Solutions to exercise 4 in EITF25 Internet - Techniques and Applications - 2013

October 26, 2013

1

In Figure 1 below, each requested component has been highlighted accordingly. In addition, the beginning of the IP packet has been emphasized.

```
08 00 20 7C 94 1C 00 00 39 51 90 37 08 00 45 00
00 3E 36 00 00 00 80 11 DA 4F 82 EB 12 7F 82 EB
12 0A 04 01 00 35 00 2A EE 6A 00 01 01 00 00 01
00 00 00 00 00 00 06 67 65 6D 69 6E 69 03 6C 64
63 02 6C 75 02 73 65 00 00 01 00 01
```

Figure 1: Header length, Protocol, Sender address, Destination address

a

Note that the Ethernet header (which does not include Preamble and SFD) has been greyed-out. As highlighted in Figure 1, the destination address is `0x82:EB:12:0A`, which in decimal is `130.235.18.10`.

$\boxed{\textbf{Answer}\text{: 130.235.18.10}}$

b

As highlighted in Figure 1, the sender address is `0x82:EB:12:7F`, which in decimal is `130.235.18.127`.

$\boxed{\textbf{Answer}\text{: 130.235.18.127}}$

c

As highlighted in Figure 1, the length of the ID header is `0x5` which is expressed in 32 bit words. As such the header length is 160 bits or 20 bytes. Note that this is enough space to include the destination IP address, leaving no room for options.

$\boxed{\textbf{Answer}\text{: 160 bits}}$

d

As highlighted in Figure 1, the data protocol is `0x11` which corresponds to UDP.

$\boxed{\textbf{Answer}\text{: UDP}}$

2

Note that the sequence includes an Ethernet header (which does not include Preamble and SFD). Furthermore, the below sequence of frame is a snapshot of a fragmented datagram in three parts, starting with offset 0 bytes and moving successively up to 2960 bytes. As all the 3 frames have the same identification, we can conclude that they are apart of the same fragmentation sequence. Note, as IP packets can arrive out of sequence, including several fragmented sequences, the ID is the only thing binds a fragmented sequence on the receiver end.

a

Frame 1

```
 ... 48 00 20 00 20 01 ...
```

Figure 2: Frame 1 flag and fragment offset

When expressed in binary

```
00100000 00000000
```

Figure 3: Do not fragment, More fragments, Fragment offset

$\boxed{\textbf{Answer}\text{: More fragments, no fragmentation offset}}$

Frame 2

```
 ... 48 00 20 B9 20 01 ...
```

Figure 4: Frame 2 flag and fragment offset

When expressed in binary

```
00100000 10111001
```

Figure 5: Do not fragment, More fragments, Fragment offset

Answer: More fragments, fragment offset of 1480 bytes

Frame 3

```
... 48 00 01 72 20 01 ...
```

Figure 6: Frame 3 flag and fragment offset

When expressed in binary

```
00000001 01110010
```

Figure 7: Do not fragment, More fragments, Fragment offset

Answer: Fragmented, last fragment, fragment offset of 2960 bytes

b

The Identification filed consists of 2 byte and are represented by the 5th and 6th bytes of the IPv4 header.

Frame 1

```
... DC 48 00 20 ...
```

Figure 8: Frame 1 identification field

Answer: 48 00

Frame 2

```
... DC 48 00 20 ...
```

Figure 9: Frame 2 identification field

Answer: 48 00, same as frame 1

Frame 3

```
... 2C 48 00 01 ...
```

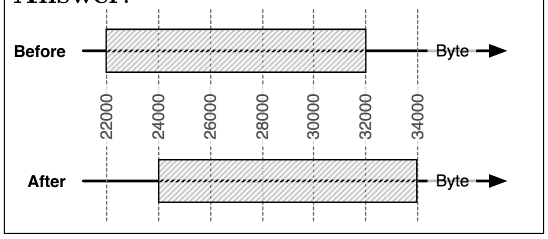
Figure 10: Frame 2 identification field

Answer: 48 00, same as frame 1 & 2

3

In general, TCP deploys cumulative acknowledgements to achieve a reliable transmission. As such, an ACK message from the receiver will tell the sender which sequence byte it i expecting next.

Answer:



4

UPD datagram

The UDP header is 8 bytes long, making the whole datagram with payload, 24 bytes long. Consequently, the utilization is $\frac{16}{24} \approx 66\%$.

Answer: 66%

UPD in IPv4

An IPv4 header without options is 20 bytes long. With the 8 byte UDP header, the total overhead is 28 bytes. Consequently, the utilization is $\frac{16}{28+16} \approx 36\%$.

Answer: 36%

UPD in IPv6

Similarly, IPv6 header without options is 40 bytes long. With the 8 byte UDP header, the total overhead is 48 bytes. Consequently, the utilization is $\frac{16}{48+16} \approx 25\%$.

Answer: 25%

5

TCP packet

The TCP header is 20 bytes long, making the whole packet with payload, 36 bytes long. Consequently, the utilization is $\frac{16}{36} \approx 44\%$.

$\boxed{\textbf{Answer}\text{: } 44\%}$

TCP in IPv4

An IPv4 header without options is 20 bytes long. With the 20 byte TCP header, the total overhead is 56 bytes. Consequently, the utilization is $\frac{16}{40+16} \approx 29\%$.

$\boxed{\textbf{Answer}\text{: } 29\%}$

TCP in IPv6

Similarly, IPv6 header without options is 40 bytes long. With the 20 byte TCP header, the total overhead is 76 bytes. Consequently, the utilization is $\frac{16}{60+16} \approx 21\%$.

$\boxed{\textbf{Answer}\text{: } 21\%}$

6

Table 1 reveals the content of the 8 byte UDP header.

	Hex	Decimal
Source port	`0x0632`	1586
Destination port	`0x000D`	13
Length (bytes)	`0x001C`	28
Checksum	`0xE217`	n/a

Table 1: UDP header breakdown

a

Table 1 reveals that the source port is 1586.

$\boxed{\textbf{Answer}\text{: } 1586}$

b

Table 1 reveals that the destination port is 13, which corresponds to Daytime Protocol.

$\boxed{\textbf{Answer}\text{: 13}}$

c

Table 1 reveals that the total length of the datagram is 28 bytes.

$\boxed{\textbf{Answer}\text{: 28 bytes}}$

d

Subtracting the header size (8) from the total length reveals that the payload is $28 - 8 = 20$ bytes.

$\boxed{\textbf{Answer}\text{: 20 bytes}}$

e

The source port is not generally use, but the destination port is. Moreover, as the source port is irrelevant to the Server, this implies that the datagram is originating from the Client.

$\boxed{\textbf{Answer}\text{: Client} \rightarrow \text{Server}}$

7

Table 2 reveals the content of the 20 byte TCP header. Note that there is no room for any options.

	Hex	Decimal
Source port	`0x0532`	1330
Destination port	`0x0017`	23
Sequence number	`0x00000001`	1
Acknowledgment number	`0x00000000`	0
Data offset and flags	`0x5002`	n/a
Window size (bytes)	`0x07FF`	2047
Checksum	`0x0000`	n/a
Urgent pointer	`0x0000`	n/a

Table 2: TCP header breakdown

a

Table 2 reveals that the source port is 1330.

$\boxed{\textbf{Answer}\text{: 1586}}$

b

Table 2 reveals that the destination port is 23, which corresponds to Telnet.

$\boxed{\textbf{Answer}\text{: 23}}$

c

Table 2 reveals that the sequence number is 1.

$\boxed{\textbf{Answer}\text{: 1}}$

d

Table 2 reveals that the ACK number is 0.

$\boxed{\textbf{Answer}\text{: 0}}$

e

From counting bytes: the size of the header is 20 bytes.

$\boxed{\textbf{Answer}\text{: 20 bytes}}$

f

As reveals in the answer to question 7 b) the destination port is 23 which implies that the packet is carrying Telnet data.

$\boxed{\textbf{Answer}\text{: Telnet}}$

g

Table 2 reveals that the windows size is 2047 bytes.

$\boxed{\textbf{Answer}\text{: 2047 bytes}}$

8

We need to start by identifying the TCP packet. We have previously learnt that, without preamble and SFD, an Ethernet header is 14 bytes long. Figure 11 reveals what is left after the Ethernet header has been removed.

```
00 00 0c 07 ac 01 00 08 74 41 af a7 08 00 45 00
00 30 88 14 40 00 80 06 d5 dc 82 eb 12 bd 82 eb
84 43 09 93 00 17 f2 d2 7a 29 00 00 00 00 70 02
40 00 2f a2 00 00 02 04 05 b4 01 01 04 02
```

Figure 11: Ethernet frame elimination

Furthermore, as illustrated in Figure 12 the subsequent 20 byte IPv4 header reveals the source and destination IP addresses: 130.235.18.189, 130.235.132.67 respectively.

```
00 00 0c 07 ac 01 00 08 74 41 af a7 08 00 45 00
00 30 88 14 40 00 80 06 d5 dc 82 eb 12 bd 82 eb
84 43 09 93 00 17 f2 d2 7a 29 00 00 00 00 70 02
40 00 2f a2 00 00 02 04 05 b4 01 01 04 02
```

Figure 12: IPv4 frame elimination
IP version, Source address, Destination address

We are finally left with the actual TCP content. See Figure 13.

```
09 93 00 17 f2 d2 7a 29 00 00 00 00 70 02
40 00 2f a2 00 00 02 04 05 b4 01 01 04 02
```

Figure 13: TCP packet dissection

The header in Figure 13 has the following field content:

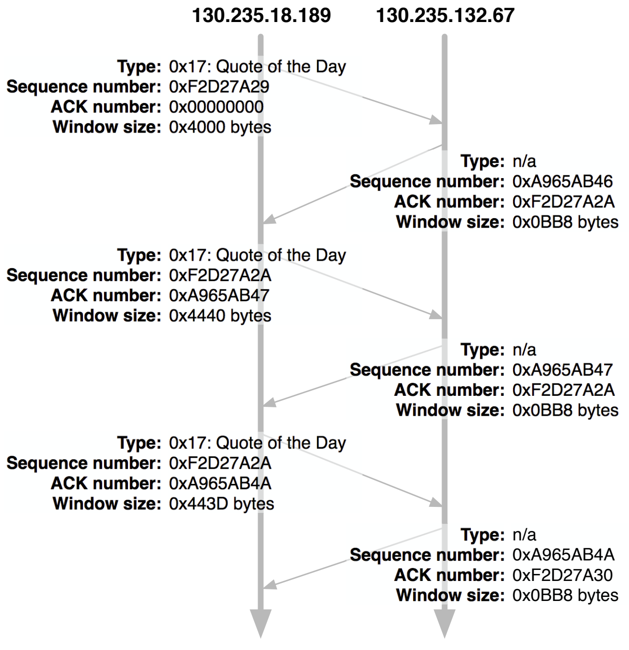
	Hex	Decimal
Source port	0x0993	2451
Destination port	0x0017	23
Sequence number	0xF2D27A29	4073880105
Acknowledgment number	0x00000000	0
Window size (bytes)	0x4000	16384

Table 3: TCP header breakdown

Dissecting each frame in the same manner as above reveals that we are observing a

”Quote of the Day” conversation between 130.235.18.189 and 130.235.132.67.

Answer:


130.235.18.189
130.235.132.67
Type: 0x17: Quote of the Day
Sequence number: 0xF2D27A29
ACK number: 0x00000000
Window size: 0x4000 bytes
Type: n/a
Sequence number: 0xA965AB46
ACK number: 0xF2D27A2A
Window size: 0x0BB8 bytes
Type: 0x17: Quote of the Day
Sequence number: 0xF2D27A2A
ACK number: 0xA965AB47
Window size: 0x4440 bytes
Type: n/a
Sequence number: 0xA965AB47
ACK number: 0xF2D27A2A
Window size: 0x0BB8 bytes
Type: 0x17: Quote of the Day
Sequence number: 0xF2D27A2A
ACK number: 0xA965AB4A
Window size: 0x443D bytes
Type: n/a
Sequence number: 0xA965AB4A
ACK number: 0xF2D27A30
Window size: 0x0BB8 bytes