.1.8 or tcp dst port 143'

    Wireshark display filter: ip.src == 10.254.1.8 or tcp.dstport == 143

4. There should be no ICMP response to an ICMP echo request sent to a broadcast address.
5. There should never be an ICMP error message response to an ICMP error message. This would result in an endless loop.
6. These packets show a covert channel over ICMP. The first indication is that there are more echo replies than echo requests. As we inspect closer we can see that there is payload that is appears to be legible, and is actually part of an OpenSSH payload of supported keys.