

מגישים: עומר ארד, אחמד ג'ורבן

הסדר: חלק שני של חלק א, לאחר מכן חלק ראשון של חלק א, ואז חלק ב.

V1:

The command to run server.py is `python2 server.py 12345`

The command to run client.py is `python2 client.py 0.0.0.0 12345`

```
2
3 import socket,sys
4
5 TCP_IP = sys.argv[1]
6 TCP_PORT = int(sys.argv[2])
7 BUFFER_SIZE = 1024
8 MESSAGE = "Hello, world!"
9
10 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11 s.connect((TCP_IP, TCP_PORT))
12 s.send(MESSAGE)
13 data = s.recv(BUFFER_SIZE)
14 s.close()
15
16 print "received data:", data
17
18 |
```

This code creates a socket and connects to a server at the specified IP address and port number passed as arguments to the script. It then sends the message "Hello, World!" to the server and receives a response with a buffer size of 1024 bytes. The response data is then printed to the screen and the socket is closed.

```
1 import socket,sys
2
3 TCP_IP = '0.0.0.0'
4 TCP_PORT = int(sys.argv[1])
5 BUFFER_SIZE = 1024
6
7 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8 s.bind((TCP_IP, TCP_PORT))
9 s.listen(1)
10
11 while True:
12     conn, addr = s.accept()
13     print 'New connection from:', addr
14     while True:
15         data = conn.recv(BUFFER_SIZE)
16         if not data: break
17         print "received:", data
18         conn.send(data.upper())
19     conn.close()
20
21
```

The code creates a simple TCP server that listens for incoming connections on the IP address 0.0.0.0 and a port number specified as a command-line argument. When a new connection is accepted, the server receives data from the client in chunks of size BUFFER_SIZE, prints the received data, and then echoes the data back to the client in uppercase. The server continues to listen for new connections and handle them in the same way.

Time	Source	Destination	Protocol	Length	Info
1.0.000000000	127.0.0.1	127.0.0.1	TCP	76	51408 → 12345 [SYN] Seq=3178576893 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=0 WS=128
2.0.000013065	127.0.0.1	127.0.0.1	TCP	76	12345 → 51408 [SYN, ACK] Seq=2842456210 Ack=3178576894 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=2218082649
3.0.000022691	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=0 TSval=2218082649 TSecr=2218082649
4.0.000032801	127.0.0.1	127.0.0.1	TCP	81	51408 → 12345 [PSH, ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=13 TSval=2218082655 TSecr=2218082649
5.0.0000332619	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=0 TSval=2218082655 TSecr=2218082655
6.0.0000339473	127.0.0.1	127.0.0.1	TCP	81	12345 → 51408 [PSH, ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=13 TSval=2218082658 TSecr=2218082655
7.0.014198672	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082658 TSecr=2218082658
8.0.017329429	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [FIN, ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082666 TSecr=2218082658
9.0.018059515	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [FIN, ACK] Seq=2842456224 Ack=3178576908 Win=65536 Len=0 TSval=2218082667 TSecr=2218082666
10.0.018124810	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576908 Ack=2842456225 Win=65536 Len=0 TSval=2218082667 TSecr=2218082667

* Frame 1: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
 * Linux cooked capture v1
 * Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 * Transmission Control Protocol, Src Port: 51408, Dst Port: 12345, Seq: 3178576893, Len: 0

Source Port: 51408
 Destination Port: 12345
 [Stream index: 0]
 [Conversation completeness: Complete, WITH_DATA (31)]
 [TCP Segment Len: 0]
 Sequence Number: 3178576893
 [Next Sequence Number: 3178576894]
 Acknowledgment Number: 0
 Acknowledgment number (raw): 0
 1010 = Header Length: 40 bytes (10)
 * **Flags: 0x002 (SYN)**
 Window: 65495
 [Calculated window size: 65495]
 Checksum: 0xfe30 [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 * Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
 * [Timestamps]

0000	00 00 03 04 00 00 00 00 00 00 00 00 00 00 00 00
0010	45 00 00 3c a9 6f 40 00 40 00 93 4a 7f 00 00 01	E--< 00 0: J
0020	7f 00 00 01 c8 d0 30 39 bd 75 3b fd 00 00 00 0009 u:
0030	a0 02 ff d7 fe 30 00 00 02 04 ff d7 04 02 00 0a0-:

1.This is a SYN packet (SYN flag) and it's the first packet in the handshake and is used to initiate the connection. It contains a sequence number which is: 3178576893 that identifies the starting point for the sequence of data packets that will be exchanged over the connection.

2.The Source Port field which is the client port: 51408 that the OS chose it. indicates the port number of the host that sent the packet which as we said above that the client is asking for a SYN packet which is explained above.

3. The Destination Port field indicates the port number of the host that the packet is being sent to, in this case it's the server port which is: 12345 (chosen by me in the command line) and here the server should be receiving the SYN packet

4. here we should keep in mind that the sequence number is used by the receiver to determine which packets have been received and which ones are missing, based on the ACK number in the received packets (we will see in the next picture)

****sequence number here is : 3178576893.**

1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	51408 → 12345 [SYN, Seq=3178576893 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=0 WS=128
2	0.000013065	127.0.0.1	127.0.0.1	TCP	76	12345 → 51408 [SYN, ACK] Seq=2842456210 Ack=3178576894 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=2218082649
3	0.000022691	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=0 TSval=2218082649 TSecr=2218082649
4	0.000328001	127.0.0.1	127.0.0.1	TCP	81	51408 → 12345 [PSH, ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=13 TSval=2218082655 TSecr=2218082649
5	0.000332619	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=0 TSval=2218082655 TSecr=2218082655
6	0.000339473	127.0.0.1	127.0.0.1	TCP	81	12345 → 51408 [PSH, ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=13 TSval=2218082658 TSecr=2218082655
7	0.014198672	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082658 TSecr=2218082658
8	0.017329429	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [FIN, ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082666 TSecr=2218082658
9	0.018058515	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [FIN, ACK] Seq=2842456224 Ack=3178576908 Win=65536 Len=0 TSval=2218082667 TSecr=2218082666
10	0.018124810	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576908 Ack=2842456225 Win=65536 Len=0 TSval=2218082667 TSecr=2218082667


```

Frame 2: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 12345, Dst Port: 51408, Seq: 2842456210, Ack: 3178576894, Len: 0
  Source Port: 12345
  Destination Port: 51408
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 2842456210
  [Next Sequence Number: 2842456211]
  Acknowledgment Number: 3178576894
  1010 .... = Header Length: 40 bytes (10)
  Flags: 0x012 [SYN, ACK]
  Window: 65483
  [Calculated window size: 65483]
  Checksum: 0xfe30 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  Options: (28 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
  [Timestamps]
  [SEQ/ACK analysis]

```

The SYN-ACK packet is the second packet in the handshake and is sent by the server with port: 12345 in response to the SYN packet (we saw above in the first page) from client with port : 51408

Old sequence number is: 3178576893

Ack number: 3178576894

We should pay attention that the ack number is the old sequence number plus 1 and that's because the ACK number is used to acknowledge the receipt of data packets in a TCP connection, and the ACK number is always set to the sequence number of the last received packet plus 1.

Here I'm going to be explaining the transportation in Wireshark :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	51408 → 12345 [SYN, Seq=3178576893 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=0 WS=128
2	0.000013065	127.0.0.1	127.0.0.1	TCP	76	12345 → 51408 [SYN, ACK] Seq=2842456210 Ack=3178576894 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2218082649 TSecr=2218082649
3	0.000022691	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=0 TSval=2218082649 TSecr=2218082649
4	0.000328001	127.0.0.1	127.0.0.1	TCP	81	51408 → 12345 [PSH, ACK] Seq=3178576894 Ack=2842456211 Win=65536 Len=13 TSval=2218082655 TSecr=2218082649
5	0.000332619	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=0 TSval=2218082655 TSecr=2218082655
6	0.000339473	127.0.0.1	127.0.0.1	TCP	81	12345 → 51408 [PSH, ACK] Seq=2842456211 Ack=3178576907 Win=65536 Len=13 TSval=2218082658 TSecr=2218082655
7	0.014198672	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082658 TSecr=2218082658
8	0.017329429	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [FIN, ACK] Seq=3178576907 Ack=2842456224 Win=65536 Len=0 TSval=2218082666 TSecr=2218082658
9	0.018058515	127.0.0.1	127.0.0.1	TCP	68	12345 → 51408 [FIN, ACK] Seq=2842456224 Ack=3178576908 Win=65536 Len=0 TSval=2218082667 TSecr=2218082666
10	0.018124810	127.0.0.1	127.0.0.1	TCP	68	51408 → 12345 [ACK] Seq=3178576908 Ack=2842456225 Win=65536 Len=0 TSval=2218082667 TSecr=2218082667

The first two rows I have explained above but now I'm going to explain what in fact happens during the termination process

In rows 9 and 10 we see that the client and the server both sends [FIN, ACK] flag which indicates that the connection is being closed gracefully by both sides and that's what indeed happening with code it self (both client and server asking for a connection termination)

V2:

The command to run server.py is `python2 server.py 8080`

The command to run client.py is `python2 client.py 0.0.0.0 8080`

Code explanation :

```
import socket,sys

TCP_IP = '0.0.0.0'
TCP_PORT = int(sys.argv[1])
BUFFER_SIZE = 5

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)

while True:
    conn, addr = s.accept()
    print 'New connection from:', addr
    while True:
        data = conn.recv(BUFFER_SIZE)
        if not data: break
        print "received:", data
        conn.send(data.upper())
    conn.close()
```

This server listens on a specified port (given as an argument) for incoming connections. When a connection is received, the server will receive data from the client in small chunks of 5 bytes at a time. The server will then print the received data to the console and send it back to the client in upper case.

```
import socket,sys

TCP_IP = sys.argv[1]
TCP_PORT = int(sys.argv[2])
BUFFER_SIZE = 1024
MESSAGE = "World! Hello, World!"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send(MESSAGE)
data = s.recv(BUFFER_SIZE)
s.close()

print "received data:", data
```

This client that connects to a server at a specified IP address and port, sends a message to the server, and then receives a response from the server. The client then prints the received data to the console and closes the connection.

Wireshark transportation explanation:

Note: we are now focusing on the differences between V2 to V1 here

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	51054 → 8080 [SYN] Seq=2653737783 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=751734223 TSecr=0 WS=128
2	0.000000920	127.0.0.1	127.0.0.1	TCP	76	8080 → 51054 [SYN, ACK] Seq=161266463 Ack=2653737784 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=751734223 TSecr=751734223
3	0.000017841	127.0.0.1	127.0.0.1	TCP	68	51054 → 8080 [ACK] Seq=2653737784 Ack=161266464 Win=65536 Len=0 TSval=751734223 TSecr=751734223
4	0.000446266	127.0.0.1	127.0.0.1	TCP	88	51054 → 8080 [PSH, ACK] Seq=2653737784 Ack=161266464 Win=65536 Len=20 TSval=751734223 TSecr=751734223 [TCP segment of a reas...
5	0.000453910	127.0.0.1	127.0.0.1	TCP	68	8080 → 51054 [ACK] Seq=161266464 Ack=2653737804 Win=65536 Len=0 TSval=751734223 TSecr=751734223
6	0.001112971	127.0.0.1	127.0.0.1	TCP	73	8080 → 51054 [PSH, ACK] Seq=161266464 Ack=2653737804 Win=65536 Len=5 TSval=751734224 TSecr=751734223 [TCP segment of a reas...
7	0.001284582	127.0.0.1	127.0.0.1	TCP	68	51054 → 8080 [ACK] Seq=2653737804 Ack=161266469 Win=65536 Len=0 TSval=751734224 TSecr=751734224
8	0.001288879	127.0.0.1	127.0.0.1	TCP	83	8080 → 51054 [PSH, ACK] Seq=161266469 Ack=2653737804 Win=65536 Len=15 TSval=751734224 TSecr=751734224 [TCP segment of a reas...
9	0.001291184	127.0.0.1	127.0.0.1	TCP	68	51054 → 8080 [ACK] Seq=2653737804 Ack=161266484 Win=65536 Len=0 TSval=751734224 TSecr=751734224
10	0.001318542	127.0.0.1	127.0.0.1	TCP	68	51054 → 8080 [FIN, ACK] Seq=2653737804 Ack=161266484 Win=65536 Len=0 TSval=751734224 TSecr=751734224
11	0.001343359	127.0.0.1	127.0.0.1	TCP	88	8080 → 51054 [FIN, ACK] Seq=161266484 Ack=2653737805 Win=65536 Len=0 TSval=751734224 TSecr=751734224
12	0.001352755	127.0.0.1	127.0.0.1	TCP	68	51054 → 8080 [ACK] Seq=2653737805 Ack=161266485 Win=65536 Len=0 TSval=751734224 TSecr=751734224

Lines: 1, 2 like V1 as I explained above, and we know the connection was established successfully

We will focus on the rest of the lines starting from 4:

- 1.Once the connection is established, the client will send a PSH-ACK (Push-Acknowledgment) packet (we know that the client did that due to the src port which is: 51054) to the server, which includes the PSH and ACK flags, and the sequence number for the data that is being sent. This packet indicates that the client is ready to transmit the data in the packet, and that the server has acknowledged the receipt of the data.
- 2.The server will then receive the data in the PSH-ACK packet and process it. In this case, the server will print the received data to the console and then send it back to the client in upper case.
- 3.The server will then send a PSH-ACK packet back to the client, which includes the PSH and ACK flags, and the sequence number for the data that is being sent. This packet indicates that the server is ready to transmit the data in the packet, and that the client has acknowledged the receipt of the data.
- 4.The client will then receive the data in the PSH-ACK packet and process it. In this case, the client will print the received data to the console and then close the connection.
- 5.The server will then send a FIN-ACK (Finish-Acknowledgment) packet to the client

Keep in mind that the main difference between V1 and V2 is that server sends [PSH,ACK] twice. The first one contains data with length 5 because this is the size of the buffer (BUFF_SIZE=5). Because this was the first package, we weren't waiting for a previous ACK. The second one contains data with length 15, and that's because the server saved the data in a **buffer** while waiting to first ack and when the ACK was received from the client the server sent all the data in the **buffer**.

V3:

The command to run server.py is `python2 server.py 9090`

The command to run client.py is `python2 client.py 0.0.0.0 9090`

```
import socket,sys

TCP_IP = '0.0.0.0'
TCP_PORT = int(sys.argv[1])
BUFFER_SIZE = 1024

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)

while True:
    conn, addr = s.accept()
    print 'New connection from:', addr
    while True:
        data = conn.recv(BUFFER_SIZE)
        if not data: break
        print "received:", data
        conn.send(data.upper()*1000) |
    conn.close()
```

This code listens for incoming connections on a specified port. It uses the socket module to create a socket and bind it to the specified IP address and port (that is taken as an argument) . It then listens for incoming connections, and when a new connection is made, it prints the address of the client and receives data from the client. It then prints the received data and sends it back to the client in upper case, 1000 times. The server continues to listen for incoming connections and repeats this process forever (outer while never stops) .

```
import socket,sys

TCP_IP = sys.argv[1]
TCP_PORT = int(sys.argv[2])
BUFFER_SIZE = 1024
MESSAGE = "Hello, World!"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send(MESSAGE)
data = s.recv(BUFFER_SIZE)|
print "received data:", data
data = s.recv(BUFFER_SIZE)
print "received data:", data
s.close()
```

This code create a client that connects to a server using the IP address and port number specified as command line arguments. The client then sends a message to the server and receives two responses, which are printed to the console. The client then closes the connection.

Wireshark transportation explanation

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	41578 → 9090 [SYN] Seq=2736042072 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=764147743 TSecr=0 WS=128
2	0.000000782	127.0.0.1	127.0.0.1	TCP	76	9090 → 41578 [SYN, ACK] Seq=48740066 Ack=2736042073 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=764147743 TSecr=764147743 WS=128
3	0.000000818	127.0.0.1	127.0.0.1	TCP	60	41578 → 9090 [ACK] Seq=2736042073 Ack=48740067 Win=65536 Len=0 TSval=764147743 TSecr=764147743
4	0.000012000	127.0.0.1	127.0.0.1	TCP	81	41578 → 9090 [PSH, ACK] Seq=2736042073 Ack=48740067 Win=65536 Len=13 TSval=764147748 TSecr=764147743
5	0.012521202	127.0.0.1	127.0.0.1	TCP	60	9090 → 41578 [ACK] Seq=48740067 Ack=2736042086 Win=65536 Len=0 TSval=764147753 TSecr=764147748
6	0.014011217	127.0.0.1	127.0.0.1	TCP	13068	9090 → 41578 [PSH, ACK] Seq=48740067 Ack=2736042086 Win=65536 Len=13000 TSval=764147757 TSecr=764147748
7	0.014021986	127.0.0.1	127.0.0.1	TCP	60	41578 → 9090 [ACK] Seq=2736042086 Ack=48753867 Win=58496 Len=0 TSval=764147757 TSecr=764147757
8	0.014142958	127.0.0.1	127.0.0.1	TCP	60	41578 → 9090 [RST, ACK] Seq=2736042086 Ack=48753867 Win=65536 Len=0 TSval=764147757 TSecr=764147757

In this capture of wire shark, after running the client and server in V3 we notice a huge difference which is a RST ACK packet the client sends to the server. It is essentially telling the server to reset the connection. This happened because the client is unable to process the large packet (13,000 byte) from the server and needs to terminate the connection. In other words, because the client's application receives 1024 bytes twice, and then closes the connection, some data is left that the client's application hasn't read, and therefore a RST is sent. Other than that it is similar to V1 and V2 in terms of initiating connection (3-way handshake)

V4:

The command to run server.py is `python2 server.py 12345`

The command to run client.py is `python2 client.py 0.0.0.0 12345`

```
import socket,sys

TCP_IP = sys.argv[1]
TCP_PORT = int(sys.argv[2])
BUFFER_SIZE = 1024
MESSAGE = "Hello, World!"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send(MESSAGE*10)
s.send(MESSAGE*10)
s.send(MESSAGE*10)
s.send(MESSAGE*10)
data = s.recv(BUFFER_SIZE)
s.close()

print "received data:", data
```

This is a TCP client that connects to a specified IP address and port. Once connected, the client sends the message "Hello, World!" multiple times (each time he sends is 10 times) to the server and then receives data back from the server. The client then prints the data it receives from the server to the console.

```
import socket,sys,time

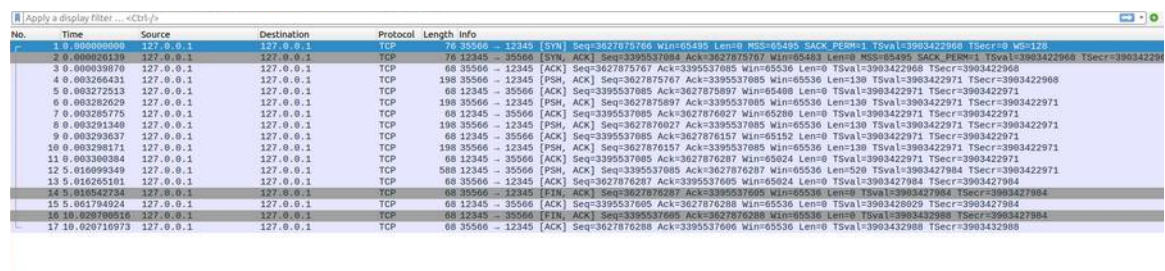
TCP_IP = '0.0.0.0'
TCP_PORT = int(sys.argv[1])
BUFFER_SIZE = 1024

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)

while True:
    conn, addr = s.accept()
    print 'New connection from:', addr
    while True:
        time.sleep(5)
        data = conn.recv(BUFFER_SIZE)
        if not data: break
        print "received:", data
        conn.send(data.upper())
    conn.close()
```

This is a TCP server that listens for incoming connections on a specified port. When a new connection is established, the server receives data from the client in 1024-byte chunks and sends the data back to the client in all upper case. The server also prints the incoming data to the console, and it introduces a delay (for 5 seconds will explain this in depth in the Wireshark capture) between receiving data from the client.

Explanation of the transport layer we see in Wireshark:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	127.0.0.1	127.0.0.1	TCP	76	35566 → 12345 [SYN] Seq=3627875766 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=3903422968 TSecr=0 WS=128
2	0.000000139	127.0.0.1	127.0.0.1	TCP	76	12345 → 35566 [SYN, ACK] Seq=3395537084 Ack=3627875767 Win=65443 Len=0 MSS=65495 SACK_PERM=1 TSval=3903422968 TSecr=3903422968
3	0.000000380	127.0.0.1	127.0.0.1	TCP	68	35566 → 12345 [ACK] Seq=3627875767 Ack=3395537085 Win=65536 Len=0 TSval=3903422968 TSecr=3903422968
4	0.000000643	127.0.0.1	127.0.0.1	TCP	198	35566 → 12345 [PSH, ACK] Seq=3627875767 Ack=3395537085 Win=65536 Len=138 TSval=3903422971 TSecr=3903422968
5	0.000000753	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [ACK] Seq=3395537085 Ack=3627875897 Win=65488 Len=0 TSval=3903422971 TSecr=3903422971
6	0.000000829	127.0.0.1	127.0.0.1	TCP	198	35566 → 12345 [PSH, ACK] Seq=3627875897 Ack=3395537085 Win=65536 Len=138 TSval=3903422971 TSecr=3903422971
7	0.000000975	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [ACK] Seq=3395537085 Ack=3627876027 Win=65536 Len=0 TSval=3903422971 TSecr=3903422971
8	0.000001340	127.0.0.1	127.0.0.1	TCP	198	35566 → 12345 [PSH, ACK] Seq=3627876027 Ack=3395537085 Win=65536 Len=138 TSval=3903422971 TSecr=3903422971
9	0.000001367	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [ACK] Seq=3395537085 Ack=3627876157 Win=65152 Len=0 TSval=3903422971 TSecr=3903422971
10	0.000001671	127.0.0.1	127.0.0.1	TCP	198	35566 → 12345 [PSH, ACK] Seq=3627876157 Ack=3395537085 Win=65536 Len=138 TSval=3903422971 TSecr=3903422971
11	0.000001884	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [ACK] Seq=3395537085 Ack=3627876287 Win=65624 Len=0 TSval=3903422971 TSecr=3903422971
12	5.016099349	127.0.0.1	127.0.0.1	TCP	588	12345 → 35566 [PSH, ACK] Seq=3395537085 Ack=3627876287 Win=65536 Len=528 TSval=3903427984 TSecr=3903422971
13	5.016265101	127.0.0.1	127.0.0.1	TCP	68	35566 → 12345 [ACK] Seq=3627876287 Ack=3395537085 Win=65624 Len=0 TSval=3903427984 TSecr=3903427984
14	5.016545224	127.0.0.1	127.0.0.1	TCP	68	35566 → 12345 [FIN, ACK] Seq=3627876287 Ack=3395537085 Win=65536 Len=0 TSval=3903427984 TSecr=3903427984
15	5.061784924	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [ACK] Seq=3395537605 Ack=3627876288 Win=65536 Len=0 TSval=3903428029 TSecr=3903427984
16	10.628780516	127.0.0.1	127.0.0.1	TCP	68	12345 → 35566 [FIN, ACK] Seq=3395537605 Ack=3627876288 Win=65536 Len=0 TSval=3903432988 TSecr=3903427984
17	10.628716973	127.0.0.1	127.0.0.1	TCP	68	35566 → 12345 [ACK] Seq=3627876288 Ack=3395537606 Win=65536 Len=0 TSval=3903432988 TSecr=3903432988

Here we see the client sent [PSH,ACK] multiple times, a total of 4 messages. Each time its getting ACKed from the server. Meanwhile the server's application is sleeping, so once it wakes up it can read all 520 bytes. Therefore we see that the server sends all the data he received with length 520 (which is the sum lengths that the client sends) after 5 seconds. So, the main difference between what we see here and any other capture before is that in the **Time section** we see delays of 5 seconds. One is between the moment the server ACKed the last PSH,ACK from the client to when the server sent the PSH,ACK to the client. The second delay happens after the server has sent the packet to the client. It goes back to sleep for 5 seconds, meanwhile the client sends FIN to the server. When the server's application wakes up it sees that the clients wants to close the connection, and that the data size in the last message was 0, therefore he exits the loop, sends FIN back to client (which the client ACKs), and closes the connection.

Note: I did not explain the what flags indications here because I explained earlier.

חלק א:

הרצת קטעי הקוד tcp_client.py ו- tcp_server.py על מחשבים שונים, ופירוט על התעבורה

1	0.000000000	10.0.2.15	172.31.19.253	TCP	74	35876 → 12345 [SYN] Seq=3178881805 Win=64240 Len=0 MSS=1460 SACK_PERM=1 Tsval=2472841853 TSecr=0 WS=128
2	0.001553805	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [SYN, ACK] Seq=53568081 Ack=3178881806 Win=65535 Len=0 MSS=1460
3	0.001654976	10.0.2.15	172.31.19.253	TCP	54	35876 → 12345 [ACK] Seq=3178881806 Ack=53568082 Win=64240 Len=0
4	0.002973546	10.0.2.15	172.31.19.253	TCP	59	35876 → 12345 [PSH, ACK] Seq=3178881806 Ack=53568082 Win=64240 Len=5
5	0.003566233	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [ACK] Seq=53568082 Ack=3178881811 Win=65535 Len=0
6	0.003566326	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [PSH, ACK] Seq=53568082 Ack=3178881811 Win=65535 Len=5
7	0.003584731	10.0.2.15	172.31.19.253	TCP	54	35876 → 12345 [ACK] Seq=3178881811 Ack=53568087 Win=64235 Len=0
8	0.003948919	10.0.2.15	172.31.19.253	TCP	63	35876 → 12345 [PSH, ACK] Seq=3178881811 Ack=53568087 Win=64235 Len=9
9	0.004616292	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [ACK] Seq=53568087 Ack=3178881820 Win=65535 Len=0
10	0.005120109	172.31.19.253	10.0.2.15	TCP	63	12345 → 35876 [PSH, ACK] Seq=53568087 Ack=3178881820 Win=65535 Len=9
11	0.005120208	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [FIN, ACK] Seq=53568016 Ack=3178881820 Win=65535 Len=0
12	0.005532090	10.0.2.15	172.31.19.253	TCP	54	35876 → 12345 [FIN, ACK] Seq=3178881820 Ack=53568017 Win=64225 Len=0
13	0.006514404	172.31.19.253	10.0.2.15	TCP	60	12345 → 35876 [ACK] Seq=53568017 Ack=3178881821 Win=65535 Len=0

שליבים 1 - 3 הם שלב "לחיצת הידיים"

שליבים 4 - 11 מתארים את העברת המידע בין השרת ללקוח וכלל בין הלקוח לשרת

שליבים 11 - 13 הם הודעות סיום התקשורת

Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881805, Len: 0
Source Port: 35876
Destination Port: 12345
[Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 3178881805
[Next Sequence Number: 3178881806]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1010 = Header Length: 40 bytes (10)
Flags: 0x002 (SYN)
Window: 64240
[Calculated window size: 64240]
Checksum: 0xcc59 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
TCP Option - Maximum segment size: 1460 bytes
Kind: Maximum Segment Size (2)
Length: 4
MSS Value: 1460

נתבונן על ההודעה הראשונה אשר נשלחת מהלקוח אל השרת.

ניתן לראות כי בתחילת ה-TCP תחת לשונית ה-Flags מופיע דגל ה-SYN.

הלקוח מבקש לעשות סינכרוניזציה (להסתנכרן) עם השרת.

פורט הלקוח הוא: 35876

פורט השרת הוא: 12345.

מבחינת מספרים סידוריים, נתבונן ב- Sequence number וב- Acknowledgment number:

Sequence number: 3178881805

Acknowledgment number: 0

כלומר הלקוח מתחיל את התקשורת ממספר זה (seq) ומאשר שקיבל עד מספר זה (ack).

נראה כיצד מספרים אלו משתנים לאורך התקשורת.

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568001, Ack: 3178881806, Len: 0
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 53568001
  [Next Sequence Number: 53568002]
  Acknowledgment Number: 3178881806
  0110 .... = Header Length: 24 bytes (6)
  Flags: 0x012 (SYN, ACK)
  Window: 65535
  [Calculated window size: 65535]
  Checksum: 0x09d3 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  Options: (4 bytes), Maximum segment size
    TCP Option - Maximum segment size: 1460 bytes
      Kind: Maximum Segment Size (2)
      Length: 4
      MSS Value: 1460

```

כעת השרת מחזיר ללקוח הודעת SYN,ACK. כלומר הוא מאשר את בקשת הסינכרוניזציה (ACK) של השרת ומבקש להסתנכרן עם הלקוח (SYN).
נשים לב שכעת החמט seq של השרת הוא 53568001 (השרת שולח הודעות החלק ממספר זה), בעוד שה-
חמט ack הוא 3178881806 שזה בדיוק 1 מעל החמט seq של הלקוח (השרת קיבל עד מספר זה). כלומר,
השרת שולח חזרה ללקוח 1 נוסף שמסמן את האק לאיבור.

```

Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881806, Ack: 53568002, Len: 0
  Source Port: 35876
  Destination Port: 12345
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 3178881806
  [Next Sequence Number: 3178881806]
  Acknowledgment Number: 53568002
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)

```

בשלב האחרון בחלק שנקרא לו "לחיצת הידיים" הלקוח שולח חזרה אל השרת הודעת ack המאשרת את
הסינכרון שביקש השרת.
ניתן לראות כי החמט seq כעת זהה לחמט ack של השרת: 3178881806. כלומר הלקוח קיבל את האישור
של השרת לבקשת הסינכרון שהוא (הלקוח) שלח אליו (השרת).
מצד שני, החמט ack הוא כעת 53568002, כלומר 1 מעל החמט ack של השרת. משמעות הדבר היא
שהלקוח מאשר את הסינכרון בכך ששולח חזרה לשרת 1 נוסף שמסמן את האק לאיבור.

```
Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881806, Ack: 53568002, Len: 5
  Source Port: 35876
  Destination Port: 12345
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 5]
  Sequence Number: 3178881806
  [Next Sequence Number: 3178881811]
  Acknowledgment Number: 53568002
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x018 (PSH, ACK)
  Window: 64240
  [Calculated window size: 64240]
  [Window size scaling factor: -2 (no window scaling used)]
  Checksum: 0xcc4a [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (5 bytes)
Data (5 bytes)
  Data: 41686d6164
  [Length: 5]
0000  52 54 00 12 35 02 08 00 27 34 88 8a 08 00 45 00  RT...5... '4...E.
0010  00 2d b0 b4 40 00 40 06 bd eb 0a 00 02 0f ac 1f  ---@-@- .....
0020  13 fd 8c 24 30 39 bd 79 e3 0e 03 31 62 02 50 18  ...$09-y ...1b.P-
0030  fa f0 cc 4a 00 00 41 68 6d 61 64                ...J...Ah mad
```

כעת הלקוח שולח אל השרת את השם "Ahmad". ניתן לראות זאת הפירוט הדאטא.
בנוסף, ניתן לראות כי אורך ההודעה הוא 5 (מסומן באדום).
מספרי seq/ack לא השתנו כצפוי.

```
Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568002, Ack: 3178881811, Len: 0
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 53568002
  [Next Sequence Number: 53568002]
  Acknowledgment Number: 3178881811
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
```

כאן ניתן לראות שהשרת מחזיר ללקוח הודעת ack המאשרת את קבלת המידע ("Ahmad") ומספר ack הוא בהתאם גדול ב5 ממספר seq של הלקוח (מספר האק האקדום של השרת): 3178881811. כלומר השרת מאשר שקרא עד מספר זה.

```

Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568002, Ack: 3178881811, Len: 5
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 5]
  Sequence Number: 53568002
  [Next Sequence Number: 53568007]
  Acknowledgment Number: 3178881811
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x018 (PSH, ACK)
  Window: 65535
  [Calculated window size: 65535]
  [Window size scaling factor: -2 (no window scaling used)]
  Checksum: 0x0eb4 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (5 bytes)
Data (5 bytes)
  Data: 41686d6164
  [Length: 5]
0000  08 00 27 34 88 8a 52 54 00 12 35 02 08 00 45 00  ..'4..RT..5...E.
0010  00 2d 10 79 00 00 40 06 9e 27 ac 1f 13 fd 0a 00  --y..@..'.-----
0020  02 0f 30 39 8c 24 03 31 62 02 bd 79 e3 13 50 18  ..09.$..1 b..y..P.
0030  ff ff 0e b4 00 00 41 68 6d 61 64 00              .....Ah mad.

```

כעת השרת שולח חזרה ללקוח את המידע שהוא קיבל. ניתן לראות זאת בפירוט הדאטא.
בשלב זה מספרי הseq/ack נשארים ללא שינוי כצפוי.

```

Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881811, Ack: 53568007, Len: 0
  Source Port: 35876
  Destination Port: 12345
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 3178881811
  [Next Sequence Number: 3178881811]
  Acknowledgment Number: 53568007
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)

```

הלקוח מקבל את המידע מהשרת ומחזיר ack. ניתן לראות שמספר הack גדל ב5 וכעת הוא 53568007.

```

Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881811, Ack: 53568007, Len: 9
  Source Port: 35876
  Destination Port: 12345
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 9]
  Sequence Number: 3178881811
  [Next Sequence Number: 3178881820]
  Acknowledgment Number: 53568007
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x018 (PSH, ACK)
  Window: 64235
  [Calculated window size: 64235]
  [Window size scaling factor: -2 (no window scaling used)]
  Checksum: 0xcc4e [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
  [SEQ/ACK analysis]
  TCP payload (9 bytes)
Data (9 bytes)
  Data: 323131343337323233
  [Length: 9]
0000  52 54 00 12 35 02 08 00 27 34 88 8a 08 00 45 00  RT..5... '4....E.
0010  00 31 b0 b6 40 00 40 06 bd e5 0a 00 02 0f ac 1f  -1..@.@. ....
0020  13 fd 8c 24 30 39 bd 79 e3 13 03 31 62 07 50 18  ...$09.y...1b.P.
0030  fa eb cc 4e 00 00 32 31 31 34 33 37 32 32 33    ...N..21 1437223

```


הלקוח שולח אל השרת מספר ת.ז. אורך ההודעה הוא 9 (מסומן באדום). מספרי הseq/ack נותרים ללא שינוי.

```
Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568007, Ack: 3178881820, Len: 0
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 53568007
  [Next Sequence Number: 53568007]
  Acknowledgment Number: 3178881820
  0101 .... = Header Length: 20 bytes (5)
  + Flags: 0x010 (ACK)
```

השרת מאשר את קבלת המידע (מספר ת.ז.) מהלקוח. מספר הack גדל ב9 וכעת הוא 3178881820 בעוד שמספר הseq התעדכן ל- 53568007.

```
Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568007, Ack: 3178881820, Len: 9
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 9]
  Sequence Number: 53568007
  [Next Sequence Number: 53568016]
  Acknowledgment Number: 3178881820
  0101 .... = Header Length: 20 bytes (5)
  + Flags: 0x018 (PSH, ACK)
  Window: 65535
  [Calculated window size: 65535]
  [Window size scaling factor: -2 (no window scaling used)]
  Checksum: 0x259d [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  + [Timestamps]
  + [SEQ/ACK analysis]
  TCP payload (9 bytes)
Data (9 bytes)
  Data: 323131343337323233
0000  08 00 27 34 88 8a 52 54 00 12 35 02 08 00 45 00  ..'4..RT..5...E.
0010  00 31 10 7b 00 00 40 06 9e 21 ac 1f 13 fd 0a 00  .1{...@...!.....
0020  02 0f 30 39 8c 24 03 31 62 07 bd 79 e3 1c 50 18  ..09.$..1b..y..P.
0030  ff ff 25 9d 00 00 32 31 31 34 33 37 32 32 33  ..%...21 1437223
```

השרת מחזיר ללקוח הודעה עם מספר ת.ז. מספרי הseq/ack ללא שינוי.

```
Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568016, Ack: 3178881820, Len: 0
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 53568016
  [Next Sequence Number: 53568017]
  Acknowledgment Number: 3178881820
  0101 .... = Header Length: 20 bytes (5)
  + Flags: 0x011 (FIN, ACK)
  Window: 65535
  [Calculated window size: 65535]
  [Window size scaling factor: -2 (no window scaling used)]
  Checksum: 0x2173 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  + [Timestamps]
```

הגענו לשלב סיום התקשורת. השרת שולח ללקוח הודעת FIN,ACK. כלומר, השרת מודיע שהוא מעוניין לסיים את התקשורת עם הלקוח. מספר הseq גדל ב9 שכן השרת שלח את הת.ז. בשלב קודם לכן. (כעת עומד על 53568016).

מספר האck נותר ללא שינוי שכן הלקוח לא שלח עוד הודעות אל השרת.

```
Transmission Control Protocol, Src Port: 35876, Dst Port: 12345, Seq: 3178881820, Ack: 53568017, Len: 0
  Source Port: 35876
  Destination Port: 12345
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 3178881820
  [Next Sequence Number: 3178881821]
  Acknowledgment Number: 53568017
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x011 (FIN, ACK)
```

הלקוח שולח לשרת הודעת FIN,ACK למטרת סיום התקשורת ביניהם. בנוסף, ניתן לראות כי מספר האck של הלקוח גדל ב-10 (!) וכעת הוא 53568017. זה מכיוון שהלקוח מאשר גם את קבלת מספר ת.ז בגודל 9 וגם מוסיף 1 נוסף שמסמן את אישור הFIN.

```
Transmission Control Protocol, Src Port: 12345, Dst Port: 35876, Seq: 53568017, Ack: 3178881821, Len: 0
  Source Port: 12345
  Destination Port: 35876
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 53568017
  [Next Sequence Number: 53568017]
  Acknowledgment Number: 3178881821
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
```

לבסוף, השרת מאשר את בקשת סיום התקשורת מהלקוח עם ACK ועם מספר ack גדול ב-1: 3178881821. מספר הseq הוא 53568017 כצפוי.

חלק ב':

סה"כ היו 10 חיבורים. החיבורים היו מול מספרי הפורט הבאים: 34646, 34976, 43892, 59288, 34692, 34688, 34686, 34680, 34678, 34662.

התעבורה מופיעה בקובץ PartB_withRef_Partial.pcapng

נתבון בחיבור הראשון (59288): (שורות 1 - 24)

1.0.00000000	127.0.0.1	127.0.0.1	TCP	76.59288 - 12345 [SYN] Seq=0 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
2.0.00000000	127.0.0.1	127.0.0.1	TCP	76.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
3.0.00000000	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
4.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
5.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
6.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
7.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
8.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
9.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
10.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
11.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
12.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
13.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
14.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
15.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
16.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
17.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
18.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
19.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
20.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
21.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
22.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
23.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
24.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303

לחיצת הידיים

בקשת הלקוח

תהליך שליחת המידע

מהשרת אל הלקוח

תשובה מהשרת

סיום התקשורת

הייתה בקשה אחת מהלקוח לקבל את התמונה 2.jpg בנתיב a/b/.

כשהוזן הקישור בדפדפן נשלחה לשרת הודעת HTTP עם תוכן הבקשה (שורה 4) (כמובן לאחר "לחיצת הידיים" בין הלקוח לשרת), והשרת, לאחר ששלח את המידע שביקש הלקוח, החזיר הודעת HTTP עם התוכן (שורה 20).

במקרה זה הדפדפן והשרת שולחים אחד לשני ומסיימים את התקשורת ביניהם.

1.0.00000000	127.0.0.1	127.0.0.1	TCP	76.59288 - 12345 [SYN] Seq=0 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
2.0.00000000	127.0.0.1	127.0.0.1	TCP	76.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
3.0.00000000	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
4.0.003354441	127.0.0.1	127.0.0.1	HTTP	504 GET /a/b/2.jpg HTTP/1.1
5.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
6.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
7.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
8.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
9.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
10.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
11.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
12.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
13.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
14.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
15.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
16.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
17.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
18.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
19.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
20.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
21.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
22.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
23.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303
24.0.003354441	127.0.0.1	127.0.0.1	TCP	68.59288 - 12345 [ACK] Seq=68 Ack=1 Win=0 Len=0 MSS=65495 SACK_PERM=1 TSval=17597303 TSecr=17597303

Transmission Control Protocol, Src Port: 59288, Dst Port: 12345, Seq: 1, Ack: 1, Len: 446	
GET /a/b/2.jpg HTTP/1.1	
Host: localhost:12345	
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0	
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	
Accept-Language: en-US,en;q=0.5	
Accept-Encoding: gzip, deflate, br	
Connection: keep-alive	
Upgrade-Insecure-Requests: 1	
Sec-Fetch-Dest: document	
Sec-Fetch-Mode: navigate	
Sec-Fetch-Site: none	
Sec-Fetch-User: ?1	
Full request URI: http://localhost:12345/a/b/2.jpg	
[HTTP request 1/1]	
[Response in frame: 20]	

בקשת הלקוח

connection: keep-alive

בקשה לשמור את החיבור חי

19	0.003657040	127.0.0.1	127.0.0.1	TCP	68 59288 → 12345 [ACK] Seq=447 Ack=196676 Win=589440 Len=0 TSva
20	0.003663029	127.0.0.1	127.0.0.1	HTTP	24218 HTTP/1.1 200 OK (JPEG image)
21	0.003668275	127.0.0.1	127.0.0.1	TCP	68 59288 → 12345 [ACK] Seq=447 Ack=220818 Win=720384 Len=0 TSva
22	1.004586913	127.0.0.1	127.0.0.1	TCP	68 12345 → 59288 [FIN, ACK] Seq=220818 Ack=447 Win=65536 Len=8
23	1.004615964	127.0.0.1	127.0.0.1	TCP	68 59288 → 12345 [FIN, ACK] Seq=447 Ack=220819 Win=720384 Len=8
24	1.004620090	127.0.0.1	127.0.0.1	TCP	68 12345 → 59288 [ACK] Seq=220819 Ack=448 Win=65536 Len=0 TSva
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1					
Transmission Control Protocol, Src Port: 12345, Dst Port: 59288, Seq: 196676, Ack: 447, Len: 24142					
[8 Reassembled TCP Segments (229817 bytes): #6(67), #8(32768), #10(32768), #12(32768), #14(32768), #16(32768), #18(32768), #20(24142)]					
Hypertext Transfer Protocol					
HTTP/1.1 200 OK\r\n					
Connection: keep-alive\r\n					
Content-Length: 220750\r\n					
\r\n					
[HTTP response 1/1]					
[Time since request: 0.000315595 seconds]					
[Request in frame: 4]					
[Request URI: http://localhost:12345/a/b/2.jpg]					
File Data: 220750 bytes					
JPEG File Interchange Format					
00000040	0a 0d 0a ff 08 ff e0 00	19 4a 46 49 46 00 01 01	JFIF ..	
00000050	00 00 01 00 01 00 00 ff	e1 00 2a 45 78 69 66 00	"Exif"	
00000060	00 49 49 2a 00 08 00 00	00 01 00 31 01 02 00 07	1	
00000070	00 00 00 1a 00 00 00 00	00 00 00 47 0f 0f 07 0c	Googl	
00000080	05 00 00 ff 0b 00 04 00	03 02 02 0a 0a 0a 0a 0a	e	
00000090	0a 0a 0a 0a 0a 0a 0a 0a	0a 0a 0a 0a 0a 0a 0a 0a		
000000a0	0a 0a 0a 08 08 08 0a 0a	0a 0a 08 08 08 08 08 0a		
000000b0	08 08 08 08 08 0a 08 08	08 08 0a 0a 0a 08 08 00		
000000c0	0d 0a 08 0d 08 08 0a 08	01 03 04 04 06 05 06 0a		
000000d0	06 06 0a 0d 0d 08 0d 0d	0d 0d 0d 0d 0d 0d 0d 00		
000000e0	0d 0d 0d 08 0d 08 08 08	08 08 08 08 08 08 08 08		
000000f0	08 08 08 08 08 08 08 08	08 08 08 08 08 08 08 08		
00000100	08 08 08 08 08 08 08 08	08 ff c0 00 11 08 06 ce		
00000110	05 1a 03 01 22 00 02 11	01 03 11 01 ff c4 00 10		
00000120	09 00 02 02 03 01 01 01	00 00 00 00 00 00 00 00		
00000130	09 02 03 01 04 00 05 06	07 08 09 ff c4 00 4c 10	L	
00000140	09 01 02 04 03 05 07 04	01 03 02 04 04 04 02 00		
00000150	01 00 02 03 11 21 f0 04	31 41 05 12 51 61 71 00	1A -Qeq-	
00000160	01 91 a1 b1 c1 d1 07 13	e1 f1 22 14 32 42 23 52	"-ZB9	
00000170	08 15 33 02 16 43 72 82	17 24 53 92 b2 a2 d2 63	3b-Cr-SS-C	
00000180	73 83 93 c2 e2 09 34 d3	25 ff c4 09 1b 01 00 83	4- %	

המשך החיבור וגודל הקובץ
השרת מבקש מהלקוח
לשמור על החיבור חי

תשובה מהשרת

תוכן הקובץ

חיבור שני: (43892) (שורות 25 - 36)
 בחיבור זה נשלחה הבקשה / (index.html). ניתן לראות את תוכן הקובץ שהשרת מחזיר ללקוח. השרת מבקש לשמור את החיבור חי. תוכן ההודעה הוא מכיל את המחזורת "hello".

פורט לקוח: 43892

בקשת הלקוח

תשובת השרת:
 Connection: keep-alive
 length: 168

תוכן הקובץ "hello"

```

25 5.979822000 127.0.0.1 127.0.0.1 TCP 43892 → 12345 [RST] Seq=0 Win=0 Len=0 MSS=65536 SACK_PERM=1 TSval=17683279 TSecr=0 WS=128
26 5.979840269 127.0.0.1 127.0.0.1 TCP 12345 → 43892 [RST] Seq=0 Ack=0 Win=0 Len=0 MSS=65536 SACK_PERM=1 TSval=17683279 TSecr=0 WS=128
27 5.979860659 127.0.0.1 127.0.0.1 TCP 68.43892 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=17683279 TSecr=17683279
28 5.983679430 127.0.0.1 127.0.0.1 HTTP 68 → 12345 GET / HTTP/1.1
29 5.983680253 127.0.0.1 127.0.0.1 TCP 68 → 12345 [ACK] Seq=1 Ack=438 Win=65536 Len=0 TSval=17683283 TSecr=17683283
30 5.98368243 127.0.0.1 127.0.0.1 TCP 132 → 43892 [PSH, ACK] Seq=1 Ack=1 Win=0 Len=0 TSval=17683283 TSecr=17683283
31 5.983672287 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=17683283 TSecr=17683283
32 5.983644524 127.0.0.1 127.0.0.1 HTTP 68 → 12345 HTTP/1.1 200 OK
33 5.983646343 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=233 Win=65488 Len=0 TSval=17683284 TSecr=17683284
34 5.985188758 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=438 Win=65536 Len=0 TSval=17684285 TSecr=17684285
35 5.985272497 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=234 Win=65536 Len=0 TSval=17684285 TSecr=17684285
36 5.985263648 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=439 Win=65536 Len=0 TSval=17684285 TSecr=17684285
37 26.892583053 127.0.0.1 127.0.0.1 TCP 76 → 43892 [RST] Seq=0 Win=0 Len=0 MSS=65536 SACK_PERM=1 TSval=17624192 TSecr=0 WS=128
38 26.892687686 127.0.0.1 127.0.0.1 TCP 76 → 43892 [RST] Seq=0 Win=0 Len=0 MSS=65536 SACK_PERM=1 TSval=17624192 TSecr=0 WS=128
39 26.892636062 127.0.0.1 127.0.0.1 TCP 68 → 43892 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=17624192 TSecr=17624192

2 [2 Reassembled TCP Segments (232 bytes): 838(44), 432(188)]
Hypertext Transfer Protocol
  HTTP/1.1 200 OK/r/n
    Connection: keep-alive/r/n
    Content-Length: 168/r/n
    /r/n
    [HTTP response 1/1]
    [Time since request: 0.000265094 seconds]
    [Request in frame: 28]
    [Request URI: http://localhost:12345/]
    File Data: 168 bytes

0000 48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f 40 0a HTTP/1.1 200 OK
0010 0a 43 0f 0e 0e 05 03 74 00 0f 0e 3a 29 0b 05 05 -Connect ion: kee
0020 78 2d 01 0c 09 76 05 0d 0a 43 0f 0e 74 05 0a 74 p-alive-Conten
0030 2d 4c 05 0e 07 7d 00 3a 20 31 20 30 0d 0a 0f 0a -Length: 168
0040 1c 00 74 00 01 3c 00 03 00 00 05 05 01 0a 30 00 <html><body web
0050 3e 00 00 3c 74 00 7d 0c 0c 05 3a 04 79 28 77 03 02 <title>emmy web
0060 10 70 01 07 05 3c 27 74 09 74 0c 05 3e 0d 0a 09 <page>/? title=
0070 09 3c 00 09 0e 0b 20 72 05 0c 3d 22 09 03 0f 0a <link r el="icon
0080 02 10 7d 79 19 02 3d 22 09 04 01 07 05 2f 79 2e <type" /> <img src
0090 09 03 0f 0e 22 20 09 72 09 06 3d 22 0a 01 79 09 icon" sr c="/fo
00a0 03 0f 0e 2e 09 03 0f 22 3e 0d 0a 00 3c 2f 0a 00 <an icon" /></ne
00b0 01 04 30 0d 0a 00 3c 02 0f 04 79 3e 0d 0a 09 09 <do
00c0 1c 02 3a 3c 75 3a 0b 05 0c 0c 0f 3c 2f 75 3a 3c <me
00d0 0f 02 3a 0d 0a 00 3c 2f 02 0f 04 79 3e 0d 0a 3c <bo
00e0 0f 02 74 0d 0c 3e 0d 0a
    
```


חיבור שלישי (34976): (שורות 37 - 48)
 בחיבור זה נשלחה הבקשה /result.html. ניתן לראות את פרטי החיבור שהשרת מחזיר ללקוח לאחר יצירת החיבור.

The image shows a Wireshark packet capture of a TCP connection. The first three packets (37-39) show the SYN, SYN-ACK, and ACK exchange. Packet 40 is an HTTP GET request for /result.html. A yellow box highlights this packet, and an arrow points to its details pane. The details pane shows the HTTP request structure, including the method (GET), URI (/result.html), and version (1.1). Below the packet list, the packet bytes are displayed in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
37	26.892563853	127.0.0.1	127.0.0.1	TCP	76	34976 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1
38	26.892667698	127.0.0.1	127.0.0.1	TCP	76	12345 → 34976 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0
39	26.892630982	127.0.0.1	127.0.0.1	TCP	68	34976 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0
40	26.896312985	127.0.0.1	127.0.0.1	HTTP	516	GET /result.html HTTP/1.1
41	26.896319759	127.0.0.1	127.0.0.1	TCP	68	12345 → 34976 [ACK] Seq=1 Ack=449 Win=65536 Len=0 TSval=17624192
42	26.896473968	127.0.0.1	127.0.0.1	TCP	68	34976 → 12345 [ACK] Seq=449 Ack=66 Win=65536 Len=0 TSval=17624192
43	26.896479806	127.0.0.1	127.0.0.1	TCP	68	34976 → 12345 [ACK] Seq=449 Ack=66 Win=65536 Len=0 TSval=17624192
44	26.896546887	127.0.0.1	127.0.0.1	HTTP	199	HTTP/1.1 200 OK
45	26.896549274	127.0.0.1	127.0.0.1	TCP	68	34976 → 12345 [ACK] Seq=449 Ack=187 Win=65536 Len=0 TSval=17624192
46	27.897717446	127.0.0.1	127.0.0.1	TCP	68	12345 → 34976 [FIN, ACK] Seq=187 Ack=449 Win=65536 Len=0 TSval=17624192
47	27.897912709	127.0.0.1	127.0.0.1	TCP	68	34976 → 12345 [FIN, ACK] Seq=449 Ack=188 Win=65536 Len=0 TSval=17624192
48	27.897952396	127.0.0.1	127.0.0.1	TCP	68	12345 → 34976 [ACK] Seq=188 Ack=450 Win=65536 Len=0 TSval=17624192
49	47.043914010	127.0.0.1	127.0.0.1	TCP	76	34646 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1
50	47.043938713	127.0.0.1	127.0.0.1	TCP	76	12345 → 34646 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0
51	47.043962228	127.0.0.1	127.0.0.1	TCP	68	34646 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=176443444
52	48.045369425	127.0.0.1	127.0.0.1	TCP	68	12345 → 34646 [FIN, ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=176443444
53	48.045565753	127.0.0.1	127.0.0.1	TCP	68	34646 → 12345 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=176443444
54	48.045587281	127.0.0.1	127.0.0.1	TCP	68	12345 → 34646 [ACK] Seq=2 Ack=2 Win=65536 Len=0 TSval=176443444

Frame 42: 132 bytes on wire (1056 bits), 132 bytes captured (1056 bits) on interface eth0, id 0
 Linux cooked capture v1
 Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
 Transmission Control Protocol, Src Port: 12345, Dst Port: 34976, Seq: 1, Ack: 449, Len: 64

000 08 00 03 84 08 06 00 00 00 00 00 01 5e 08 80
 010 45 00 00 74 f2 ff 40 00 40 06 49 82 7f 00 00 81 E-t-@-@-I-
 020 7f 00 00 81 38 39 88 a0 d8 79 89 c9 45 b4 7b 78 ---09--y-E-(x
 030 88 18 02 80 fe 68 00 00 01 81 88 0a 01 0c ec 84 ---h-
 040 01 0c ec 84 48 54 54 50 2f 31 2e 31 28 32 30 30 ---HTTP/1.1 200
 050 28 4f 4b 8d 8a 43 6f 6e 6e 65 63 74 69 6f 6e 3a OK-Connection:
 060 28 6b 65 65 78 2d 61 6c 69 76 65 6d 8a 43 6f 6e keep-alive-Conn
 070 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 28 31 32 32 tent-Length: 122
 080 0d 8a 8d 8a

חיבור רביעי (34646): (שורות 49 - 54)
 בfirefox כל הקלדה בשורת URL יוצרת חיבור עם השרת. כאן באה לידי ביטוי ההקלדה של הבקשה הבאה.
 (הערה: חיבור זה איזו מופיע בקובץ הקצב, בשל מגבלות מקום החלטנו לא לשמור את התעבורה)

49	47.043914010	127.0.0.1	127.0.0.1	TCP	76	34646 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495
50	47.043938713	127.0.0.1	127.0.0.1	TCP	76	12345 → 34646 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0
51	47.043962228	127.0.0.1	127.0.0.1	TCP	68	34646 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=176443444
52	48.045369425	127.0.0.1	127.0.0.1	TCP	68	12345 → 34646 [FIN, ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=176443444
53	48.045565753	127.0.0.1	127.0.0.1	TCP	68	34646 → 12345 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0 TSval=176443444
54	48.045587281	127.0.0.1	127.0.0.1	TCP	68	12345 → 34646 [ACK] Seq=2 Ack=2 Win=65536 Len=0 TSval=176443444

חיבור 5 - 10 (34662, 34678, 34680, 34686, 34688, 34692): משורה 55 עד הסוף. עבור הבקשה a/b/ref.html

הערה: חלק מהחבילות (אלו שלא מופיעות בצילומי המסך) הוסרו מקובץ הקפס המקורי בשל מגבלות הגודל של ההגשה במודל. על כן, חלק ממספרי השורות לא יהיו תואמות. עם זאת, כל התוכן המוצג כאן מופיע גם בקובץ הקפס.

הפעם נשלחת יותר מבקשה אחת על אותו חיבור, למשל אחרי ש - a/b/ref.html נשלחת, גם a/1.jpg

54	48.045587281	127.0.0.1	127.0.0.1	TCP	68	12345	-	34646	[ACK]	Seq=2	Ack=2	Win=65536	Len=0	TSval=
55	48.661175543	127.0.0.1	127.0.0.1	TCP	76	34662	-	12345	[SYN]	Seq=0	Win=65495	Len=0	MSS=65495	
56	48.661199181	127.0.0.1	127.0.0.1	TCP	76	12345	-	34662	[SYN, ACK]	Seq=0	Ack=1	Win=65483	Len=0	
57	48.661222481	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=1	Ack=1	Win=65536	Len=0	TSval=
58	48.758758922	127.0.0.1	127.0.0.1	HTTP	517	GET	/a/b/ref.html	HTTP/1.1						
59	48.758770233	127.0.0.1	127.0.0.1	TCP	68	12345	-	34662	[ACK]	Seq=1	Ack=450	Win=65152	Len=0	TSval=
60	48.758872949	127.0.0.1	127.0.0.1	TCP	132	12345	-	34662	[PSH, ACK]	Seq=1	Ack=450	Win=65536	Len=0	
61	48.758876732	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=450	Ack=68	Win=65536	Len=0	TSval=
62	48.758948182	127.0.0.1	127.0.0.1	HTTP	671	HTTP/1.1	200 OK							
63	48.758941986	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=450	Ack=68	Win=65536	Len=0	TSval=
64	48.786419048	127.0.0.1	127.0.0.1	HTTP	451	GET	/a/1.jpg	HTTP/1.1						
65	48.786522314	127.0.0.1	127.0.0.1	TCP	135	12345	-	34662	[PSH, ACK]	Seq=668	Ack=833	Win=65536	Len=0	
66	48.786708725	127.0.0.1	127.0.0.1	TCP	76	34678	-	12345	[SYN]	Seq=0	Win=65495	Len=0	MSS=65495	
67	48.786714068	127.0.0.1	127.0.0.1	TCP	76	12345	-	34678	[SYN, ACK]	Seq=0	Ack=1	Win=65483	Len=0	
68	48.786719620	127.0.0.1	127.0.0.1	TCP	68	34678	-	12345	[ACK]	Seq=1	Ack=1	Win=65536	Len=0	TSval=
69	48.786757822	127.0.0.1	127.0.0.1	HTTP	453	GET	/a/b/1.jpg	HTTP/1.1						
70	48.786759816	127.0.0.1	127.0.0.1	TCP	68	12345	-	34678	[ACK]	Seq=1	Ack=386	Win=65152	Len=0	TSval=
71	48.786809062	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[ACK]	Seq=735	Ack=833	Win=65536	Len=0	TSval=
72	48.786839579	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=33503	Win=65536	Len=0	
73	48.786869448	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=33503	Ack=833	Win=65536	Len=0	
74	48.786874649	127.0.0.1	127.0.0.1	TCP	32836	[TCP window full]	12345	-	34662	[ACK]	Seq=66271	Ack=833	Win=65536	
75	48.786881127	127.0.0.1	127.0.0.1	TCP	76	34680	-	12345	[SYN]	Seq=0	Win=65495	Len=0	MSS=65495	
76	48.786883768	127.0.0.1	127.0.0.1	TCP	76	12345	-	34680	[SYN, ACK]	Seq=0	Ack=1	Win=65483	Len=0	
77	48.786886733	127.0.0.1	127.0.0.1	TCP	68	34680	-	12345	[ACK]	Seq=1	Ack=1	Win=65536	Len=0	TSval=
78	48.786897588	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=99039	Win=65536	Len=0	
79	48.786902788	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=99039	Ack=833	Win=65536	Len=0	
80	48.786907623	127.0.0.1	127.0.0.1	TCP	32836	[TCP window full]	12345	-	34662	[ACK]	Seq=131807	Ack=833	Win=65536	
81	48.786915156	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=164575	Win=32752	Len=0	
82	48.786919112	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=164575	Ack=833	Win=65536	Len=0	

Frame 58: 517 bytes on wire (4136 bits), 517 bytes captured (4136 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 34662, Dst Port: 12345, Seq: 1, Ack: 1, Len: 449
Hypertext Transfer Protocol
GET /a/b/ref.html HTTP/1.1

פורט לקוח: 34662

64	48.786419048	127.0.0.1	127.0.0.1	HTTP	451	GET	/a/1.jpg	HTTP/1.1					
65	48.786522314	127.0.0.1	127.0.0.1	TCP	135	12345	-	34662	[PSH, ACK]	Seq=668	Ack=833	Win=65536	Len=0
66	48.786708725	127.0.0.1	127.0.0.1	TCP	76	34678	-	12345	[SYN]	Seq=0	Win=65495	Len=0	MSS=65495
67	48.786714068	127.0.0.1	127.0.0.1	TCP	76	12345	-	34678	[SYN, ACK]	Seq=0	Ack=1	Win=65483	Len=0
68	48.786719620	127.0.0.1	127.0.0.1	TCP	68	34678	-	12345	[ACK]	Seq=1	Ack=1	Win=65536	Len=0
69	48.786757822	127.0.0.1	127.0.0.1	HTTP	453	GET	/a/b/1.jpg	HTTP/1.1					
70	48.786759816	127.0.0.1	127.0.0.1	TCP	68	12345	-	34678	[ACK]	Seq=1	Ack=386	Win=65152	Len=0
71	48.786809062	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[ACK]	Seq=735	Ack=833	Win=65536	Len=0
72	48.786839579	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=33503	Win=65536	Len=0
73	48.786869448	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=33503	Ack=833	Win=65536	Len=0
74	48.786874649	127.0.0.1	127.0.0.1	TCP	32836	[TCP window full]	12345	-	34662	[ACK]	Seq=66271	Ack=833	Win=65536
75	48.786881127	127.0.0.1	127.0.0.1	TCP	76	34680	-	12345	[SYN]	Seq=0	Win=65495	Len=0	MSS=65495
76	48.786883768	127.0.0.1	127.0.0.1	TCP	76	12345	-	34680	[SYN, ACK]	Seq=0	Ack=1	Win=65483	Len=0
77	48.786886733	127.0.0.1	127.0.0.1	TCP	68	34680	-	12345	[ACK]	Seq=1	Ack=1	Win=65536	Len=0
78	48.786897588	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=99039	Win=65536	Len=0
79	48.786902788	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=99039	Ack=833	Win=65536	Len=0
80	48.786907623	127.0.0.1	127.0.0.1	TCP	32836	[TCP window full]	12345	-	34662	[ACK]	Seq=131807	Ack=833	Win=65536
81	48.786915156	127.0.0.1	127.0.0.1	TCP	68	34662	-	12345	[ACK]	Seq=833	Ack=164575	Win=32752	Len=0
82	48.786919112	127.0.0.1	127.0.0.1	TCP	32836	12345	-	34662	[PSH, ACK]	Seq=164575	Ack=833	Win=65536	Len=0

Frame 64: 451 bytes on wire (3608 bits), 451 bytes captured (3608 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 34662, Dst Port: 12345, Seq: 450, Ack: 668, Len: 383
Hypertext Transfer Protocol
GET /a/1.jpg HTTP/1.1

בקשה נוספת על אותו חיבור

95	48.787026669	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=833 Ack=525023 Win=842624 Len=0 TSval
96	48.787029461	127.0.0.1	127.0.0.1	HTTP	7783 HTTP/1.1 200 OK (JPEG JFIF image)
97	48.787278020	127.0.0.1	127.0.0.1	HTTP	451 GET /a/2.jpg HTTP/1.1
98	48.787280038	127.0.0.1	127.0.0.1	TCP	68 12345 → 34680 [ACK] Seq=1 Ack=384 Win=65152 Len=0 TSval=1764
99	48.787318051	127.0.0.1	127.0.0.1	HTTP	451 GET /a/3.jpg HTTP/1.1
100	48.787341153	127.0.0.1	127.0.0.1	TCP	76 34686 → 12345 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PER
101	48.787366731	127.0.0.1	127.0.0.1	TCP	76 12345 → 34686 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=654
102	48.787369539	127.0.0.1	127.0.0.1	TCP	68 34686 → 12345 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=176460
103	48.787377532	127.0.0.1	127.0.0.1	TCP	135 12345 → 34662 [PSH, ACK] Seq=532658 Ack=1216 Win=65536 Len=0
104	48.787433205	127.0.0.1	127.0.0.1	HTTP	453 GET /a/b/2.jpg HTTP/1.1
105	48.787435116	127.0.0.1	127.0.0.1	TCP	68 12345 → 34686 [ACK] Seq=1 Ack=386 Win=65152 Len=0 TSval=1764
106	48.787525337	127.0.0.1	127.0.0.1	TCP	551 12345 → 34662 [ACK] Seq=532725 Ack=1216 Win=65536 Len=65483
107	48.787532142	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=1216 Ack=598208 Win=1068032 Len=0 TS
Transmission Control Protocol, Src Port: 34662, Dst Port: 12345, Seq: 833, Ack: 532658, Len: 383					
Hypertext Transfer Protocol					
GET /a/3.jpg HTTP/1.1\r\n					
Host: localhost:12345\r\n					
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0\r\n					
Accept: image/avif,image/webp,*/*\r\n					
Accept-Language: en-US,en;q=0.5\r\n					

בקשה שנייה על אותו חיבור

גדל בכל פעם seq

גם הבקשות עבור a/4.jpg , a/5.jpg , a/b/5.jpg , a/6.jpg , a/b/6.jpg עוברות על גבי אותו החיבור הנ"ל.

לעומת זאת הבקשה עבור a/b/1.jpg/ שנשלחת מפורט אחר, כלומר, חיבור חדש.

68	48.786719620	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=1 Ack=1 Wi
69	48.786757822	127.0.0.1	127.0.0.1	HTTP	453 GET /a/b/1.jpg HTTP/1.1
70	48.786759816	127.0.0.1	127.0.0.1	TCP	68 12345 → 34678 [ACK] Seq=1 Ack=386
71	48.786809062	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [ACK] Seq=735 Ack=83
72	48.786839579	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=833 Ack=33
73	48.786869448	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [PSH, ACK] Seq=33503
74	48.786874649	127.0.0.1	127.0.0.1	TCP	32836 [TCP Window Full] 12345 → 34662 [A
75	48.786881127	127.0.0.1	127.0.0.1	TCP	76 34680 → 12345 [SYN] Seq=0 Win=6549
76	48.786883768	127.0.0.1	127.0.0.1	TCP	76 12345 → 34680 [SYN, ACK] Seq=0 Ack
77	48.786886733	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=1 Ack=1 Wi
78	48.786897588	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=833 Ack=99
79	48.786902788	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [PSH, ACK] Seq=99039
80	48.786907623	127.0.0.1	127.0.0.1	TCP	32836 [TCP Window Full] 12345 → 34662 [A
81	48.786915156	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=833 Ack=16
82	48.786919112	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [PSH, ACK] Seq=16457
83	48.786922291	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [ACK] Seq=197343 Ack
84	48.786925835	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [PSH, ACK] Seq=23011
85	48.786929570	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [ACK] Seq=262879 Ack
86	48.786933914	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34662 [PSH, ACK] Seq=29564
Transmission Control Protocol, Src Port: 34678, Dst Port: 12345, Seq: 1, Ack: 1, Len: 385					
Hypertext Transfer Protocol					
GET /a/b/1.jpg HTTP/1.1\r\n					
Host: localhost:12345\r\n					
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0\r\n					
Accept: image/avif,image/webp,*/*\r\n					
Accept-Language: en-US,en;q=0.5\r\n					

34678

כנל לגבי הבקשה עבור a/2.jpg. כך גם כל הבקשות: a/b/2.jpg, a/b/3.jpg, a/b/4.jpg נשלחות על פורט נפרד עבור כל בקשה.

96	48.787029461	127.0.0.1	127.0.0.1	HTTP	7783 HTTP/1.1 200 OK (JPEG JFIF image)
97	48.787278020	127.0.0.1	127.0.0.1	HTTP	451 GET /a/2.jpg HTTP/1.1
98	48.787280038	127.0.0.1	127.0.0.1	TCP	68 12345 → 34680 [ACK] Seq=1 Ack=384 Win=6515
99	48.787318051	127.0.0.1	127.0.0.1	HTTP	451 GET /a/3.jpg HTTP/1.1
100	48.787341153	127.0.0.1	127.0.0.1	TCP	76 34686 → 12345 [SYN] Seq=0 Win=65495 Len=0
101	48.787366731	127.0.0.1	127.0.0.1	TCP	76 12345 → 34686 [SYN, ACK] Seq=0 Ack=1 Win=6
102	48.787369539	127.0.0.1	127.0.0.1	TCP	68 34686 → 12345 [ACK] Seq=1 Ack=1 Win=65536
103	48.787377532	127.0.0.1	127.0.0.1	TCP	135 12345 → 34662 [PSH, ACK] Seq=532658 Ack=12
104	48.787433205	127.0.0.1	127.0.0.1	HTTP	453 GET /a/b/2.jpg HTTP/1.1
105	48.787435116	127.0.0.1	127.0.0.1	TCP	68 12345 → 34686 [ACK] Seq=1 Ack=386 Win=6515
106	48.787525337	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=532725 Ack=1216 Wi
107	48.787532142	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=1216 Ack=598208 Wi
Transmission Control Protocol, Src Port: 34680, Dst Port: 12345, Seq: 1, Ack: 1, Len: 383					
Hypertext Transfer Protocol					
GET /a/2.jpg HTTP/1.1\r\n					
Host: localhost:12345\r\n					
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0\r\n					
Accept: image/avif,image/webp,*/*\r\n					
Accept-Language: en-US,en;q=0.5\r\n					

34680

166	48.789228342	127.0.0.1	127.0.0.1	TCP	60001 12345 → 34662 [ACK] Seq=2380047 Ack=2750 Win=65536 Len=0 TSval=1
167	48.789228342	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=2750 Ack=2185547 Win=3012224 Len=0 TSval=1
168	48.789232375	127.0.0.1	127.0.0.1	HTTP	8127 HTTP/1.1 200 OK (JPEG JFIF image)
169	48.789250061	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=2750 Ack=2259089 Win=3041824 Len=0 TSval=1
170	48.789342445	127.0.0.1	127.0.0.1	HTTP	453 GET /a/b/6.jpg HTTP/1.1
171	48.789378930	127.0.0.1	127.0.0.1	TCP	135 12345 → 34662 [PSH, ACK] Seq=2259089 Ack=3135 Win=65536 Len=67 TSval=1
172	48.789647641	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2259156 Ack=3135 Win=65536 Len=65483 TSval=1
173	48.789653851	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=3135 Ack=2259156 Win=0 TSval=1
174	48.789660096	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=23246 Ack=3135 Win=65536 Len=65483 TSval=1
175	48.789684632	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2390122 Ack=3135 Win=65536 Len=65483 TSval=1
176	48.789689076	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=3135 Ack=2455605 Win=3112448 Len=0 TSval=1
177	48.789701329	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2455605 Ack=3135 Win=65536 Len=65483 TSval=1
178	48.789717533	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2521808 Ack=3135 Win=65536 Len=65483 TSval=1
179	48.789721852	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=3135 Ack=2586571 Win=3112448 Len=0 TSval=1
180	48.789733957	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2586571 Ack=3135 Win=65536 Len=65483 TSval=1
181	48.789750205	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2652854 Ack=3135 Win=65536 Len=65483 TSval=1
182	48.789754387	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=3135 Ack=2717537 Win=3112448 Len=0 TSval=1
183	48.789767359	127.0.0.1	127.0.0.1	TCP	65551 12345 → 34662 [ACK] Seq=2717537 Ack=3135 Win=65536 Len=65483 TSval=1
184	48.789779599	127.0.0.1	127.0.0.1	HTTP	8127 HTTP/1.1 200 OK (JPEG JFIF image)
185	48.789781091	127.0.0.1	127.0.0.1	TCP	68 34662 → 12345 [ACK] Seq=3135 Ack=2791079 Win=3141120 Len=0 TSval=1
186	48.790045430	127.0.0.1	127.0.0.1	TCP	68 12345 → 34662 [FIN, ACK] Seq=2791079 Ack=3135 Win=65536 Len=0 TSval=1
187	48.790213426	127.0.0.1	127.0.0.1	TCP	68 12345 → 12345 [FIN, ACK] Seq=3135 Ack=2791080 Win=3141120 Len=0 TSval=1
188	48.790242399	127.0.0.1	127.0.0.1	TCP	68 12345 → 34662 [ACK] Seq=2791080 Ack=3136 Win=65536 Len=0 TSval=176
189	48.790395783	127.0.0.1	127.0.0.1	TCP	135 12345 → 34678 [PSH, ACK] Seq=1 Ack=386 Win=65536 Len=67 TSval=1764
190	48.790418713	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=68 Win=65536 Len=8 TSval=17647890
191	48.791032380	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34678 [ACK] Seq=68 Ack=386 Win=65536 Len=32768 TSval=176478
192	48.791061503	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=32836 Win=48512 Len=0 TSval=176470
193	48.791104746	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34678 [ACK] Seq=68 Ack=386 Win=65536 Len=32768 TSval=176470

הבקשה האחרונה

הראשון FIN

כעת לאחר שהלוקו שלח את כל הבקשות מתחילים להיסגר כל החיבורים. כאן ניתן לראות את החיבור הראשון שנסגר, שהוא גם החיבור הראשון שנפתח עבור הבקשה הראשונה a/b/ref.html. כפי שצוין מקודם, הבקשה a/b/6.jpg יושבת על החיבור עם פורט 34662, אותו חיבור ראשוני שעכשיו נסגר.

בהמשך לאחר שנשלח מידע נוסף (מידע בינארי של התמונות שמוצגות) נסגרים חיבורים נוספים.

217	48.791861338	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=398958 Win=982272 Len=0 TSval=17647891 TSecr=17647891
218	48.791869425	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=393284 Win=123344 Len=0 TSval=17647891 TSecr=17647891
219	48.791870233	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=428952 Win=1244288 Len=0 TSval=17647891 TSecr=17647891
220	48.791886649	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=458626 Win=1375232 Len=0 TSval=17647891 TSecr=17647891
221	48.791894821	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=491580 Win=1396176 Len=0 TSval=17647891 TSecr=17647891
222	48.791902921	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=524256 Win=1437120 Len=0 TSval=17647891 TSecr=17647891
223	48.791911586	127.0.0.1	127.0.0.1	HTTP	7763 HTTP/1.1 200 OK (JPEG JFIF image)
224	48.791922187	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=531901 Win=1768064 Len=0 TSval=17647891 TSecr=17647891
225	48.792711582	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [FIN, ACK] Seq=386 Ack=531901 Win=65536 Len=0 TSval=17648092 TSecr=17647891
226	48.792879431	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [FIN, ACK] Seq=386 Ack=531902 Win=1768064 Len=0 TSval=17648092 TSecr=17648092
227	48.792909443	127.0.0.1	127.0.0.1	TCP	68 34678 → 12345 [ACK] Seq=386 Ack=531902 Win=65536 Len=0 TSval=17648092 TSecr=17648092
228	48.793129524	127.0.0.1	127.0.0.1	TCP	135 12345 → 34680 [PSH, ACK] Seq=1 Ack=384 Win=65536 Len=67 TSval=17648093 TSecr=17648087 [TCP segment of a
229	48.793140847	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=384 Win=65536 Len=0 TSval=17648093 TSecr=17648093
230	48.793149952	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34680 [ACK] Seq=68 Ack=384 Win=65536 Len=32768 TSval=17648093 TSecr=17648093 [TCP segment of a r
231	48.793171374	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=32836 Win=484912 Len=0 TSval=17648093 TSecr=17648093
232	48.793230719	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34680 [PSH, ACK] Seq=384 Ack=32836 Win=65536 Len=32768 TSval=17648093 TSecr=17648093 [TCP segment
233	48.793440550	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=65886 Win=480512 Len=0 TSval=17648093 TSecr=17648093 [TCP segment of
234	48.793757751	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34680 [ACK] Seq=65884 Ack=384 Win=65536 Len=32768 TSval=17648093 TSecr=17648093 [TCP segment of
235	48.793770883	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=98372 Win=489512 Len=0 TSval=17648093 TSecr=17648093
236	48.793790557	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34680 [PSH, ACK] Seq=384 Ack=98372 Win=65536 Len=32768 TSval=17648093 TSecr=17648093 [TCP segment
237	48.793835860	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=131140 Win=487648 Len=0 TSval=17648093 TSecr=17648093
238	48.793950883	127.0.0.1	127.0.0.1	TCP	68 [TCP window update] 34680 → 12345 [ACK] Seq=384 Ack=131140 Win=65536 Len=0 TSval=17648094 TSecr=17648093
239	48.793975700	127.0.0.1	127.0.0.1	TCP	32836 12345 → 34680 [ACK] Seq=131140 Ack=384 Win=65536 Len=32768 TSval=17648094 TSecr=17648094 [TCP segment of
240	48.793980537	127.0.0.1	127.0.0.1	TCP	32836 [TCP window full] 12345 → 34680 [ACK] Seq=131140 Ack=384 Win=65536 Len=32768 TSval=17648094 TSecr=17648094
241	48.794010790	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=134400 Win=134400 Len=0 TSval=17648094 TSecr=17648094
242	48.794039411	127.0.0.1	127.0.0.1	HTTP	84219 HTTP/1.1 200 OK (JPEG JFIF image)
243	48.794047100	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=130576 Win=327552 Len=0 TSval=17648094 TSecr=17648094
244	48.794050136	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=384 Ack=228819 Win=458496 Len=0 TSval=17648094 TSecr=17648094
245	48.794110812	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [FIN, ACK] Seq=228819 Ack=384 Win=65536 Len=0 TSval=17648094 TSecr=17648094
246	48.794209584	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [FIN, ACK] Seq=384 Ack=228819 Win=458496 Len=0 TSval=17648094 TSecr=17648094
247	48.794209323	127.0.0.1	127.0.0.1	TCP	68 34680 → 12345 [ACK] Seq=228819 Ack=385 Win=65536 Len=0 TSval=17648094 TSecr=17648094
248	48.794530321	127.0.0.1	127.0.0.1	TCP	135 12345 → 34680 [PSH, ACK] Seq=1 Ack=384 Win=65536 Len=67 TSval=17648094 TSecr=17648087 [TCP segment of a

חיבורים נוספים נסגרים

Frame 175: 85551 bytes on wire (524400 bits), 85551 bytes captured (524400 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 12345, Dst Port: 34662, Seq: 2390122, Ack: 3135, Len: 85483

עד שלבסוף נסגר החיבור האחרון (34692).

308	53.800488189	127.0.0.1	127.0.0.1	TCP	32836	12345 → 34692	[PSH, ACK] Seq=368516 Ack=386 Win=65536 Len=32768 TSval=1765110
309	53.800495199	127.0.0.1	127.0.0.1	TCP	32836	12345 → 34692	[ACK] Seq=393284 Ack=386 Win=65536 Len=32768 TSval=1765110
310	53.800518971	127.0.0.1	127.0.0.1	TCP	32836	12345 → 34692	[PSH, ACK] Seq=426052 Ack=386 Win=65536 Len=32768 TSval=1765110
311	53.800526437	127.0.0.1	127.0.0.1	TCP	32836	12345 → 34692	[ACK] Seq=458828 Ack=386 Win=65536 Len=32768 TSval=1765110
312	53.800604647	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
313	53.800627678	127.0.0.1	127.0.0.1	TCP	32836	12345 → 34692	[ACK] Seq=458828 Ack=386 Win=65536 Len=32768 TSval=1765110
314	53.800651811	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
315	53.800668678	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
316	53.800683234	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
317	53.800684399	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
318	53.800681874	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
319	53.800686596	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=229444 Win=458496 Len=0 TSval=1765110
320	54.801358896	127.0.0.1	127.0.0.1	TCP	68	12345 → 34692	[FIN, ACK] Seq=531991 Ack=386 Win=65536 Len=0 TSval=17652101
321	54.801552313	127.0.0.1	127.0.0.1	TCP	68	34692 → 12345	[ACK] Seq=386 Ack=531992 Win=449928 Len=0 TSval=17652101
322	54.801574873	127.0.0.1	127.0.0.1	TCP	68	12345 → 34692	[ACK] Seq=531992 Ack=387 Win=65536 Len=0 TSval=17652101

התשובה האחרונה של השרת

34692

Frame 67: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
Transmission Control Protocol, Src Port: 12345, Dst Port: 34678, Seq: 0, Ack: 1, Len: 0

0000 00 00 03 04 00 00 00 00 00 00 00 00 ff c3 00 00
0010 45 00 00 3c 00 00 40 00 40 00 3c ba 7f 00 00 01
0020 7f 00 00 01 30 39 87 76 a8 b9 9b 35 14 de 27 24

דוגמא נוספת לקובץ שלא קיים (נמצא בקובץ PartB_close_example.pcapng):

21	1.538270101	127.0.0.1	127.0.0.1	TCP	76	58048 → 12345	[SYN] Seq=0 Win=65495 Len=0 MSS=65536
22	1.538294320	127.0.0.1	127.0.0.1	TCP	76	12345 → 58048	[SYN, ACK] Seq=0 Ack=1 Win=65495 Len=0
23	1.538317973	127.0.0.1	127.0.0.1	TCP	68	58048 → 12345	[ACK] Seq=1 Ack=1 Win=65536 Len=0
24	1.540948359	127.0.0.1	127.0.0.1	HTTP	510	GET /chtml HTTP/1.1	
25	1.540954461	127.0.0.1	127.0.0.1	TCP	68	12345 → 58048	[ACK] Seq=1 Ack=443 Win=65152 Len=0
26	1.541015816	127.0.0.1	127.0.0.1	TCP	113	12345 → 58048	[PSH, ACK] Seq=1 Ack=443 Win=65152 Len=0
27	1.541028803	127.0.0.1	127.0.0.1	TCP	68	58048 → 12345	[ACK] Seq=443 Ack=46 Win=65536 Len=0
28	1.541032744	127.0.0.1	127.0.0.1	HTTP	68	HTTP/1.1 404 Not Found	
29	1.541093065	127.0.0.1	127.0.0.1	TCP	68	58048 → 12345	[FIN, ACK] Seq=443 Ack=47 Win=0 Len=0
30	1.541096355	127.0.0.1	127.0.0.1	TCP	68	12345 → 58048	[ACK] Seq=47 Ack=444 Win=65536 Len=0

בקשה לקובץ

html

שלא קיים

השרת מחזיר הודעה שהקובץ לא נמצא

ומבקש לסגור את החיבור עם close

2 Reassembled TCP Segments (45 bytes): #26(45), #28(0)]

ypertext Transfer Protocol

HTTP/1.1 404 Not Found\r\n

Connection: close\r\n

\r\n

[HTTP response 1/1]

48 54 54 50 2f 31 2e 31 20 34 30 34 20 4e 6f 74 43 6f 6e 6e 65 63 74 69
20 46 6f 75 6e 64 0d 0a 43 6f 6e 6e 65 63 74 69
6f 6e 3a 20 63 6c 6f 73 65 0d 0a 0d 0a

HTTP/1.1 404 Not Found - Connecti
on: close

25	1.540954461	127.0.0.1	127.0.0.1	TCP	68	12345 → 58048	[ACK] Seq=1 Ack=443 Win=65152 Len=0
26	1.541015816	127.0.0.1	127.0.0.1	TCP	113	12345 → 58048	[PSH, ACK] Seq=1 Ack=443 Win=65152 Len=0
27	1.541028803	127.0.0.1	127.0.0.1	TCP	68	58048 → 12345	[ACK] Seq=443 Ack=46 Win=65536 Len=0
28	1.541032744	127.0.0.1	127.0.0.1	HTTP	68	HTTP/1.1 404 Not Found	
29	1.541093065	127.0.0.1	127.0.0.1	TCP	68	58048 → 12345	[FIN, ACK] Seq=443 Ack=47 Win=0 Len=0
30	1.541096355	127.0.0.1	127.0.0.1	TCP	68	12345 → 58048	[ACK] Seq=47 Ack=444 Win=65536 Len=0

החבילה שהשרת שלח

מכילה את הבקשה

לסגור את החיבור

[Timestamps]

[SEQ/ACK analysis]

TCP payload (45 bytes)

[Reassembled PDU in frame: 28]

TCP segment data (45 bytes)

7f 00 00 01 30 39 e2 c0 0c 7a 25 85 f7 51 6b 42
80 18 02 00 fe 55 00 00 01 01 08 0a 01 4f 63 ed
01 4f 63 ed 48 54 54 50 2f 31 2e 31 20 34 30 34
20 4e 6f 74 20 46 6f 75 6e 64 0d 0a 43 6f 6e 6e
65 63 74 69 6f 6e 3a 20 63 6c 6f 73 65 0d 0a 0d
0a

...09...%..QkB
...U... ..0c
..0c..HTTP /1.1 404
Not Fou nd..Conn
ection: close...