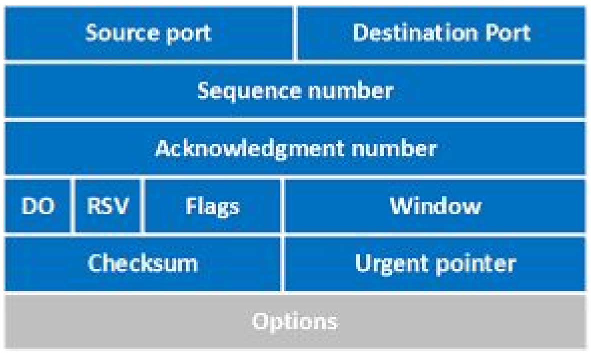
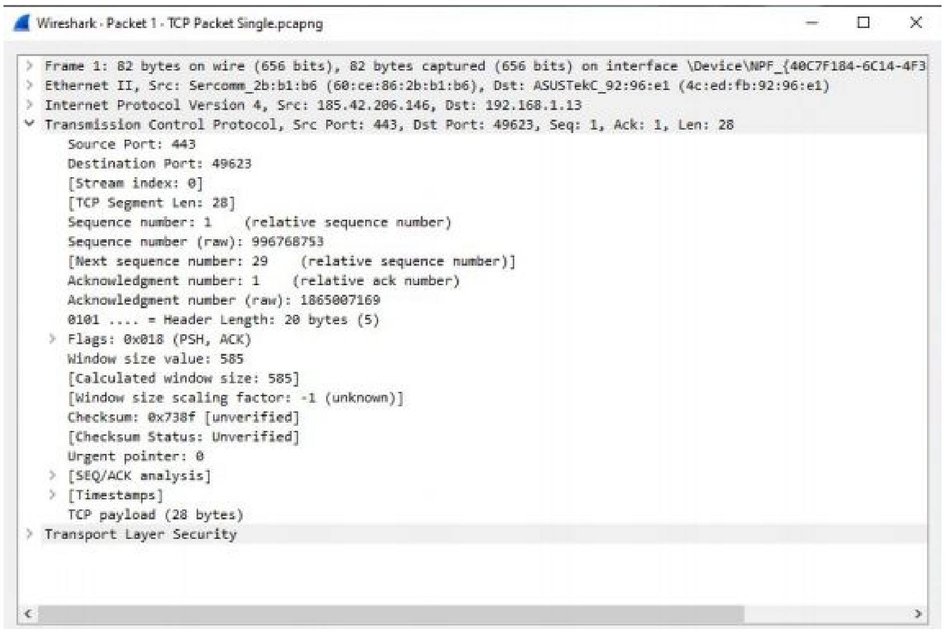
Lab 5

Question 1:



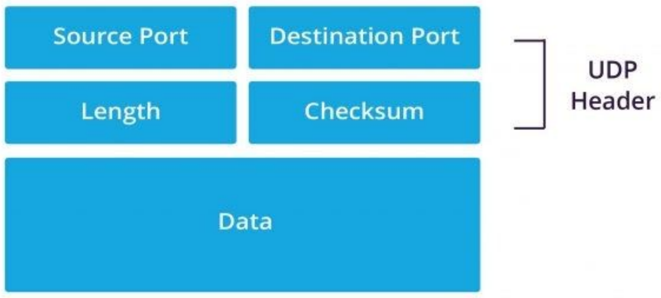


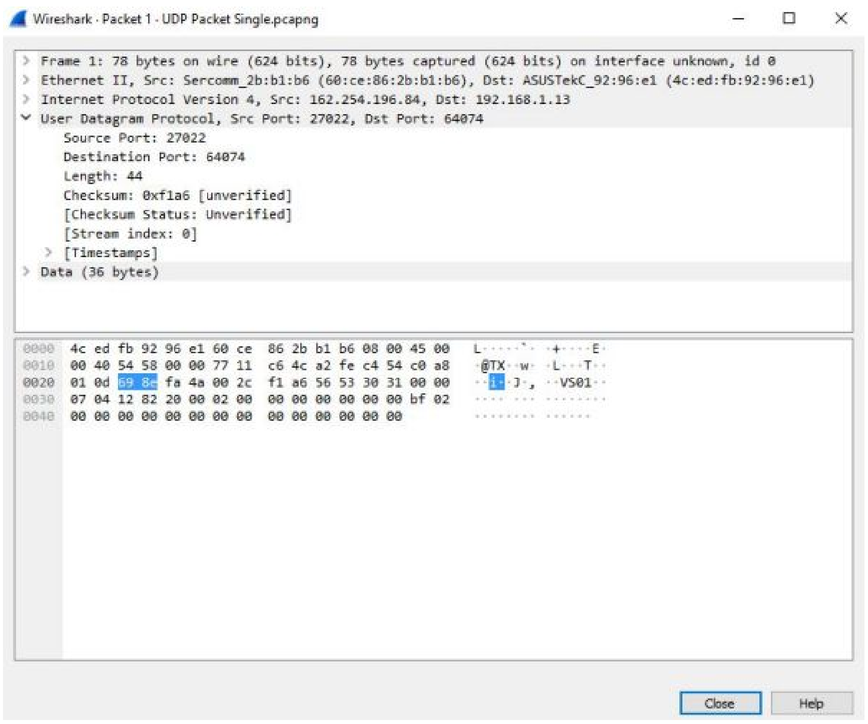
```
> Frame 1: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface \Device\NPF_{40C7F184-6C14-4F3
> Ethernet II, Src: Sercomm_2b:b1:b6 (60:ce:86:2b:b1:b6), Dst: ASUSTekC_92:96:e1 (4c:ed:fb:92:96:e1)
> Internet Protocol Version 4, Src: 185.42.206.146, Dst: 192.168.1.13
v Transmission Control Protocol, Src Port: 443, Dst Port: 49623, Seq: 1, Ack: 1, Len: 28
    Source Port: 443
    Destination Port: 49623
    [Stream index: 0]
    [TCP Segment Len: 28]
    Sequence number: 1    (relative sequence number)
    Sequence number (raw): 996768753
    [Next sequence number: 29    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Acknowledgment number (raw): 1865007169
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
    Window size value: 585
    [Calculated window size: 585]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x738f [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (28 bytes)
> Transport Layer Security
```

Source port: this is a 16 bit field that specifies the port number of the sender. The source port is 443 in this TCP packet capture.
Destination port: this is a 16 bit field that specifies the port number of the receiver.The destination port in this packet is 49623.
Sequence number: the sequence number is a 32 bit field that indicates how much data is sent during the TCP session. When you establish a new TCP connection (3 way handshake) then the initial sequence number is a random 32 bit value. The receiver will use this sequence number and send back an acknowledgment. Protocol analyzers like wireshark will often use a relative sequence number of 0 since it's easier to read than some high random number. The Sequence Number is 1 in this packet.
Acknowledgment number: this 32 bit field is used by the receiver to request the next TCP segment. This value will be the sequence number incremented by 1. In this case this is true as the ACK flag is set to 1 meaning the next expected sequence value is 29.
DO (Data Offset): this is the 4 bit data offset field, also known as the header length. It indicates the length of the TCP header so that we know where the actual data begins. In this packet it is 20 bytes.
RSV: these are 3 bits for the reserved field. They are unused and are always set to 0.

Flags: there are 9 bits for flags, we also call them control bits. We use them to establish connections, send data and terminate connections:
URG: urgent pointer. When this bit is set, the data should be treated as priority over other data.
ACK: used for the acknowledgment.
PSH: this is the push function. This tells an application that the data should be transmitted immediately and that we don't want to wait to fill the entire TCP segment.
RST: this resets the connection, when you receive this you have to terminate the connection right away. This is only used when there are unrecoverable errors and it's not a normal way to finish the TCP connection. This is not true in this situation.
SYN: we use this for the initial three way handshake and it's used to set the initial sequence number. In this packet it is set to true.
FIN: this finish bit is used to end the TCP connection. TCP is full duplex so both parties will have to use the FIN bit to end the connection. This is the normal method for how we end a connection.
Window: the 16 bit window field specifies how many bytes the receiver is willing to receive. It is used so the receiver can tell the sender that it would like to receive more data than what it is currently receiving. It does so by specifying the number of bytes beyond the sequence number in the acknowledgment field. In this packet it is 585.
Checksum: 16 bits are used for a checksum to check if the TCP header is OK or not.
Urgent pointer: these 16 bits are used when the URG bit has been set, the urgent pointer is used to indicate where the urgent data ends. The value of the checksum in this packet is 0x738f.
Options: this field is optional and can be anywhere between 0 and 320 bits.

Question 2:





```
> Frame 1: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface unknown, id 0
> Ethernet II, Src: Sercomm_2b:b1:b6 (60:ce:86:2b:b1:b6), Dst: ASUSTekC_92:96:e1 (4c:ed:fb:92:96:e1)
> Internet Protocol Version 4, Src: 162.254.196.84, Dst: 192.168.1.13
v User Datagram Protocol, Src Port: 27022, Dst Port: 64074
    Source Port: 27022
    Destination Port: 64074
    Length: 44
    Checksum: 0xf1a6 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
  > [Timestamps]
> Data (36 bytes)

0000  4c ed fb 92 96 e1 60 ce  86 2b b1 b6 08 00 45 00   L·····`·  ·+····E·
0010  00 40 54 58 00 00 77 11  c6 4c a2 fe c4 54 c0 a8   ·@TX··w·  ·L···T··
0020  01 0d 69 8e fa 4a 00 2c  f1 a6 56 53 30 31 00 00   ··i··J·,  ··VS01··
0030  07 04 12 82 20 00 02 00  00 00 00 00 00 00 bf 02   ········  ········
0040  00 00 00 00 00 00 00 00  00 00 00 00 00 00         ········  ······
```

Source port: The port of the device sending the data. This field can be set to zero if the destination computer doesn't need to reply to the sender. In this packet it is 27022
Destination port: The port of the device receiving the data. UDP port numbers can be between 0 and 65,535. The port id in this packet is 64074
Length: Specifies the number of bytes comprising the UDP header and the UDP payload data. The limit for the UDP length field is determined by the underlying IP protocol used to transmit the data. The length is 44 bytes in this packet.
Checksum: The checksum allows the receiving device to verify the integrity of the packet header and payload. It is optional in IPv4 but was made mandatory in IPv6. In this case the value of the checksum is 0xf1a6.

Question 3:
a2fe + c454 = 16752 16752 + c0a8 = 227FA 227FA + 010d = 22907 22907 + 0011 = 22918 22918 + 002c = 22944 22944 + 698e = 292D2 292D2 + fa4a = 38D1C 38D1C + 002c = 38D48 38D48 + 5653 = 3E39B 3E39B + 3031 = 413CC 413CC + 0000 = 413CC 413CC +

0704 = 41AD0 41AD0 + 1282 = 42D52 42D52 + 2000 = 44D52 44D52 + 0200 = 44F52 44F52 + 0000 = 44F52 44F52 + 0000 = 44F52 44F52 + 0000 = 44F52 44F52 + bf02 = 50E54
1010000111001010100 Add upper with lower
= 0E59
To Binary 0000111001011001 One’s Complement
1111000110100110
Convert to Hex F1a6
<u>(*Q3 COPIED FROM KRYSTIAN***) I couldn’t figure it out myself</u>**

Question 4: I don’t know how to tell the difference between normal packets to a steaming application packet.

Question 5:

1. Client sends segments with SYN because it wants to start connection with the server. This segment informs the server that the client is likely to start communication and what sequence number it starts segments with.
2. Server will then respond to the client request with SYN - ACK signal bits. ACK is the received segment and the SYN is the sequence number it starts segments with.
3. The client acknowledges the response to the server and both now create a connection in which they will start to transfer data.

Question 6:

1. Client sends a FIN and ACK flag to inform the server that its trying to close the connection
2. When the server receives the FIN flag it sends back the ACK flag back to the client
3. Client then enters the second waiting face to receive FIN flag from the server side.
4. Server sends FIN bit segments to the client after some time when the server sends the ACK segment because the client is closing it back.
5. The client would then send a final ACK to inform the server that it should release all the data as the client has fully closed connection with them.

BONUS:

1: How does TCP work?

Ans: TCP uses a three-way handshake to establish a connection between client and server. It uses SYN, ACK and FIN flags (1 bit) for connecting two endpoints. After the establishment of the connection, data is transferred sequentially. If there is any loss of packet, it retransmits data.

<REF>
https://allabouttesting.org/top-10-interview-questions-tcpudp/ Q3

2:Is UDP better than TCP?

Ans: Both protocols are used for different purposes. If the user wants error-free and guarantees to deliver data, TCP is the choice. If the user wants fast transmission of data and little loss of data is not a problem, UDP is the choice.

<REF>
https://allabouttesting.org/top-10-interview-questions-tcpudp/ Q6