

430/530 – Wireshark Lab

Name: Jacob Boicken

Please review the lab directions and example screen capture provided in Canvas for exactly what is expected for each packet header image submission.

Ethernet Header Screen Capture

```
▼ Ethernet II, Src: Cisco_31:01:aa (00:00:0c:31:01:aa), Dst: Andiamo_1e:40:40 (00:05:30:1e:40:40)
  ▼ Destination: Andiamo_1e:40:40 (00:05:30:1e:40:40)
    Address: Andiamo_1e:40:40 (00:05:30:1e:40:40)
      ....0. .... = LG bit: Globally unique address (factory default)
      ....0. .... = IG bit: Individual address (unicast)
  ▼ Source: Cisco_31:01:aa (00:00:0c:31:01:aa)
    Address: Cisco_31:01:aa (00:00:0c:31:01:aa)
      ....0. .... = LG bit: Globally unique address (factory default)
      ....0. .... = IG bit: Individual address (unicast)
  Type: IPv4 (0x0800)
  ▶ Internet Protocol Version 4, Src: 199.100.16.100, Dst: 64.39.3.194
  ▶ Transmission Control Protocol, Src Port: 3128, Dst Port: 36010, Seq: 1672, Ack: 2241, Len: 0

0000  00 05 30 1e 40 40 00 00 0c 31 01 aa 08 00 45 00  ..0..  .1...E.
0010  00 34 00 00 40 00 3c 06 23 13 c7 64 10 64 40 27  .4...@.<.#.d.d@'
0020  03 c2 0c 38 8c aa f5 37 e5 97 92 11 3c 66 80 10  ...8...7...<f...
0030  04 03 e8 6a 00 00 01 01 08 0a cf 40 0e 4b 2d 48  ...j....@.K-H
0040  21 a0  !.
```

Ethernet Header – Broadcast Packet Screen Capture

Find an Ethernet packet that is a broadcast packet. This capture should be a different packet from the **Ethernet Header Screen Capture** on the previous page.

```
▼ Ethernet II, Src: Andiamo_1e:4a:4a (00:05:30:1e:4a:4a), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▼ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    Address: Broadcast (ff:ff:ff:ff:ff:ff)
    ....1. .... = LG bit: Locally administered address (this is NOT the factory default)
    ....1. .... = IG bit: Group address (multicast/broadcast)
  ▼ Source: Andiamo_1e:4a:4a (00:05:30:1e:4a:4a)
    Address: Andiamo_1e:4a:4a (00:05:30:1e:4a:4a)
    ....0. .... = LG bit: Globally unique address (factory default)
    ....0. .... = IG bit: Individual address (unicast)
  Type: IPv4 (0x0800)
▶ Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
▶ User Datagram Protocol, Src Port: 68, Dst Port: 67
▶ Dynamic Host Configuration Protocol (Discover)

0000 ff ff ff ff ff 00 05 30 1e 4a 4a 08 00 45 c0 ..... 0.JJ..E.
0010 01 44 00 00 40 00 40 11 38 ea 00 00 00 00 ff ff .D..@. 8.....
0020 ff ff 00 44 00 43 01 30 ac 8e 01 01 06 00 4b 5c ...D.C.0 .....K\
0030 ca a1 00 01 00 00 00 00 00 00 00 00 00 00 00 00 .....
0040 00 00 00 00 00 00 00 05 30 1e 4a 4a 00 00 00 00 ..... 0.JJ....
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0110 00 00 00 00 00 00 00 63 82 53 63 35 01 01 3d 07 01 .....c. Sc5...=..
0120 00 05 30 1e 4a 4a 37 11 01 02 06 0c 0f 1a 1c 79 ..0.JJ7.....y
0130 03 21 28 29 2a 77 f9 fc 11 39 02 02 40 32 04 40 .!(*)w...9..@2.
0140 27 03 78 0c 0c 6b 61 6c 69 2d 73 74 75 64 65 6e 'x..kal i-studen
0150 74 ff t.
```

IP Header with Kali IP Screen Capture

Most specifically, find an IP packet where the IP address matches the IP address of your Kali box. In addition to a screen capture of the IP header, include a screen capture of how you determined the IP address of your Kali box (circle the IP address on the capture, if necessary).

```
▼ Internet Protocol Version 4, Src: 199.100.16.100, Dst: 64.39.3.208
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ▼ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 52
  Identification: 0x0000 (0)
  ▼ Flags: 0x4000, Don't fragment
    0... .. = Reserved bit: Not set
    .1.. .. = Don't fragment: Set
    ..0. .. = More fragments: Not set
  Fragment offset: 0
  Time to live: 60
  Protocol: TCP (6)
  Header checksum: 0x2305 [validation disabled]
  [Header checksum status: Unverified]
  Source: 199.100.16.100
  Destination: 64.39.3.208
▶ Transmission Control Protocol, Src Port: 3128, Dst Port: 32850, Seq: 1, Ack: 228, Len: 0

0000  00 05 30 1e 0c 0c 00 00 0c 31 01 aa 08 00 45 00  ..0... .1...E.
0010  00 34 00 00 40 00 3c 06 23 05 c7 64 10 64 40 27  .4..@.< #..d@d'
0020  03 d0 0c 38 80 52 1d 95 51 e7 0d f0 a7 c8 80 10  ...8R.. Q.....
0030  04 03 ec 3d 00 00 01 01 08 0a e3 28 85 bb c2 fa  ...=... (...
0040  8d 1e                                     ..
```

Kali IP Address Screen Capture

```
student@kali-student:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 64.39.3.208 netmask 255.255.255.0 broadcast 64.39.3.255
    inet6 fe80::205:30ff:fe1e:c0c prefixlen 64 scopeid 0x20<link>
    ether 00:05:30:1e:0c:0c txqueuelen 1000 (Ethernet)
    RX packets 14 bytes 1848 (1.8 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 31 bytes 3075 (3.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

ARP Header – Request Packet Screen Capture

▼ Address Resolution Protocol (request)

Hardware type: Ethernet (1)

Protocol type: IPv4 (0x0800)

Hardware size: 6

Protocol size: 4

Opcode: request (1)

Sender MAC address: Andiamo_1e:3c:3c (00:05:30:1e:3c:3c)

Sender IP address: 64.39.3.158

Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)

Target IP address: 64.39.3.254

0000	00 00 0c 31 01 aa 00 05 30 1e 3c 3c 08 06 00 01	...1.... 0.<<....
0010	08 00 06 04 00 01 00 05 30 1e 3c 3c 40 27 03 9e 0.<<@'..
0020	00 00 00 00 00 00 40 27 03 fe 00 00 00 00 00 00@'.....
0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00

ARP Header – Reply Packet Screen Capture

▼ Address Resolution Protocol (reply)		
Hardware type: Ethernet (1)		
Protocol type: IPv4 (0x0800)		
Hardware size: 6		
Protocol size: 4		
Opcode: reply (2)		
Sender MAC address: Cisco_31:01:aa (00:00:0c:31:01:aa)		
Sender IP address: 64.39.3.254		
Target MAC address: Andiamo_1e:3c:3c (00:05:30:1e:3c:3c)		
Target IP address: 64.39.3.158		
0000	00 05 30 1e 3c 3c 00 00 0c 31 01 aa 08 06 00 01	..0.<<..1.....
0010	08 00 06 04 00 02 00 00 0c 31 01 aa 40 27 03 fe1..@'..
0020	00 05 30 1e 3c 3c 40 27 03 9e 00 00 00 00 00 00	..0.<<@'..
0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00

ICMP Header Screen Capture

▶ Internet Protocol Version 4, Src: 64.39.3.1, Dst: 64.39.3.122

▼ Internet Control Message Protocol

- Type: 8 (Echo (ping) request)
- Code: 0
- Checksum: 0x47b7 [correct]
- [Checksum Status: Good]
- Identifier (BE): 45128 (0xb048)
- Identifier (LE): 18608 (0x48b0)
- Sequence number (BE): 0 (0x0000)
- Sequence number (LE): 0 (0x0000)
- [\[Response frame: 1836\]](#)

▼ Data (20 bytes)

- Data: 00
- [Length: 20]

0000	00 05 30 1e 4a 4a 00 50 56 86 18 56 08 00 45 00	..0.JJ.P V..V..E.
0010	00 30 9f 46 00 00 40 01 54 be 40 27 03 01 40 27	.0.F..@. T.@'..@'
0020	03 7a 08 00 47 b7 b0 48 00 00 00 00 00 00 00 00	.z..G..H
0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

UDP Header Screen Capture

```
▼ User Datagram Protocol, Src Port: 38041, Dst Port: 53
  Source Port: 38041
  Destination Port: 53
  Length: 39
  Checksum: 0x190f [unverified]
  [Checksum Status: Unverified]
  [Stream index: 4]
  ▼ [Timestamps]
    [Time since first frame: 0.000000000 seconds]
    [Time since previous frame: 0.000000000 seconds]
  ▶ Domain Name System (query)
```

0000	00 00 0c 31 01 aa 00 05 30 1e 3c 3c 08 00 45 00	...1...0...<<...E...
0010	00 3b 95 86 40 00 40 11 89 9e 40 27 03 9e c7 64	...;...@...@...@'...d
0020	10 64 94 99 00 35 00 27 19 0f 68 4c 01 00 00 01	...d...5...'...hL...
0030	00 00 00 00 00 00 03 63 64 6e 05 66 77 75 70 64	...c dn fwupd
0040	03 6f 72 67 00 00 01 00 01	...org....

TCP Header Screen Capture

```

Transmission Control Protocol, Src Port: 49806, Dst Port: 3128, Seq: 2247, Ack: 4069, Len: 39
  Source Port: 49806
  Destination Port: 3128
  [Stream index: 33]
  [TCP Segment Len: 39]
  Sequence number: 2247 (relative sequence number)
  Sequence number (raw): 3371189385
  [Next sequence number: 2286 (relative sequence number)]
  Acknowledgment number: 4069 (relative ack number)
  Acknowledgment number (raw): 2193102174
  1000 .... = Header Length: 32 bytes (8)
  Flags: 0x018 (PSH, ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....1... = Push: Set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....0... = Fin: Not set
  [TCP Flags: .....AP...]
  Window size value: 501
  [Calculated window size: 64128]
  [Window size scaling factor: 128]
  Checksum: 0xa86d [unverified]
  [Checksum Status: Unverified]
  Urgent pointer: 0
  Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
    TCP Option - No-Operation (NOP)
      Kind: No-Operation (1)
    TCP Option - No-Operation (NOP)
      Kind: No-Operation (1)
    TCP Option - Timestamps: TSval 2796842952, TSecr 825360392
      Kind: Time Stamp Option (8)
      Length: 10
      Timestamp value: 2796842952
      Timestamp echo reply: 825360392
  [SEQ/ACK analysis]
    [iRTT: 0.001329942 seconds]
    [Bytes in flight: 39]
    [Bytes sent since last PSH flag: 39]
  [Timestamps]
    [Time since first frame in this TCP stream: 239.007197633 seconds]
    [Time since previous frame in this TCP stream: 58.877637146 seconds]
  TCP payload (39 bytes)
  Hypertext Transfer Protocol
  Transport Layer Security
0000 00 00 0c 31 01 aa 00 05 30 1e 5f 5f 08 00 45 00 ...1...0...E
0010 00 5b 4c db 40 00 40 06 d2 94 40 27 03 3e c7 64 .[L@. @. @'>.d
0020 10 64 c2 8e 0c 38 c8 f0 44 89 82 b8 15 5e 80 18 .d...8...D...^...
0030 01 f5 a8 6d 00 00 01 01 08 0a a6 b4 6f c8 31 32 ...m...o.12
0040 00 08 17 03 03 00 22 b7 bb 4c 01 0e 3f 76 c8 11 .....".L?v...
0050 a9 31 1d f5 e6 1f 33 06 c3 a3 17 d6 53 17 d2 60 .1...3...S...
0060 ff ed fb 1f f1 d7 c6 2d 60 .....

```


DNS Header Screen Capture

```

Domain Name System (response)
Transaction ID: 0xa842
Flags: 0x8180 Standard query response, No error
1... .. = Response: Message is a response
.000 0... .. = Opcode: Standard query (0)
... .. = Authoritative: Server is not an authority for domain
... .. = Truncated: Message is not truncated
... .. = Recursion desired: Do query recursively
... .. = Recursion available: Server can do recursive queries
... .. = Z: reserved (0)
... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
... .. = Non-authenticated data: Unacceptable
... .. = Reply code: No error (0)
Questions: 1
Answer RRs: 1
Authority RRs: 1
Additional RRs: 0
Queries
  cdn.fwupd.org: type AAAA, class IN
    Name: cdn.fwupd.org
    [Name Length: 13]
    [Label Count: 3]
    Type: AAAA (IPv6 Address) (28)
    Class: IN (0x0001)
  Answers
  Authoritative nameservers
  [Request In: 14362]
  [Time: 0.000892101 seconds]

0000 00 05 30 1e 40 40 00 00 0c 31 01 aa 08 00 45 00 ..0..@.. -1....E-
0010 00 ab c1 59 00 00 3c 11 a1 37 c7 64 10 64 40 27 ...Y...<-7-d-d@'
0020 03 c2 00 35 81 02 00 97 f3 4f a8 42 81 80 00 01 ...5....-0-B....
0030 00 01 00 01 00 00 03 63 64 6e 05 66 77 75 70 64 .....c dn-fwupd
0040 03 6f 72 67 00 00 1c 00 01 c0 0c 00 05 00 01 00 .org.....
0050 00 00 37 00 1d 02 70 32 06 73 68 61 72 65 64 06 ..7...p2 shared
0060 67 6c 6f 62 61 6c 06 66 61 73 74 6c 79 03 6e 65 global-f astly-ne
0070 74 00 06 66 61 73 74 6c 79 03 6e 65 74 00 00 06 t-fastly-net...
0080 00 01 00 00 00 19 00 31 03 6e 73 31 c0 48 0a 68 .....1 ns1-H-h
0090 6f 73 74 6d 61 73 74 65 72 06 66 61 73 74 6c 79 ostmaste r-fastly
00a0 03 63 6f 6d 00 78 39 c6 29 00 00 0e 10 00 00 02 .com-x9- ).....
00b0 58 00 09 3a 80 00 00 00 1e X.:....

```

Lab Reflection

Answer the following reflection questions using complete sentences - use details and be specific to avoid vague answers. All responses should be *at least* three sentences. Please spend time thinking of thoughtful responses.

a) Why would a “good actor” want to use a network packet analyzer program like Wireshark?

A good actor, like a network admin or myself on a Saturday night trying to fix network issues, could use wireshark to view their network to diagnose where in the network connection issues are. Such as my laptop not being able to reach my web proxy, I can use wireshark to see if packets are generated to my web proxy or not to then determine if it say my host firewall blocking out bound or proxy settings not being set up.

b) Why would a “bad actor” want to use a program like Wireshark? How could they use Wireshark to their advantage? What information could potentially be gained by using a program like Wireshark?

A bad actor / attacker could use wireshark for internal network reconnaissance or eavesdropping. They learn what devices talk to each other, the protocols/ports used, and any unencrypted traffic. This gives them a map of the network and can build attack points from there. Unencrypted traffic could hold information like password, auth/api tokens, or other sensitive pieces of information that is allow since it is internal, which an attacker could use.

c) Are there any countermeasures that could decrease the impact of the information a bad actor could obtain when using a program like Wireshark?

Countermeasures to malicious wireshark usage would first be internal encryption anywhere possible. From then on it depends on the network design/implementation. The computer can only read what is sent to its network device, so if the device is directly behind a switch it should only be able see traffic addressed to itself. The same goes for encrypted wireless. Hubs/buses and other network designs can allow for a single client to see more traffic than want it is meant to read.

d) What connection do you see between the book's materials, lectures, and assignments and what is presented in Wireshark? What are your reactions to the amount of information Wireshark provides about each packet?

From the lectures, we hear Dr. Jacobson talk about network devices operating in promiscuous mode. Wireshark shows us what that mode is able to pick up and what sniffing looks like (in scale). For the amount of information that wireshark picks up, its simply copying what the network device on the computer holds, so we get the headers/data of Ethernet, IP, TCP/UDP, and etc. This matches up with what we have seen in the hw/quizzes as well.

e) If you're new to Wireshark, what did you find most interesting about Wireshark? Be specific. If you're already familiar with Wireshark, seek out some new information, such as a different way to use filters, and share what you have learned.

I have used wireshark somewhat before. However, since I don't know the filters very well, I ended up trying to use “!=” as a not equals. This didn't work. Rather, you have to do !(ip.addr == x.x.x.x). As well, looking at the drop down list when typing in a filter, I found interesting searches you can do like looking up by geoip/location or selecting usb or usbaudio which are interfaces. You can then filter by usb/audio fields. I did not know wireshark can be used for all sorts of devices.