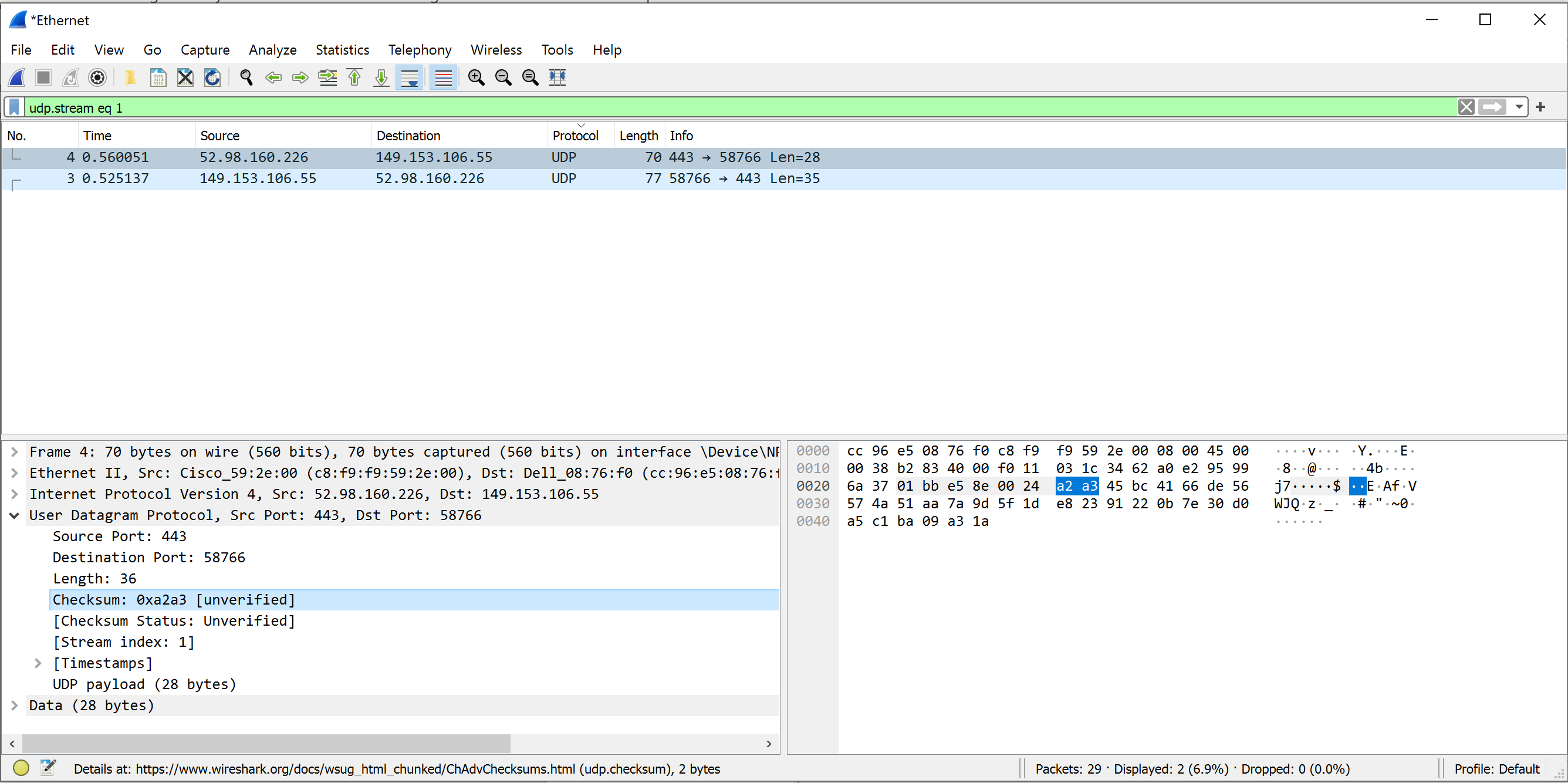
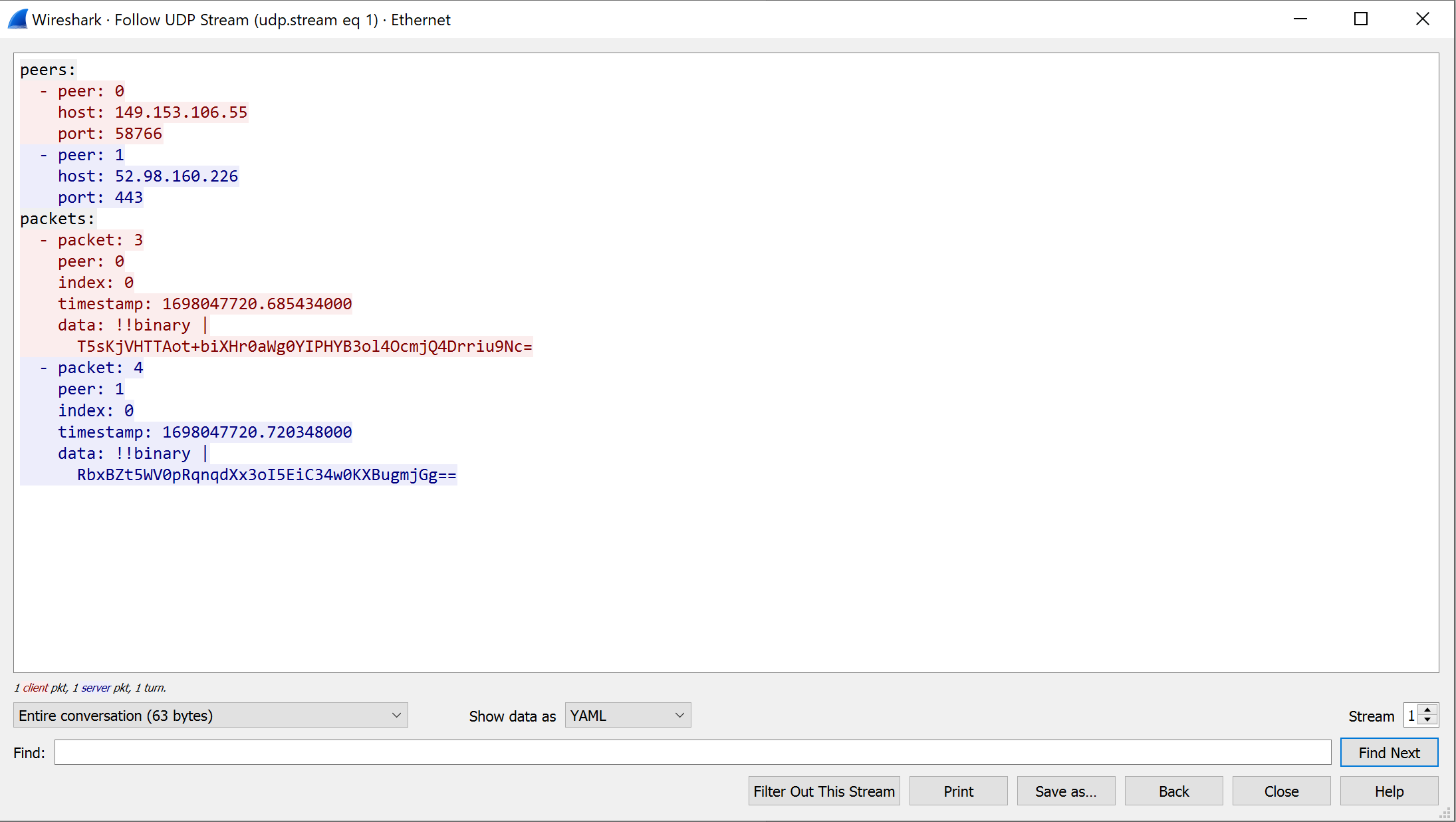
1. Capture a UDP packet, verify the checksum using **16-bit One's Complement Sum** algorithm.  
     
     
     
     
     
   Checksum: 0xa2a3 [unverified] a2 a3 ـ ـ
2. What's TCP 3-Way Handshake? Draw a diagram to illustrate the process using real packets captured in a TCP session. Fill in the values of some key fields of the packets.  
     
   TCP 3-Way Handshake:

The TCP 3-Way Handshake is the process that establishes a TCP connection between two hosts. It involves three steps:

**SYN**: The client initiates the connection by sending a TCP packet with the SYN (synchronize) flag set. The initial sequence number (ISN) is also included.  
**SYN-ACK**: The server receives the SYN packet, acknowledges it by sending a packet with the SYN and ACK flags set. It also generates its own ISN.   
**ACK**: The client receives the SYN-ACK packet, acknowledges it, and the connection is established.  
  
Diagram:  
  
Client Server

[SYN] -------> [SYN, ACK] ------->

Source: Port X Source: Port Y  
Dest: Port Z Dest: Port X

Seq: 1000 Seq: 2000

Ack: 0 Ack: 1001

<------- [ACK] <------- [ACK]

Source: Port X Source: Port Y

Dest: Port Z Dest: Port X

Seq: 1001 Seq: 2001

Ack: 2001 Ack: 1002  
  
The client (left) initiates the connection by sending a packet with the SYN flag set.  
The server (right) responds with a packet that has both the SYN and ACK flags set, acknowledging the client's request.  
The client acknowledges the server's response with a packet containing the ACK flag.

1. What's TCP 4-Way teardown? Draw a diagram to illustrate the process using real packets captured in a TCP session. Fill in the values of some key fields of the packets.  
     
   TCP 4-Way Teardown:

The TCP 4-Way Teardown is the process that terminates a TCP connection. It involves four steps:

**FIN (Client)**: The client initiates the teardown by sending a packet with the FIN (finish) flag set, indicating it wants to close the connection.   
**ACK (Server)**: The server acknowledges the FIN with an ACK.   
**FIN (Server)**: Once the server is ready to close the connection, it sends its own FIN packet.   
**ACK (Client)**: The client acknowledges the server's FIN. The connection is now fully closed.

Diagram:  
Client Server

[FIN] -------> [ACK] ------->

Source: Port X Source: Port Y

Dest: Port Z Dest: Port X

Seq: 3000 Seq: 4000

Ack: 2001 Ack: 3001

<------- [ACK] <------- [FIN] ------->

Source: Port X Source: Port Y

Dest: Port Z Dest: Port X

Seq: 2001 Seq: 3001

Ack: 3001 Ack: 4001

<------- [FIN] -------> <------- [ACK]

Source: Port X Source: Port Y

Dest: Port Z Dest: Port X

Seq: 2002 Seq: 4001

Ack: 3001 Ack: 3002

[ACK] ------->

Source: Port X

Dest: Port Z

Seq: 3002

Ack: 4002  
  
The client initiates the teardown by sending a FIN (Finish) packet, indicating that it has no more data to send. The server acknowledges this with an ACK (Acknowledgment).

The server, after processing its remaining data, also initiates the teardown by sending a FIN packet to the client. The client acknowledges the server's FIN with an ACK.

The client, after processing any remaining data, acknowledges the server's FIN.

Finally, the server acknowledges the client's ACK, and the connection is fully closed.

Bonus:

Find two interview questions about TCP, and provide the answer. please provide the reference.

**Bonus**:

Here are two interview questions about TCP along with their answers:

**Question** 1: What is the Three-Way Handshake in TCP, and why is it important?

**Answer** 1: The Three-Way Handshake is a fundamental process in TCP (Transmission Control Protocol) used to establish a reliable connection between a client and a server. It consists of three steps:

* SYN (Synchronize): The client sends a TCP packet with the SYN (Synchronize) flag set to the server, indicating its desire to establish a connection and specifying an initial sequence number.
* SYN-ACK (Synchronize-Acknowledge): The server receives the SYN packet, acknowledges it with a packet containing both the SYN and ACK (Acknowledgment) flags set, and also specifies its initial sequence number.
* ACK (Acknowledgment): Finally, the client acknowledges the server's response by sending a packet with the ACK flag set. The connection is now established.

The Three-Way Handshake is important because it ensures that both the client and server are aware of each other's willingness to communicate and agree on initial sequence numbers. This process helps establish a reliable and synchronized connection, preventing data corruption and ensuring data delivery.

Reference: Comer, D. E. (2015). Internetworking with TCP/IP, Volume 1: Principles, Protocols, and Architecture. Pearson.

**Question** 2: What is TCP congestion control, and why is it crucial for network performance?

**Answer** 2: TCP congestion control is a mechanism that regulates the rate at which data is sent over a network to prevent network congestion, packet loss, and degradation of service quality. It is essential for network performance because it addresses the following key issues:

* Fairness: TCP congestion control ensures that network resources are shared fairly among all connected hosts. It prevents any single host from overwhelming the network and causing poor performance for others.
* Congestion Avoidance: By monitoring the network's state and adjusting its sending rate accordingly, TCP congestion control helps prevent congestion before it occurs. This proactive approach reduces the likelihood of packet loss and retransmissions.
* Adaptability: TCP congestion control algorithms dynamically respond to network conditions, such as increased traffic or congestion. They slow down the sending rate when congestion is detected and gradually increase it as the network stabilizes.
* Efficiency: It maximizes the utilization of network resources while minimizing the chances of network congestion, leading to optimal network performance.

TCP congestion control algorithms like TCP Reno, TCP Cubic, and TCP NewReno play a crucial role in maintaining the stability and efficiency of the Internet.

Reference: Floyd, S., & Jacobson, V. (1993). Random Early Detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking, 1(4), 397-413.