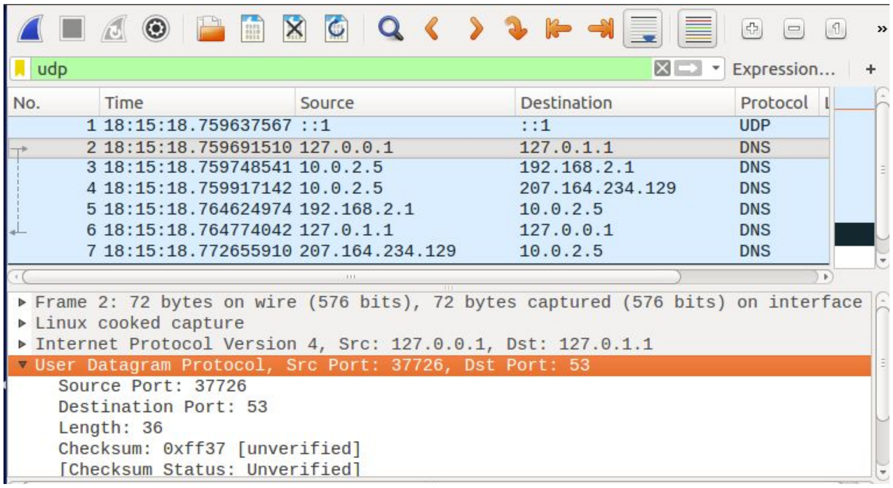
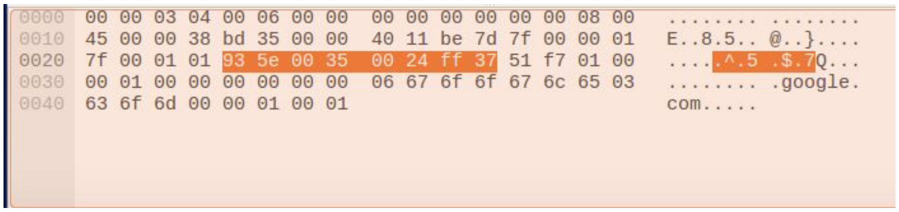
CPS 706 - Wireshark UDP

1. *Select one packet. From this packet, determine how many fields there are in the UDP header. Name these fields.*
   - There are 4 fields in the UDP header
   - Source port, destination port, length and checksum



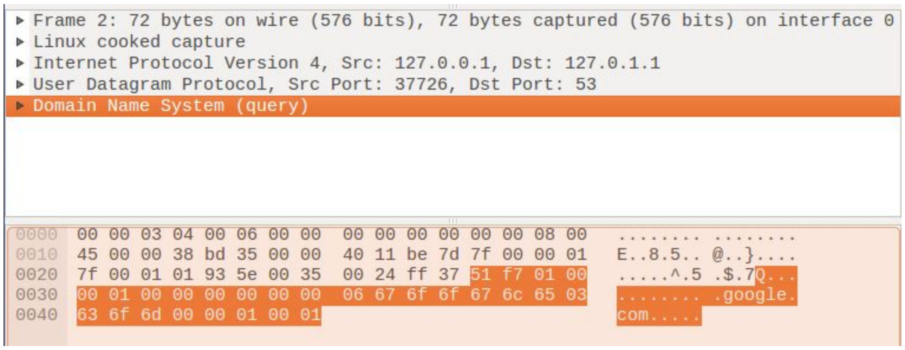
```
udp                                                        Expression...  +

No.   Time                   Source            Destination       Protocol
    1 18:15:18.759637567 ::1                   ::1               UDP
    2 18:15:18.759691510 127.0.0.1             127.0.1.1         DNS
    3 18:15:18.759748541 10.0.2.5              192.168.2.1       DNS
    4 18:15:18.759917142 10.0.2.5              207.164.234.129   DNS
    5 18:15:18.764624974 192.168.2.1           10.0.2.5          DNS
    6 18:15:18.764774042 127.0.1.1             127.0.0.1         DNS
    7 18:15:18.772655910 207.164.234.129       10.0.2.5          DNS

▸ Frame 2: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▾ User Datagram Protocol, Src Port: 37726, Dst Port: 53
    Source Port: 37726
    Destination Port: 53
    Length: 36
    Checksum: 0xff37 [unverified]
    [Checksum Status: Unverified]
```

2. *From the packet content field, determine the length (in bytes) of each of the UDP header fields.*
   - The total length of UDP header fields is 8 bytes
   - Each fields is 2 bytes



```
0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 38 bd 35 00 00  40 11 be 7d 7f 00 00 01   E..8.5.. @..}....
0020  7f 00 01 01 93 5e 00 35  00 24 ff 37 51 f7 01 00   .....^.5 .$.7Q...
0030  00 01 00 00 00 00 00 00  06 67 6f 6f 67 6c 65 03   ........ .google.
0040  63 6f 6d 00 00 01 00 01                            com.....
```

3. *The value in the Length field is the length of what? Verify your claim with your captured UDP packet*
   - The value of the Length field is the total length of both UDP header and data sent
   - Length of data sent is 28 + Length of UDP header is 8 = 36

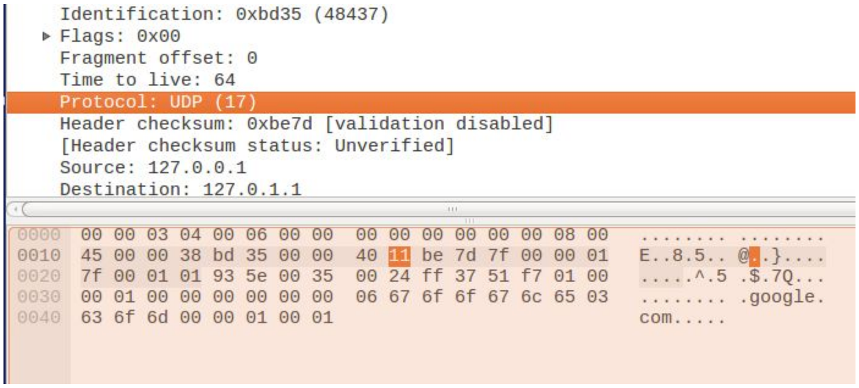


```
▸ Frame 2: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.1.1
▸ User Datagram Protocol, Src Port: 37726, Dst Port: 53
▸ Domain Name System (query)

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 38 bd 35 00 00  40 11 be 7d 7f 00 00 01   E..8.5.. @..}....
0020  7f 00 01 01 93 5e 00 35  00 24 ff 37 51 f7 01 00   .....^.5 .$.7Q...
0030  00 01 00 00 00 00 00 00  06 67 6f 6f 67 6c 65 03   ........ .google.
0040  63 6f 6d 00 00 01 00 01                            com.....
```

4. *What is the maximum number of bytes that can be included in a UDP payload?*
    - The maximum number of bytes that can be included is 65,527 bytes
    - 2^16 - 8 (bytes used for header) = 65,535 - 8 = 65,527 bytes

5. *What is the largest possible source port number?*
    - The largest possible source port is 2^16 - 1 = 65,535

6. *What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation*
    - Protocol number for UDP is 17 in decimal and 0x11 in hexadecimal

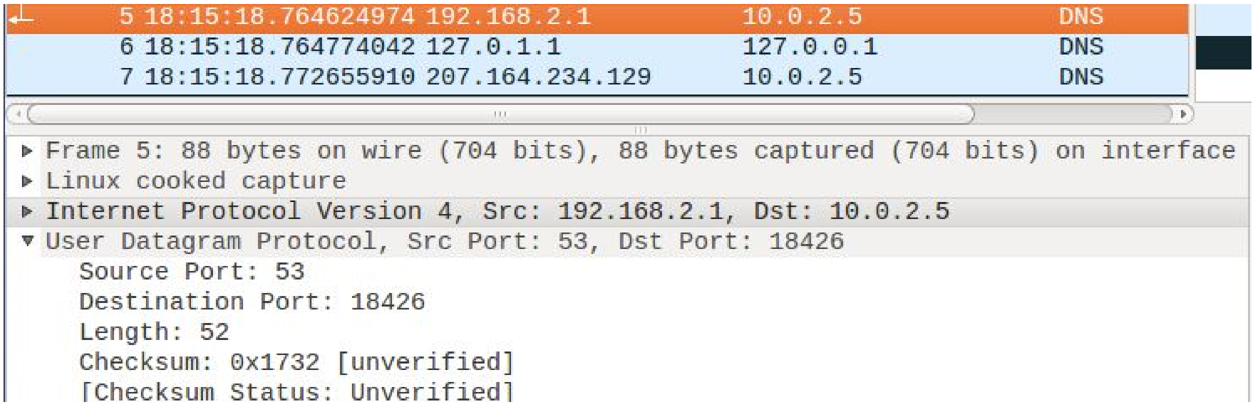


```
   Identification: 0xbd35 (48437)
 ▸ Flags: 0x00
   Fragment offset: 0
   Time to live: 64
   Protocol: UDP (17)
   Header checksum: 0xbe7d [validation disabled]
   [Header checksum status: Unverified]
   Source: 127.0.0.1
   Destination: 127.0.1.1

0000  00 00 03 04 00 06 00 00  00 00 00 00 00 00 08 00   ........ ........
0010  45 00 00 38 bd 35 00 00  40 11 be 7d 7f 00 00 01   E..8.5.. @..}....
0020  7f 00 01 01 93 5e 00 35  00 24 ff 37 51 f7 01 00   .....^.5 .$.7Q...
0030  00 01 00 00 00 00 00 00  06 67 6f 6f 67 6c 65 03   ........ .google.
0040  63 6f 6d 00 00 01 00 01                            com.....
```

7. *Search "UDP" in Google and determine the fields over which the UDP checksum is calculated.*
    - UDP checksum is the complement of 16-bit one's-complement sum calculated over an IP "pseudo-header" and actual UDP data
    - The IP pseudo-header is the source address, destination address, protocol (padded with zero byte) and UDP length

8. *Examine a pair of UDP packets in which the first packet is sent by your host and the second packet is a reply to the first packet. Describe the relationship between the port numbers in the two packets*
    - The port number of the UDP packet sent by my host are
        - Source: 53
        - Destination: 18426
    - The port number of the UDP reply packet sent to my host are:
        - Source: 18426
        - Destination: 53
    - The source port of the packet sent by my host is the same as the destination port of the reply packet. The destination port of the packet sent by my host is the same as the source port of the reply packet.

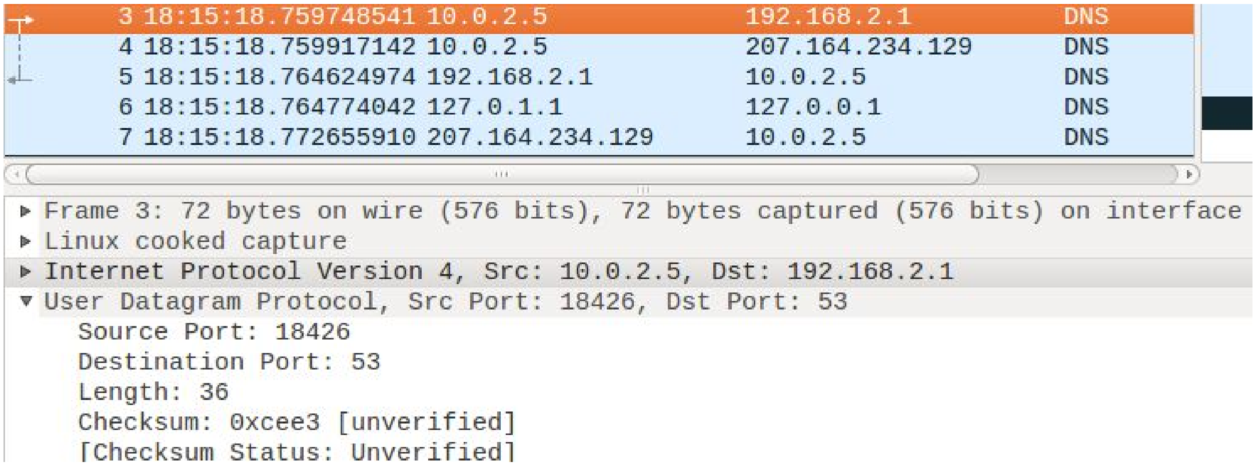
Sent by host



```
5 18:15:18.764624974 192.168.2.1       10.0.2.5          DNS
6 18:15:18.764774042 127.0.1.1         127.0.0.1         DNS
7 18:15:18.772655910 207.164.234.129   10.0.2.5          DNS
```

```
▸ Frame 5: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) on interface
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 192.168.2.1, Dst: 10.0.2.5
▾ User Datagram Protocol, Src Port: 53, Dst Port: 18426
    Source Port: 53
    Destination Port: 18426
    Length: 52
    Checksum: 0x1732 [unverified]
    [Checksum Status: Unverified]
```

Reply to host



```
3 18:15:18.759748541 10.0.2.5          192.168.2.1       DNS
4 18:15:18.759917142 10.0.2.5          207.164.234.129   DNS
5 18:15:18.764624974 192.168.2.1       10.0.2.5          DNS
6 18:15:18.764774042 127.0.1.1         127.0.0.1         DNS
7 18:15:18.772655910 207.164.234.129   10.0.2.5          DNS
```

```
▸ Frame 3: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 10.0.2.5, Dst: 192.168.2.1
▾ User Datagram Protocol, Src Port: 18426, Dst Port: 53
    Source Port: 18426
    Destination Port: 53
    Length: 36
    Checksum: 0xcee3 [unverified]
    [Checksum Status: Unverified]
```