ITN-262

Lab 02 - Command Line Decoding

Last updated: 09/03/2021

Objective

The primary objective of this assignment is to become familiar with the Domain Name Service (DNS) application layer protocol, Wireshark, and decoding packets.

STUDENT LEARNING OUTCOMES

1. Find name servers with DNS queries.
2. Explain the meaning of packet colors in Hex Packet Decoder and the layers in Wireshark.
3. Examine packet headers.
4. Examine packet captures in tcpdump.
5. Examing packet captures in Wireshark.

Setup

This lab will require installation of specific tools. This can be accomplished via:

```
apt-get update
apt-get upgrade -y
apt-get install -y tcpdump wireshark
```

You'll find the below-mentioned "dns-query-response.pcapng" file in the desktop's Labs folder. Wireshark can be accessed via the upper-left menu by selecting "Applications -> Internet -> Wireshark". Using tcpdump will require that you open a terminal window.

Steps

Let us start by examining the simplest DNS query capture: a single packet query followed by a single packet response. Open dns-query-response.pcapng in Wireshark, then answer the following questions:

1) Flag Question: What is the IP address of the DNS server using dot-decimal notation?

2) Flag Question: Which transport protocol (from the OSI model) is used?

3) Flag Question: What domain name is being looked up?

Let's take a deeper look into the tcpdump program. Read the following:

- Comprehensive Guide to tcpdump (Part 1) by Raj Chandel
  (https://www.hackingarticles.in/comprehensive-guide-to-tcpdump-part-1/)
- Comprehensive Guide to tcpdump (Part 2) by Raj Chandel
  (https://www.hackingarticles.in/comprehensive-guide-to-tcpdump-part-2/)
- Comprehensive Guide to tcpdump (Part 3) by Raj Chandel
  (https://www.hackingarticles.in/comprehensive-guide-to-tcpdump-part-3/)

Access your Linux machine via the link in RTB. Answer the questions by submitting them in RTB.

Reading the pcapng file into the tcpdump program:

```
tcpdump -r dns-query-response.pcapng
reading from file dns-query-response.pcapng, link-type EN10MB (Ethernet)
11:22:47.202144 IP 192.168.0.114.1060 > dns.asm.bellsouth.net.domain: 6159+ A?
wireshark.org. (31)
111:22:47.293308 IP dns.asm.bellsouth.net.domain > 192.168.0.114.1060: 6159 1/0/0
A
128.121.50.122 (47)
```

Notice there are no IP addresses displayed? Take a look at the man page for tcpdump using: *man tcpdump* How can you get the IP addresses to appear?

```
tcpdump -n -r dns-query-response.pcapng
reading from file dns-query-response.pcapng, link-type EN10MB (Ethernet)
11:22:47.202144 IP 192.168.0.114.1060 > 205.152.37.23.53: 6159+ A? wireshark.org.
(31)
11:22:47.293308 IP 205.152.37.23.53 > 192.168.0.114.1060: 6159 1/0/0 A
128.121.50.122 (47)
```

Try adding some different flags to the end and examine the output. Just try one at a time. What do the -n, -e, -#, -q, -vvv, -X flags do? Can you combine them?

```
tcpdump -n -r dns-query-response.pcapng -enqvvvX#
reading from file dns-query-response.pcapng, link-type EN10MB (Ethernet)
1 11:22:47.202144 00:16:ce:6e:8b:24 > 00:05:5d:21:99:4c, IPv4, length 73:
(tos 0x0, ttl 128, id 7975, offset 0, flags [none], proto UDP (17), length 59)
192.168.0.114.1060 > 205.152.37.23.53: [udp sum ok] UDP, length 31
0x0000: 4500 003b 1f27 0000 8011 67c1 c0a8 0072 E..;.'....g....r
0x0010: cd98 2517 0424 0035 0027 036d 180f 0100 ..%..$.5.'.m....
0x0020: 0001 0000 0000 0000 0977 6972 6573 6861 .........wiresha
0x0030: 726b 036f 7267 0000 0100 01
rk.org.....
2 11:22:47.293308 00:05:5d:21:99:4c > 00:16:ce:6e:8b:24, IPv4, length 89:
(tos 0x0, ttl 50, id 30333, offset 0, flags [DF], proto UDP (17), length 75)
205.152.37.23.53 > 192.168.0.114.1060: [udp sum ok] UDP, length 47
0x0000: 4500 004b 767d 4000 3211 1e5b cd98 2517 E..Kv}@.2..[..%.
0x0010: c0a8 0072 0035 0424 0037 3c20 180f 8180 ...r.5.$.7<.....
0x0020: 0001 0001 0000 0000 0977 6972 6573 6861 .........wiresha
0x0030: 726b 036f 7267 0000 0100 01c0 0c00 0100 rk.org..........
0x0040: 0100 0038 4000 0480 7932 7a
...8@...y2z
```

Take a close look at the output.

```
0x0000:  4500 003b 1f27 0000 8011 67c1 c0a8 0072   E..;.'....g....r
0x0010:  cd98 2517 0424 0035 0027 036d 180f 0100   ..%..$.5.'.m....
0x0020:  0001 0000 0000 0000 0977 6972 6573 6861   .........wiresha
0x0030:  726b 036f 7267 0000 0100 01               rk.org.....
```

What OSI layer is this packet displayed at? Ethernet, IP, TCP, UDP? Notice that it starts with the hexidecimal values 45. What flag would you add to see the link level (Data Link) headers?

```
tcpdump -n -r dns-query-response.pcapng -xx
reading from file dns-query-response.pcapng, link-type EN10MB (Ethernet)
11:22:47.202144 IP 192.168.0.114.1060 > 205.152.37.23.53: 6159+ A? wireshark.org.
(31)
0x0000: 0005 5d21 994c 0016 ce6e 8b24 0800 4500
0x0010: 003b 1f27 0000 8011 67c1 c0a8 0072 cd98
0x0020: 2517 0424 0035 0027 036d 180f 0100 0001
0x0030: 0000 0000 0000 0977 6972 6573 6861 726b
0x0040: 036f 7267 0000 0100 01
11:22:47.293308 IP 205.152.37.23.53 > 192.168.0.114.1060: 6159 1/0/0 A
128.121.50.122 (47)
0x0000: 0016 ce6e 8b24 0005 5d21 994c 0800 4500
0x0010: 004b 767d 4000 3211 1e5b cd98 2517 c0a8
0x0020: 0072 0035 0424 0037 3c20 180f 8180 0001
0x0030: 0001 0000 0000 0977 6972 6573 6861 726b
0x0040: 036f 7267 0000 0100 01c0 0c00 0100 0100
0x0050: 0038 4000 0480 7932 7a
```

Notice what is displayed now? The destination & source MAC addresses and EtherType or most of the Ethernet header information is included now. The Preamble and start of the frame delimiter are typically excluded from tcpdump and Wireshark.

```
0x0000: 0005 5d21 994c 0016 ce6e 8b24 0800 4500
```

To match the image below it would have to look like this:

```
0x0000: 80 00 20 7A 3F 3E 80 00 20 02 3A AE 08 00 45 00
```
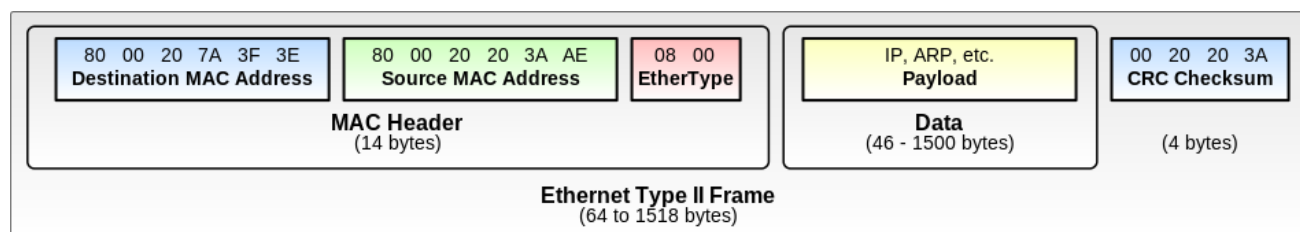

What about the 45 00 at the end? That is actually in the Data part of the image, and what layer of the OSI model is next?

That *45 00* is the IPv4 version information or the very begining of the next layer, in this case the IP layer. Remember that the networking layers are encapsulated. Take a look at Matt Baxter's IPv4 diagram from the Nmap book (https://nmap.org/book/images/hdr/MJB-IP-Header-800x576.png).

This is going to be helpful in decoding packets because the packet is shown including the MAC addresses from the Data Link layer, that is how the Hex Packet Decoder web application by Salim Gasmi works best, and its a good habit to develop when getting started decoding packets. Find a few sets of flags that help you understand the packets, this will be dependent on what you are doing, but for protocol analysis these are a good start.

```
tcpdump -r dns-query-response.pcapng -enxx#X
reading from file dns-query-response.pcapng, link-type EN10MB (Ethernet)
1 11:22:47.202144 00:16:ce:6e:8b:24 > 00:05:5d:21:99:4c, ethertype IPv4 (0x0800),
length 73: 192.168.0.114.1060 > 205.152.37.23.53: 6159+ A? wireshark.org. (31)
 0x0000: 4500 003b 1f27 0000 8011 67c1 c0a8 0072 E..;.'....g....r
 0x0010: cd98 2517 0424 0035 0027 036d 180f 0100 ..%..$.5.'.m....
 0x0020: 0001 0000 0000 0000 0977 6972 6573 6861 .........wiresha
 0x0030: 726b 036f 7267 0000 0100 01              rk.org.....
2 11:22:47.293308 00:05:5d:21:99:4c > 00:16:ce:6e:8b:24, ethertype IPv4 (0x0800),
length 89: 205.152.37.23.53 > 192.168.0.114.1060: 6159 1/0/0 A 128.121.50.122 (47)
 0x0000: 4500 004b 767d 4000 3211 1e5b cd98 2517 E..Kv}@.2..[..%.
 0x0010: c0a8 0072 0035 0424 0037 3c20 180f 8180 ...r.5.$.7<.....
 0x0020: 0001 0001 0000 0000 0977 6972 6573 6861 .........wiresha
 0x0030: 726b 036f 7267 0000 0100 01c0 0c00 0100 rk.org..........
 0x0040: 0100 0038 4000 0480 7932 7a              ...8@...y2z
```

Okay, let's start decoding some packets.

Browse to Hex Packet Decoder (by Salim Gasmi (https://hpd.gasmi.net/) (HPD) and select the "Decode" button:

```
00 05 5D 21 99 4C 00 16 CE 6E 8B 24 08 00 45 00 00 3B 1F 27 00 00 80 11 67 C1 C0
A8 00 72
CD 98 25 17 04 24 00 35 00 27 03 6D 18 0F 01 00 00 01 00 00 00 00 00 00 09 77 69
72 65 73
68 61 72 6B 03 6F 72 67 00 00 01 00 01
```

4) Flag Question: What is the source port (in Hexadecimal)?

Enter the following hexadecimal values into HPD values and select the "Decode" button:

```
00 16 CE 6E 8B 24 00 05 5D 21 99 4C 08 00 45 00 00 4B 76 7D 40 00 32 11 1E 5B CD
98 25 17
C0 A8 00 72 00 35 04 24 00 37 3C 20 18 0F 81 80 00 01 00 01 00 00 00 00 09 77 69
72 65 73
68 61 72 6B 03 6F 72 67 00 00 01 00 01 C0 0C 00 01 00 01 00 00 38 40 00 04 80 79
32 7A
```

Remember that binary to decimal and hexadecimal conversion you learned? You are going to need that here.

5) Flag Question: Modify the hexadecimal values above to change the source address from source: 205.152.37.23 to source: 222.202.253.173 and submit the entire hexidemical output with the modified values.

Some key takeaways:

- Understand how big a byte is (8 bits), but how is it represented in hexadecimal in packet decoding?
- Understand how big a nibble is (4 bits), but how is it represented in hexadecimal in packet decoding?
- What would the hexadecimal value: 0F look like in binary, and decimal?
- What does the binary value 1001 represent in decimal and hexadecimal?
- Understand different flags for tcpdump, and tcpdump in general.
- Understand how Wireshark can be used to look at different parts of the Ethernet Frame (and what parts are omitted).

References

- Dot-decimal notation From Wikipedia, the free encyclopedia (https://en.wikipedia.org/wiki/Dot-decimal_notation)
- Hex Packet Decoder by Salim Gasmi (https://hpd.gasmi.net/)
- Ethernet Type II Frame Public Domain (https://commons.wikimedia.org/w/index.php?curid=1546835)