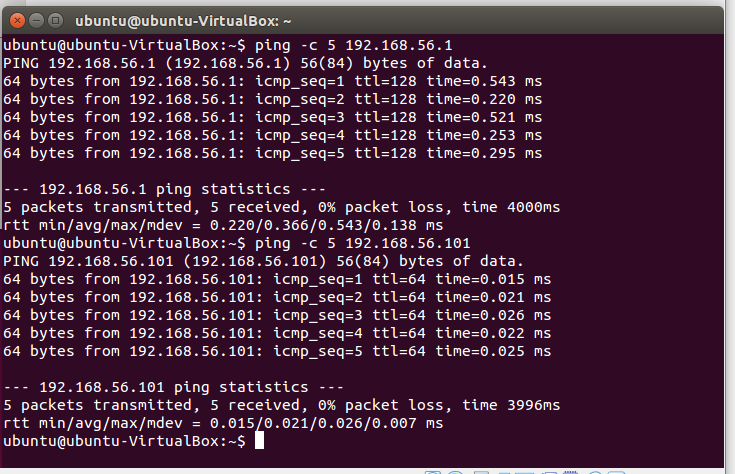
|  |
| --- |
|  |
| Assignment1 |
|  |

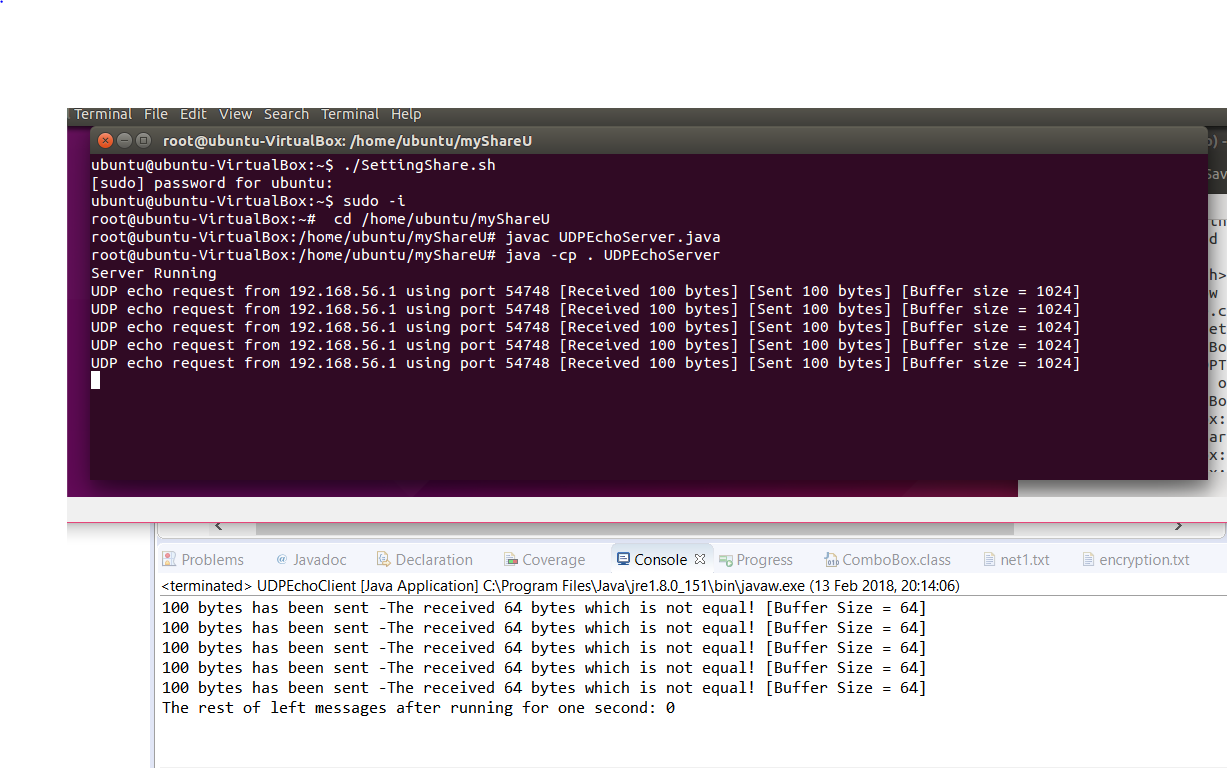
|  |
| --- |
| Aya Kathem  2018-02-12 |

1. Problem

Pinging my host machine 192.169.56.1, the longest round trip is 543ms and the shortest is 220ms.



2. An image of server side and client. The image shows the amount of sending data and receiving data during one seccond from host to the virtual machine. On the server side we can see, the IP address of host machine and the used port, in addition, the size of the message and buffer size.

The client side shows the size of the sent message and the received buffer size 64. It shows also if there is any message left (throw away).

The program in the client side handle several errors can provide by the client.

• All the four arguments should be implemented by the client if not it shows an error message.

• Handling wrong filled IP address if greater than 255 or less than 0 if the IP does not contain four dots “. “.

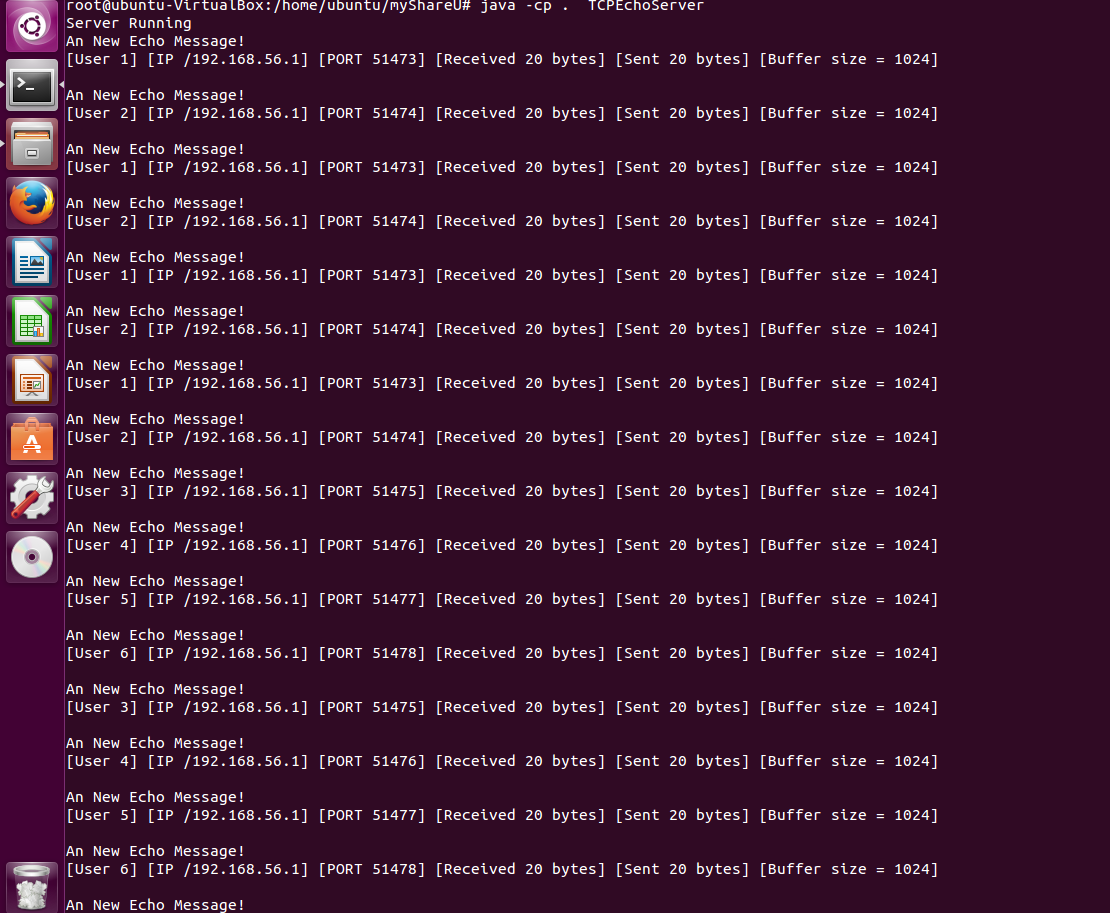
• Handel the port number, if the port number is not an integer, and when the filled number greater the 65535 or less the 1.

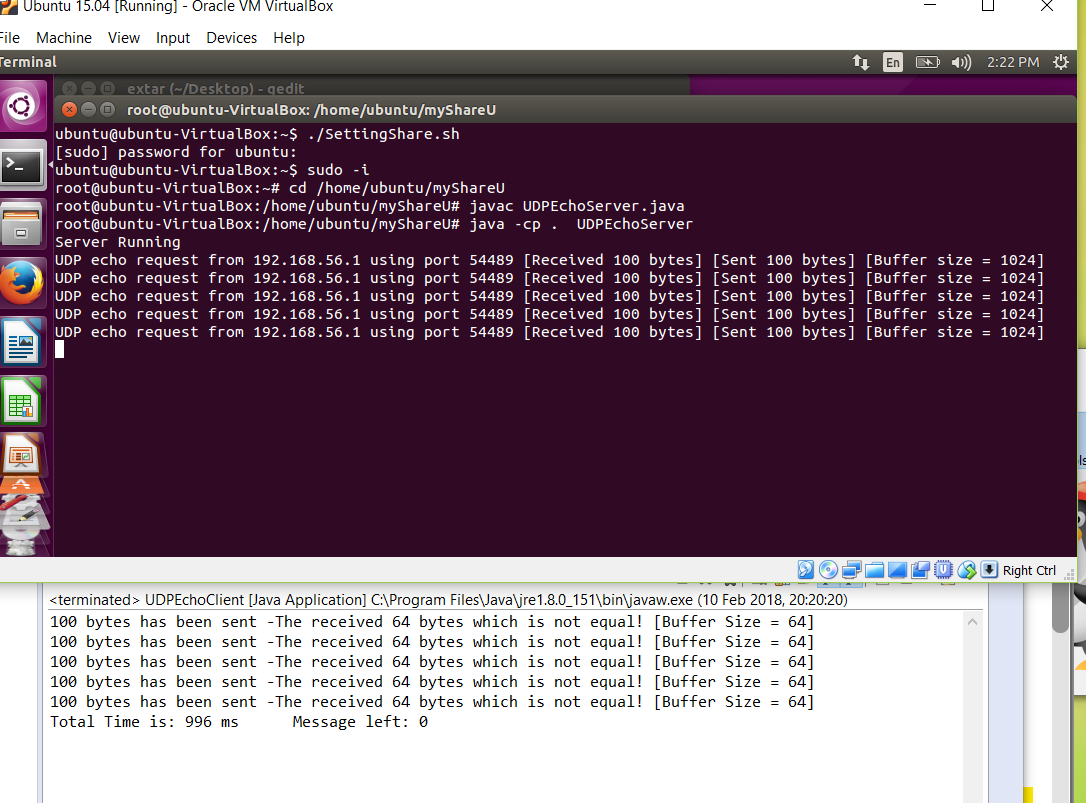
• Handel if the transfer rate is less than 0

• The UDP allowed the 65507 packet size, the program handles the sent message if it is greater than the allowed packet size by UDP and if the message is empty.

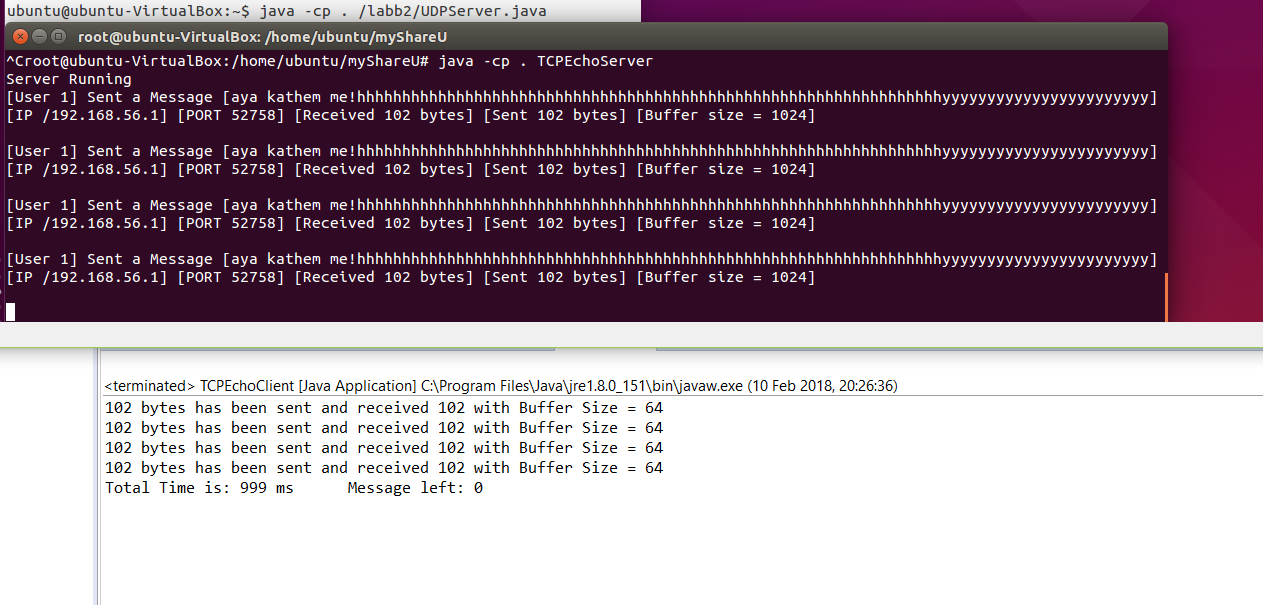
• Finally, handle buffer size cause of memory exception.

3.

The screenshot belongs to TCP server. There are several clients request the server at the same time. The TCP server supports multiple client connections. The client side is also handling the error that can occur as well as UDP client.

**UDP:**

**TCP**



What is the difference and why?

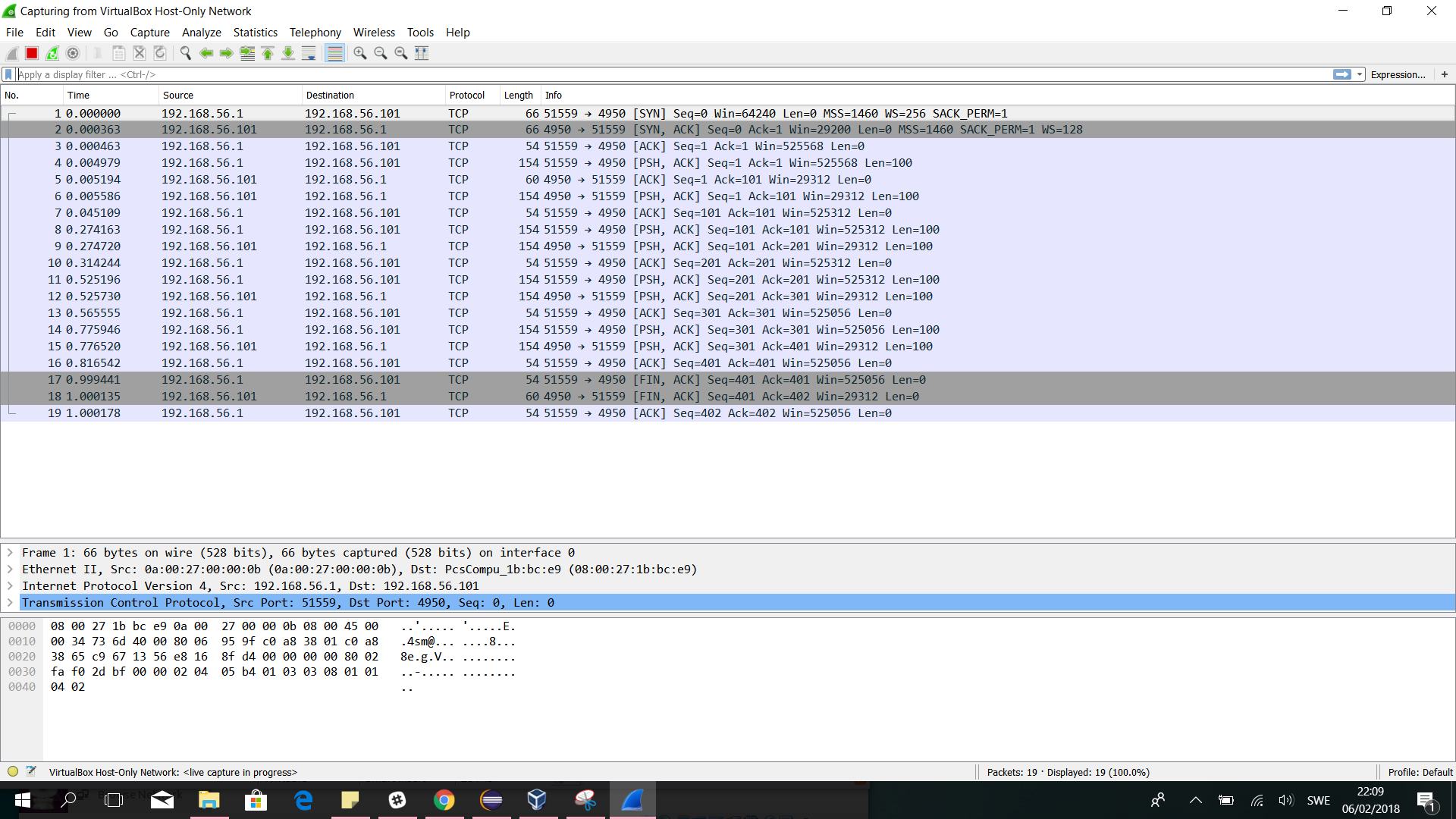
The difference between is the TCP client receive the same amount of sent data. In compares, UDP the client receives 64 of 100 bytes. The reason is TCP read data as a stream. Every iteration in TCP creates a new buffer and add the message into the same string. In comparison, UDP use datagram packets to send data individually, and the packet has limited length specify by the buffer size. In this case, as it shows in the image we get only 64 of 100 which is the length of the buffer, UDP drop the rest of the bytes and we can’t get them back.

4.

. SYN, when the client the start to communicate with TCP server, the client and server start to exchange a series of messages which known as SYN [Synchronize]. When the process start the client send a SYN message to the server and server response with SYN-ACK acknowledge back to the client.

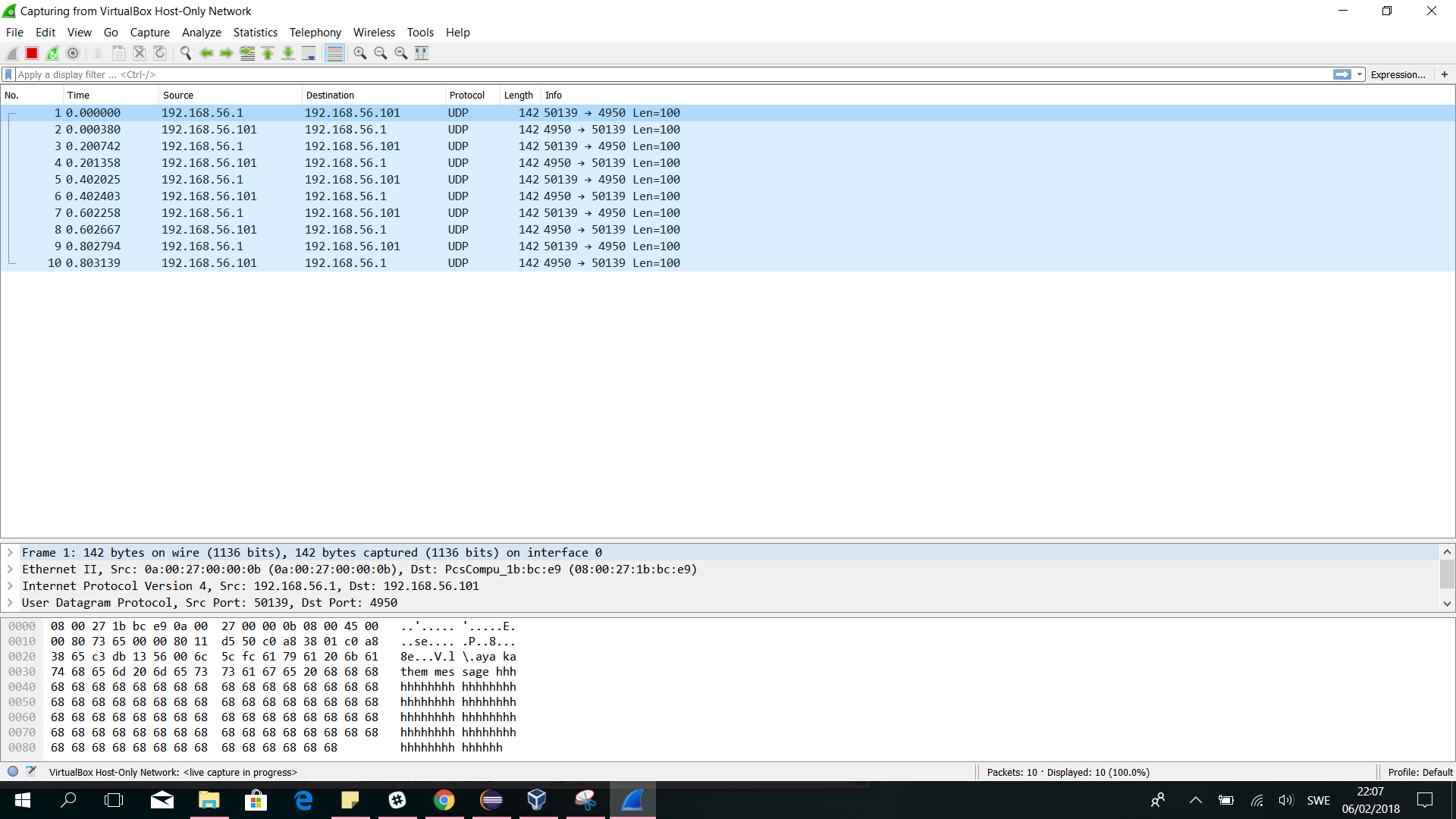
. ACK, acknowledges received data. When the connection is established, all packets sent by either side will contain an ACK.

. PSH, push flag, it pushing data out immediately without waiting to fill whole buffer size which in this case is 1024.



TCP.

1. The client starts with sending a SYN to the server to start communicate, The sequence number = 0.
2. A replied came from the server with SYN-ACK message, the seq = 0 and the ACK =1.
3. The connection established by sending ACK =1 by the client.
4. The first message sent by the client with 100 lengths. In this case the length of the message is less than the buffer size, so PSH send the message without waiting until buffer be full.
5. Then the server has received the message and sent back ACK = 101.
6. The server set the message back to the client with length 100 and ACK 101 and the push flag is sets.
7. The client gets the message and set back ACK = 101.
8. Again the client sent a new message with length 100 to the server, ACK = 101 and PSH flag sets.
9. Then the server has received the message and sent back ACK = 201.
10. The client gets the message and set back ACK = 201.
11. the client sent a new message with length 100 to the server, ACK = 201 and PSH flag sets.
12. The server received the message and sent ACK=301.
13. The client gets the message and set back ACK = 301.
14. The client sent a new message with length 100 to the server, ACK = 301 and PSH flag sets.
15. The server received the message and sent back ACK=401.
16. The client gets the message and set back ACK =401.
17. The client sent a Fin (finish) and ACK = 401.
18. The server replied with FIN and ACK 402.
19. Finally, the client received the message from the server and sent back ACK = 402.



UDP

1. The client sent a message to UDP server with length 100, the server gets (message) 100 byte + 42(header)byte = 142.

2. The server replied the message back to the client

3. In step 3 – 10 the same process happens again and again.

What is the difference between TCP and UDP?

• Reliability, TCP is connection-oriented protocol, all sent packet should deliver except if the connections fail. In case the connection lost, the TCP server will request the lost part. On the other hand UDP, UDP is connectionless protocol, the data could be lost while sending it and UDP can’t get it back.

• Ordered, TCP takes care of order even if the message sent at the same time you will get them in order. In comparison, UDP doesn’t care about the order which means it is faster than TCP.

• Streaming, TCP use stream to read data, In addition, there may be multiple packets per reading call.

• UDP use datagram and its support only one packet per one read call.

• UDP is lightweight internet protocol because it has a little work to do. In comparison, TCP is heavyweight protocol, the reason is TCP has a lot of work to such as taking care of the order.