VANIER COLLEGE – Computer Engineering Technology – Winter 2021

**Network Fundamentals (247-409-VA)**

Leonardo Fusser (1946995)

# LABORATORY EXPERIMENT 7

# Observing TCP and UDP

NOTE:
To be completed in one lab session of 3 hrs.
To be submitted using the typical lab format, one week later – **before 23:30,** of your respective lab session.
This exercise is to be done individually except where specified in the procedure. **Each** student must submit a lab report with original observations and conclusions.

## OBJECTIVES:

After performing this experiment, the student will be able to:

1. Explain common netstat command parameters and outputs.
2. Use netstat to examine protocol information on a host computer.
3. Identify TCP header fields and operation using a Wireshark FTP session capture.

## BACKGROUND

**netstat** is an abbreviation for the network statistics utility, available on both Windows and Unix / Linux computers. Passing optional parameters with the command will change output information. Netstat displays incoming and outgoing network connections (TCP and UDP), host computer routing table, and interface statistics.

During the life of a TCP connection, the connection passes through a series of states. The following table is a summary of TCP states, compiled from RFC 793, Transmission Control Protocol, September 1981, as reported by netstat:

| State | Connection Description |
|---|---|
| LISTEN | The local connection is waiting for a connection request from any remote device. |
| ESTABLISHED | The connection is open, and data may be exchanged through the connection. This is the normal state for the data-transfer phase of the connection. |
| TIME-WAIT | The local connection is waiting a default period of time after sending a connection termination request before closing the connection. This is a normal condition and will normally last between 30 and 120 seconds. |
| CLOSE-WAIT | The connection is closed, but is waiting for a termination request from the local user. |
| SYN-SENT | The local connection is waiting for a response after sending a connection request. The connection should transition quickly through this state. |
| SYN_RECEIVED | The local connection is waiting for a confirming connection request acknowledgment. The connection should transition quickly through this state. Multiple connections in SYN_RECEIVED state may indicate a TCP SYN attack. |

IP address displayed by netstat fall into several categories, as shown below.

| IP Address | Description |
|---|---|
| 127.0.0.1 | This address refers to the local host, or this computer. |
| 0.0.0.0 | A global address, meaning any. |
| Remote Address | The address of the remote device that has a connection with this computer. |

The two protocols in the TCP/IP Transport Layer are the transmission control protocol (TCP), defined in RFC 761, January 1980, and user datagram protocol (UDP), defined in RFC 768, August 1980. Both protocols support upper-layer protocol communication. For example, TCP is used to provide Transport Layer support for the HTTP and FTP protocols, among others. UDP provides Transport Layer support for domain name services (DNS) and trivial file transfer protocol (TFTP), among others.

TCP segment fields:

TCP Segment

<table>
<tr><th>0</th><th>4</th><th>10</th><th>16</th><th>24 31</th></tr>
<tr><td colspan="3">TCP Source Port Number</td><td colspan="2">TCP Destination Port Number</td></tr>
<tr><td colspan="5">Sequence Number</td></tr>
<tr><td colspan="5">Acknowledgment Number</td></tr>
<tr><td>HLEN</td><td>Reserved</td><td>Code Bits</td><td colspan="2">Window</td></tr>
<tr><td colspan="3">TCP Checksum</td><td colspan="2">Urgent Pointer</td></tr>
<tr><td colspan="4">Options (If Any)</td><td>Padding</td></tr>
<tr><td colspan="5">Data</td></tr>
<tr><td colspan="5">Data...</td></tr>
</table>

| Code Bits: | URG | ACK | RST | PSH | SYN | FIN |
|---|---|---|---|---|---|---|

## PROCEDURE

*** The lab will be using the same Eagle-server Topology Diagram as in our previous labs. If you have changed your computer, please modify your network IP address accordingly and ensure all other network connections have been disabled to avoid unnecessary conflicts.*

### Part A: Understanding netstat command parameters and outputs.

1. Open a terminal window and display help information about the netstat command, use the /? Options. Fill in the appropriate option that best matches the description:

| Option | Description |
|---|---|
| -a | Display all connections and listening ports. |
| -n | Display addresses and port numbers in numerical form. |
| interval | Redisplay statistics every five seconds. Press CTRL+C to stop redisplaying statistics. |
| -p proto | Shows connections for the protocol specified by proto; proto may be any of: TCP, UDP, TCPv6, or UDPv6. If used with the –s option to display per-protocol statistics, proto may be any of: IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6. |
| -a 30 | Redisplay all connections and listening ports every 30 seconds. |

2. Use netstat to view existing connections.

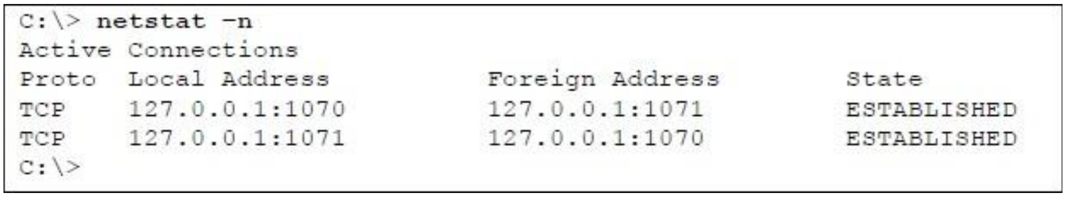
   a. Issue the command with option "-a". Then, issue the command again with option "-an" that display output in raw format. You can use 2 windows to compare easier. Compare the outputs, and comment on what are the differences.

      ➢ When using the netstat command with option "-a", all the display connections are shown (along with the ports), whereas when using the netstat command with option "-an", the output is only in numerical form.

   b. Write down 3 TCP and 3 UDP connections from the netstat –a output, and the corresponding translated port numbers from netstat –an output. If there are fewer than 3 connections that translate, note that in your table.

| Connection | option used | Proto | Local Address | Foreign Address | State |
|---|---|---|---|---|---|
| TCP | -a | TCP | 0.0.0.0:135 | Leonardo:0 | Listening |
|  | -an | TCP | 0.0.0.0:49668 | 0.0.0.0:0 | Listening |
| TCP | -a | TCP | 0.0.0.0:445 | Leonardo:0 | Listening |
|  | -an | TCP | 0.0.0.0:49669 | 0.0.0.0:0 | Listening |
| TCP | -a | TCP | 0.0.0.0:3389 | Leonardo:0 | Listening |
|  | -an | TCP | 0.0.0.0:49670 | 0.0.0.0:0 | Listening |
| UDP | -a | UDP | 0.0.0.0:500 | *;* | N/A |
|  | -an | UDP | 0.0.0.0:49295 | *;* | N/A |
| UDP | -a | UDP | 0.0.0.0:3389 | *;* | N/A |
|  | -an | UDP | 0.0.0.0:52721 | *;* | N/A |
| UDP | -a | UDP | 0.0.0.0:3702 | *;* | N/A |
|  | -an | UDP | 0.0.0.0:53936 | *;* | N/A |

3. Refer to the following netstat output. A new network engineer suspects that his host computer has been compromised by an outside attack against ports 1070 and 1071. How would you respond?



```
C:\> netstat -n
Active Connections
Proto  Local Address            Foreign Address          State
TCP    127.0.0.1:1070           127.0.0.1:1071           ESTABLISHED
TCP    127.0.0.1:1071           127.0.0.1:1070           ESTABLISHED
C:\>
```

➢ I would respond that there is no reason to worry. The reason is because what is represented above does not mean that there is a hacker trying to compromise the network engineer's computer. What is occurring is just his computer performing a network test on itself (127.0.0.1 is not an external address, it is the loopback address for his computer), therefore there is no need to suspect that a hacker is trying to compromise the network engineer's computer.

4. Establish multiple concurrent TCP connections and record netstat output.

In this task, several simultaneous connections will be made with Eagle Server. The telnet command is used to access Eagle Server network services, thus providing several protocols to examine with netstat.

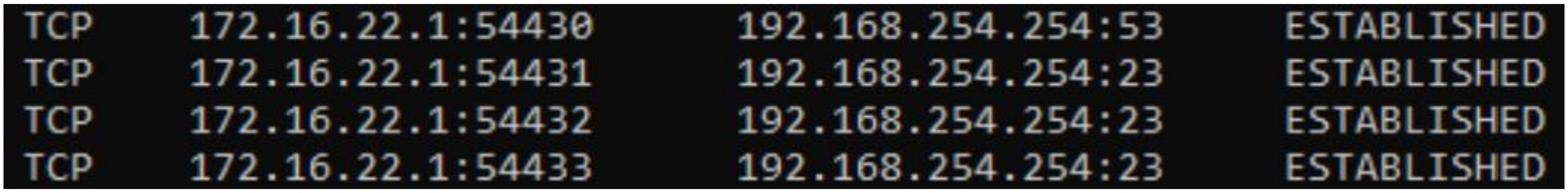
Several network services on Eagle Server will respond to a telnet connection. In this lab, you will use:

- **DNS** – Domain Name Server, port 53
- **FTP** – FTP server, port 21
- **SMTP** – SMTP mail server, port 25
- **TELNET** – Telnet server, port 23

a. Open 4 command windows. In the first telnet terminal window, telnet to Eagle Server on port 53. In the last windows, telnet on port 23. Example of command is shown below:

```
C:\> telnet eagle-server.example.com 53
```

b. In the large terminal window, record established connection with Eagle Server*. (You should observe 4 established connections)*



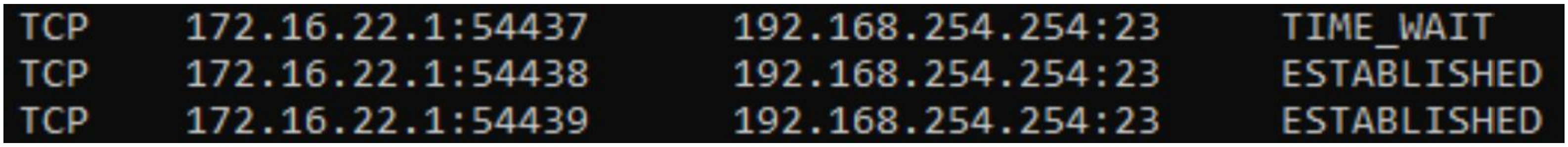
```
TCP    172.16.22.1:54430    192.168.254.254:53    ESTABLISHED
TCP    172.16.22.1:54431    192.168.254.254:23    ESTABLISHED
TCP    172.16.22.1:54432    192.168.254.254:23    ESTABLISHED
TCP    172.16.22.1:54433    192.168.254.254:23    ESTABLISHED
```

*Multiple telnet connections to eagle server port 23 and one to port 53 shown above. All connections indicate "established", even the one with the wrong port number (53 instead of 23).*

c. Why would telnet to UDP ports fail?

- Telnet to UDP ports will always fail simply because the telnet command only operates using the TCP protocol. UDP testing cannot be done with the telnet command.

5. Closed Established session abruptly (close the terminal window), and issue the netstat –an command. Try to view connections in stages different from ESTABLISHED. Record your observation.



```
TCP    172.16.22.1:54437    192.168.254.254:23    TIME_WAIT
TCP    172.16.22.1:54438    192.168.254.254:23    ESTABLISHED
TCP    172.16.22.1:54439    192.168.254.254:23    ESTABLISHED
```

*Multiple telnet connections to eagle server port 23. Two of the connections indicate "established" and only one that indicates "time_wait" (the one that closed abruptly).*

## Part B: Identify TCP Header Fields and Operation in FTP Session

TCP sessions are well controlled and managed by information exchanged in the TCP header fields. In this task, an FTP session will be made to Eagle Server. When finished, the session capture will be analyzed.

6. Capture an FTP session.
   a. Start your Wireshark capture session on your intended interface.
   b. Start an FTP connection to Eagle Server. Enter the following command:

      ```
      ftp eagle-server.example.com
      ```

   c. When prompted for a user ID, type *anonymous*. When prompted for a password, press <Enter>.

   d. Change the FTP directory to /pub/eagle_labs/eagle1/chapter4/:

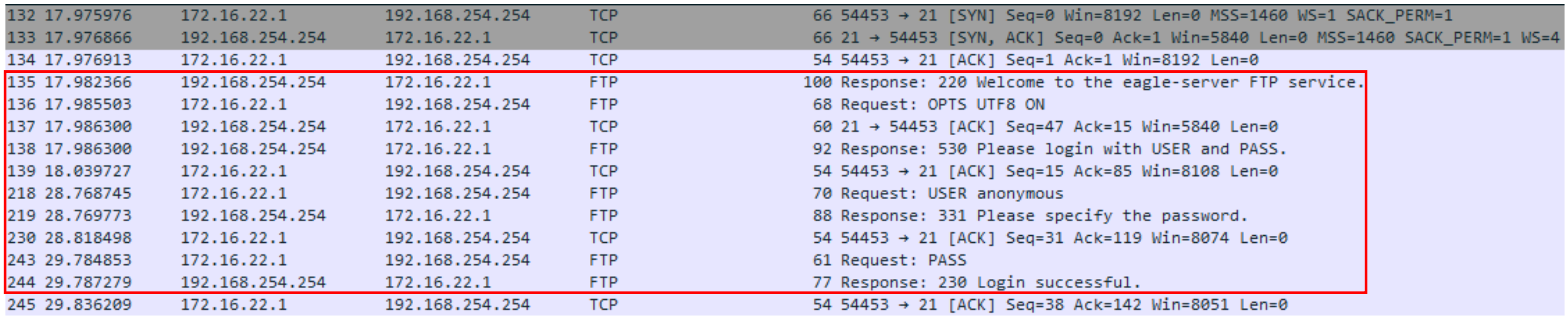
      ```
      ftp> cd /pub/eagle_labs/eagle1/chapter4/
      ```

   e. Download the file s1-central:

      ```
      ftp> get s1-central
      ```

   f. When finished, terminate the FTP sessions with the **quit** command. Close the command-line window and stop the Wireshark capture.

7. Analyze the TCP fields
   a. When the FTP client is connected to the FTP server, the transport layer protocol TCP created a reliable session. Identify the packets used to create a reliable session. Explain and annotate the details of the data segment from your Wireshark capture.



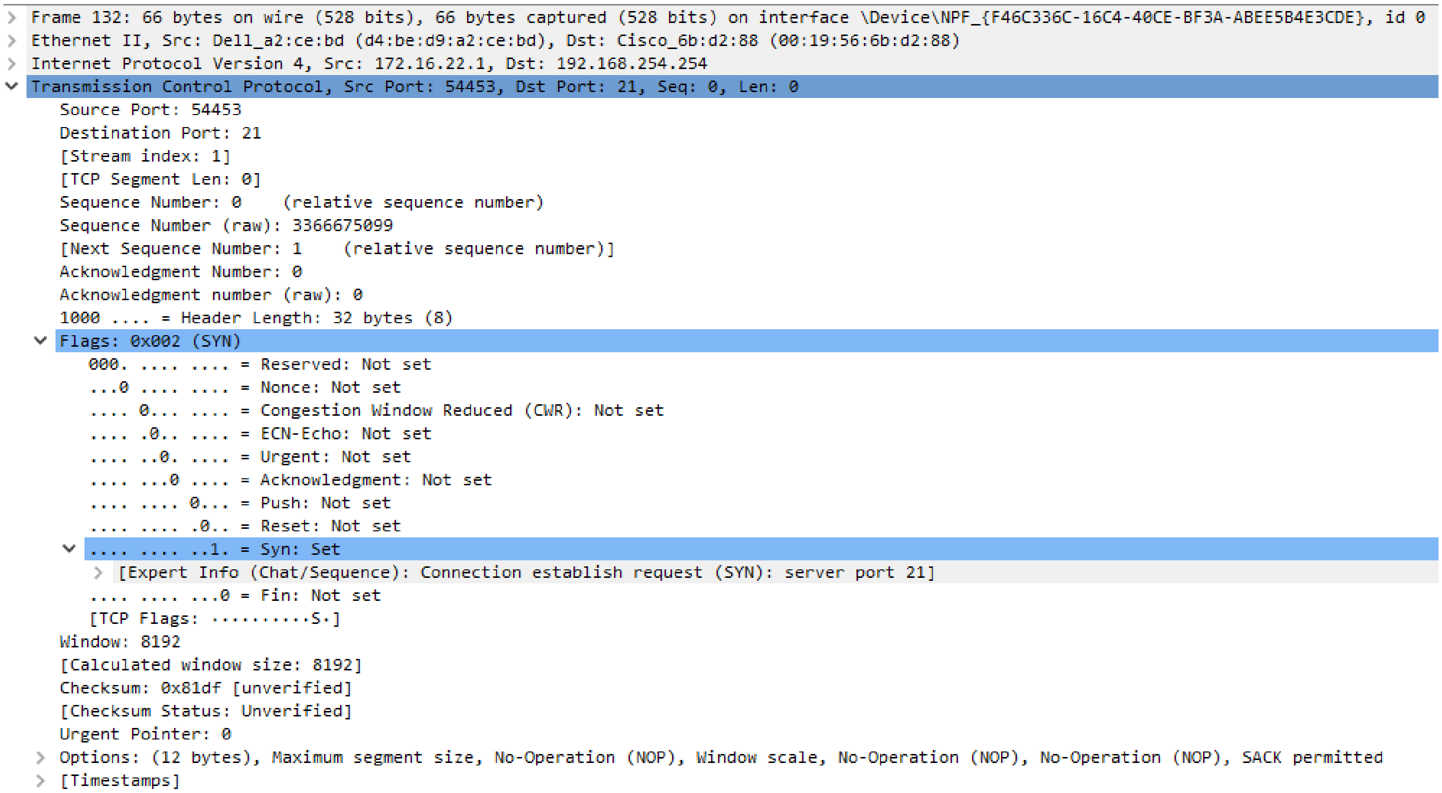
```
132 17.975976    172.16.22.1        192.168.254.254    TCP              66 54453 → 21 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=1 SACK_PERM=1
133 17.976866    192.168.254.254    172.16.22.1        TCP              66 21 → 54453 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=4
134 17.976913    172.16.22.1        192.168.254.254    TCP              54 54453 → 21 [ACK] Seq=1 Ack=1 Win=8192 Len=0
135 17.982366    192.168.254.254    172.16.22.1        FTP             100 Response: 220 Welcome to the eagle-server FTP service.
136 17.985503    172.16.22.1        192.168.254.254    FTP              68 Request: OPTS UTF8 ON
137 17.986300    192.168.254.254    172.16.22.1        TCP              60 21 → 54453 [ACK] Seq=47 Ack=15 Win=5840 Len=0
138 17.986300    192.168.254.254    172.16.22.1        FTP              92 Response: 530 Please login with USER and PASS.
139 18.039727    172.16.22.1        192.168.254.254    TCP              54 54453 → 21 [ACK] Seq=15 Ack=85 Win=8108 Len=0
218 28.768745    172.16.22.1        192.168.254.254    FTP              70 Request: USER anonymous
219 28.769773    192.168.254.254    172.16.22.1        FTP              88 Response: 331 Please specify the password.
230 28.818498    172.16.22.1        192.168.254.254    TCP              54 54453 → 21 [ACK] Seq=31 Ack=119 Win=8074 Len=0
243 29.784853    172.16.22.1        192.168.254.254    FTP              61 Request: PASS
244 29.787279    192.168.254.254    172.16.22.1        FTP              77 Response: 230 Login successful.
245 29.836209    172.16.22.1        192.168.254.254    TCP              54 54453 → 21 [ACK] Seq=38 Ack=142 Win=8051 Len=0
```

*Shown in read above are all the packets created to create a reliable session between the user and the FTP server. There are few steps ranging from the welcome banner (beginning of the login session) to the actual message from the server indicating to the user they have logged in successfully (end of the login session). A complete explanation of how this session is created can be found at the end of this report. The portion where the file is retrieved is not shown here but in the next few pages.*

b. Using the Wireshark capture of the first TCP session start-up (SYN bit set to 1), fill in information about the TCP header in the table below.

TCP Header: SYN bit set to 1.

| Parameters/Fields | Answer |
|---|---|
| Source IP address | 172.16.22.1 |
| Destination IP address | 192.168.254.254 |
| Source port number | 54453 |
| Destination port number | 21 (FTP) |
| Sequence number | 0 (relative sequence number) |
| Acknowledge number | 0 |
| Header length | 32 bytes |
| Window size | 8192 |



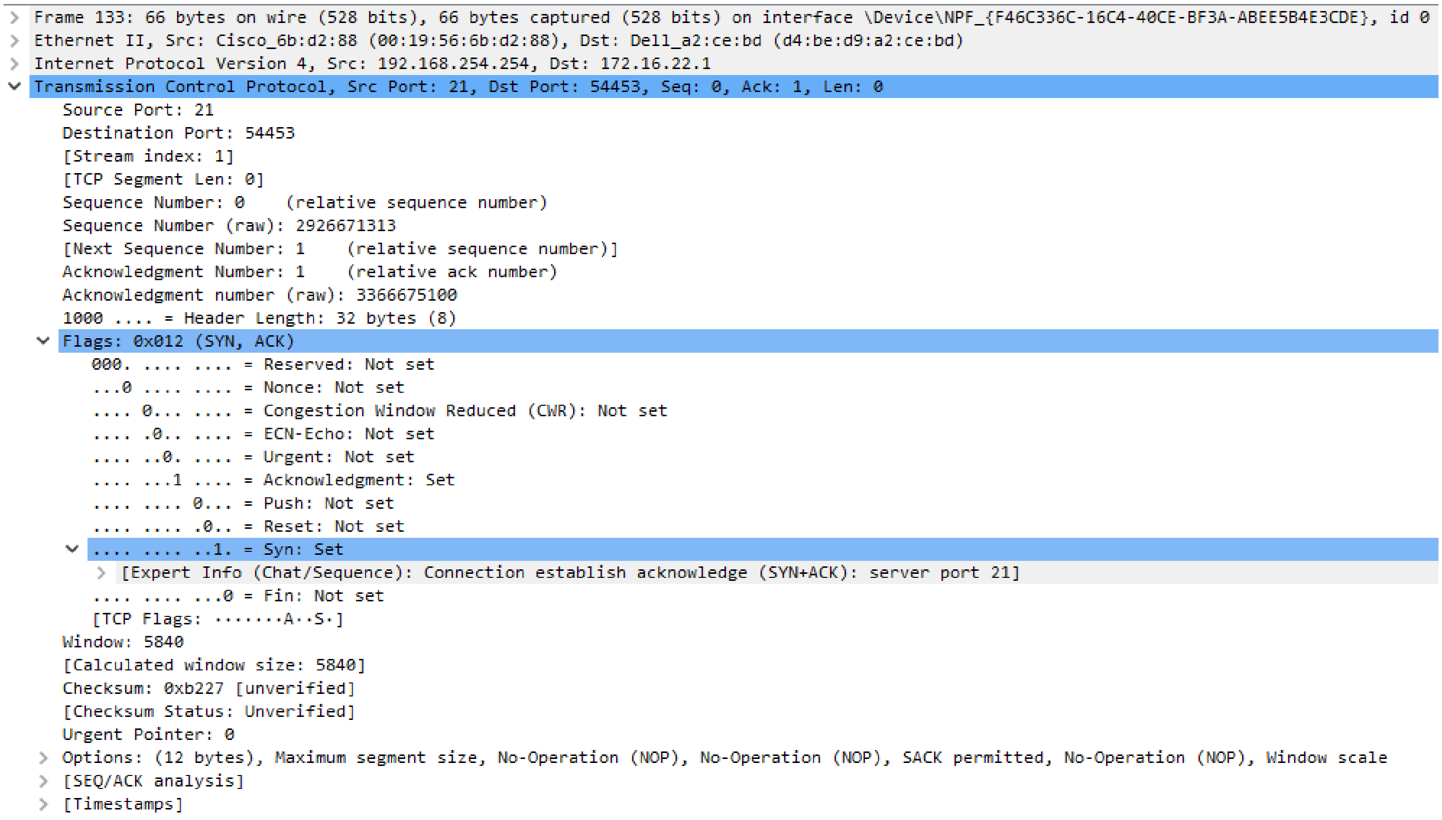
```
> Frame 132: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{F46C336C-16C4-40CE-BF3A-ABEE5B4E3CDE}, id 0
> Ethernet II, Src: Dell_a2:ce:bd (d4:be:d9:a2:ce:bd), Dst: Cisco_6b:d2:88 (00:19:56:6b:d2:88)
> Internet Protocol Version 4, Src: 172.16.22.1, Dst: 192.168.254.254
v Transmission Control Protocol, Src Port: 54453, Dst Port: 21, Seq: 0, Len: 0
    Source Port: 54453
    Destination Port: 21
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 3366675099
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1000 .... = Header Length: 32 bytes (8)
  v Flags: 0x002 (SYN)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...0 .... = Acknowledgment: Not set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    v .... .... ..1. = Syn: Set
      > [Expert Info (Chat/Sequence): Connection establish request (SYN): server port 21]
      .... .... ...0 = Fin: Not set
      [TCP Flags: ··········S·]
    Window: 8192
    [Calculated window size: 8192]
    Checksum: 0x81df [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
  > [Timestamps]
```

*Screenshot in relation to question 7b.*

c. Using the Wireshark capture of the first TCP session start-up (SYN and ACK bits set to 1), fill in information about the TCP header in the table below.

TCP Header: SYN and ACK Bits set to 1.

| Parameters/Fields | Answer |
|---|---|
| Source IP address | 192.168.254.254 |
| Destination IP address | 172.16.22.1 |
| Source port number | 21 (FTP) |
| Destination port number | 54453 |
| Sequence number | 0 (relative sequence number) |
| Acknowledge number | 1 (relative ack number) |
| Header length | 32 bytes |
| Window size | 5840 |



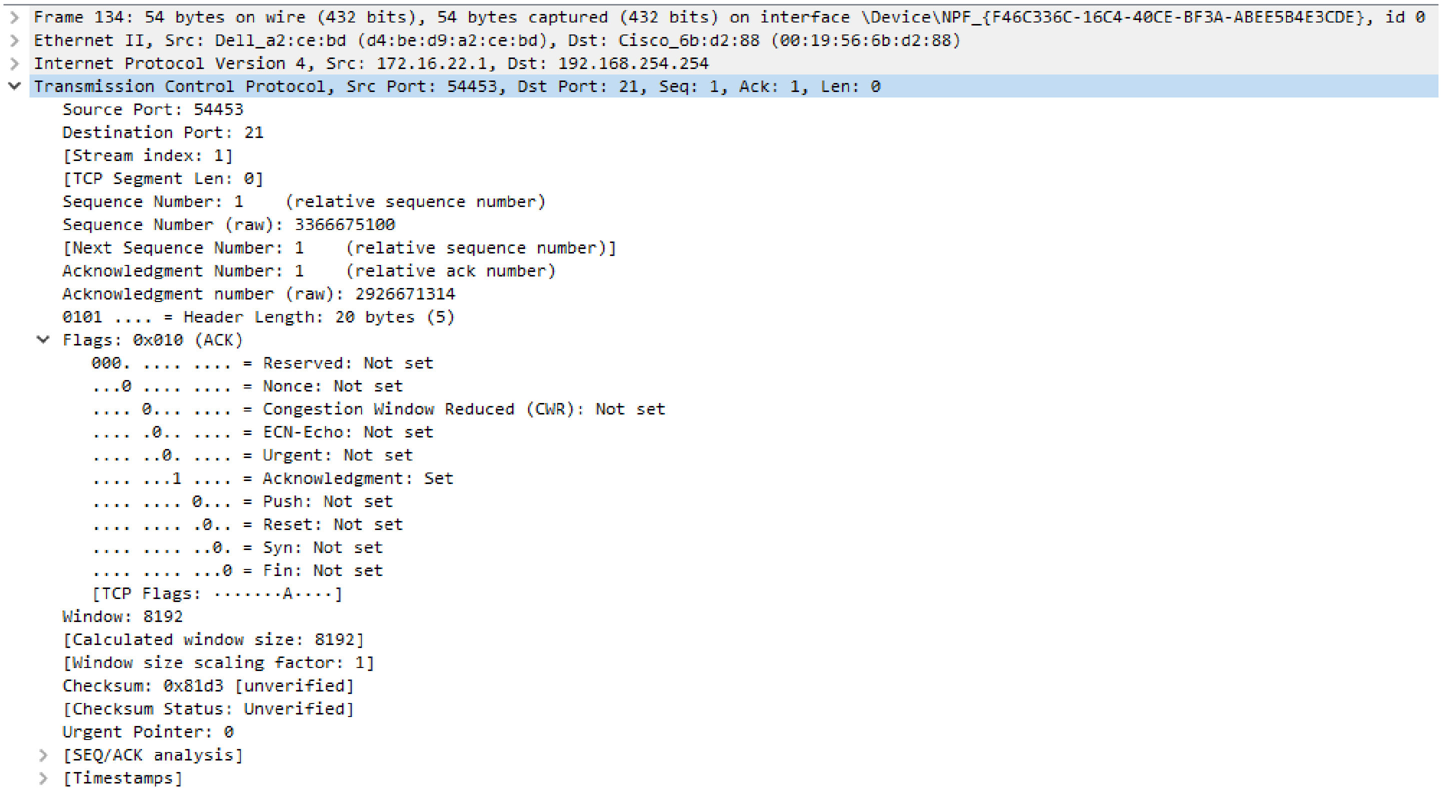
```
> Frame 133: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{F46C336C-16C4-40CE-BF3A-ABEE5B4E3CDE}, id 0
> Ethernet II, Src: Cisco_6b:d2:88 (00:19:56:6b:d2:88), Dst: Dell_a2:ce:bd (d4:be:d9:a2:ce:bd)
> Internet Protocol Version 4, Src: 192.168.254.254, Dst: 172.16.22.1
v Transmission Control Protocol, Src Port: 21, Dst Port: 54453, Seq: 0, Ack: 1, Len: 0
    Source Port: 21
    Destination Port: 54453
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 2926671313
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 3366675100
    1000 .... = Header Length: 32 bytes (8)
  v Flags: 0x012 (SYN, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
    v .... .... ..1. = Syn: Set
      > [Expert Info (Chat/Sequence): Connection establish acknowledge (SYN+ACK): server port 21]
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·······A··S·]
    Window: 5840
    [Calculated window size: 5840]
    Checksum: 0xb227 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, No-Operation (NOP), Window scale
  > [SEQ/ACK analysis]
  > [Timestamps]
```

*Screenshot in relation to question 7c.*

d. Using the Wireshark capture of the first TCP session start-up (only ACK bit set to 1), fill in information about the TCP header in the table below.

TCP Header: ACK Bit set to 1.

| Parameters/Fields | Answer |
|---|---|
| Source IP address | 172.16.22.1 |
| Destination IP address | 192.168.254.254 |
| Source port number | 54453 |
| Destination port number | 21 (FTP) |
| Sequence number | 1 (relative sequence number) |
| Acknowledge number | 1 (relative ack number) |
| Header length | 20 bytes |
| Window size | 8192 |



```
> Frame 134: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{F46C336C-16C4-40CE-BF3A-ABEE5B4E3CDE}, id 0
> Ethernet II, Src: Dell_a2:ce:bd (d4:be:d9:a2:ce:bd), Dst: Cisco_6b:d2:88 (00:19:56:6b:d2:88)
> Internet Protocol Version 4, Src: 172.16.22.1, Dst: 192.168.254.254
v Transmission Control Protocol, Src Port: 54453, Dst Port: 21, Seq: 1, Ack: 1, Len: 0
    Source Port: 54453
    Destination Port: 21
    [Stream index: 1]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 3366675100
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 2926671314
    0101 .... = Header Length: 20 bytes (5)
  v Flags: 0x010 (ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
        [TCP Flags: ·······A····]
    Window: 8192
    [Calculated window size: 8192]
    [Window size scaling factor: 1]
    Checksum: 0x81d3 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
```

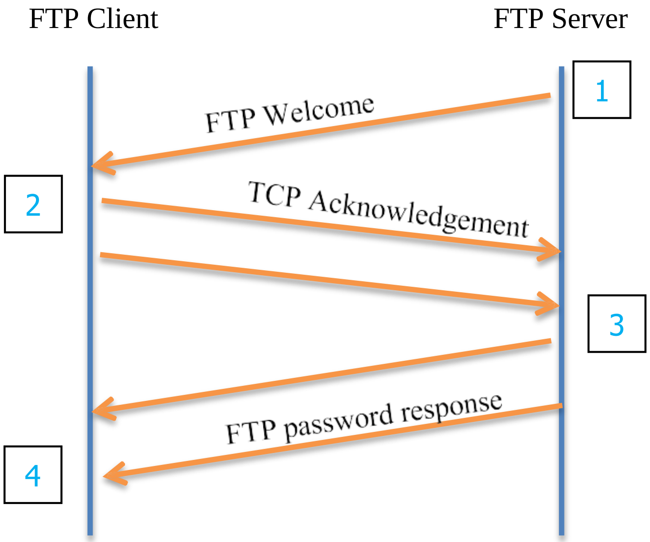
*Screenshot in relation to question 7d.*

e. Ignoring the TCP session started when a data transfer occurred (as examined above), how many other TCP datagram contained a SYN bit?

- There are no other TCP datagrams that contain a SYN bit.

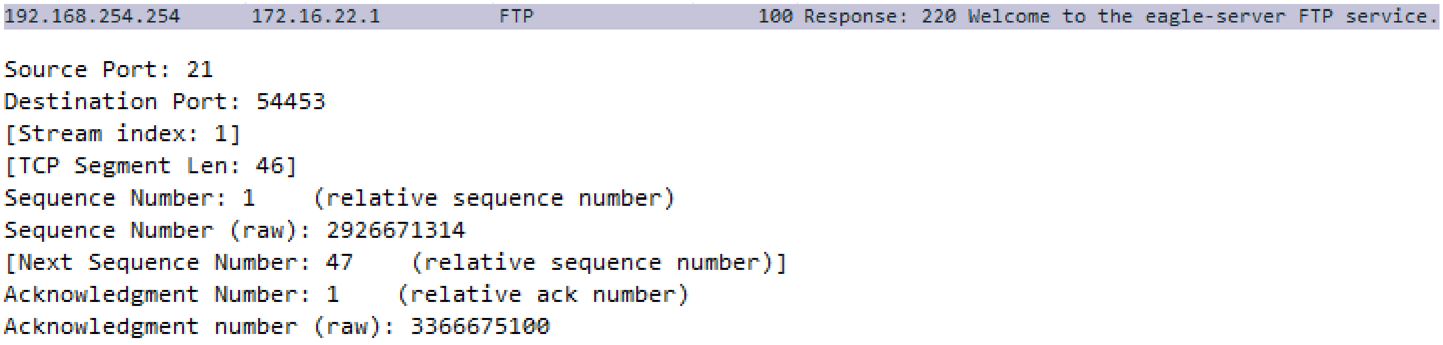
8. TCP session management

   a. The FTP client and server communicate between each other, unaware and uncaring that TCP has control and management over the session. When the FTP server sends a Response: 220 to the FTP client, the TCP session on the FTP client sends an acknowledgment to the TCP session on Eagle Server. Complete the diagram below by annotating the sequence based on your Wireshark capture.



*Pay attention to sequence number and acknowledgement numbers below.

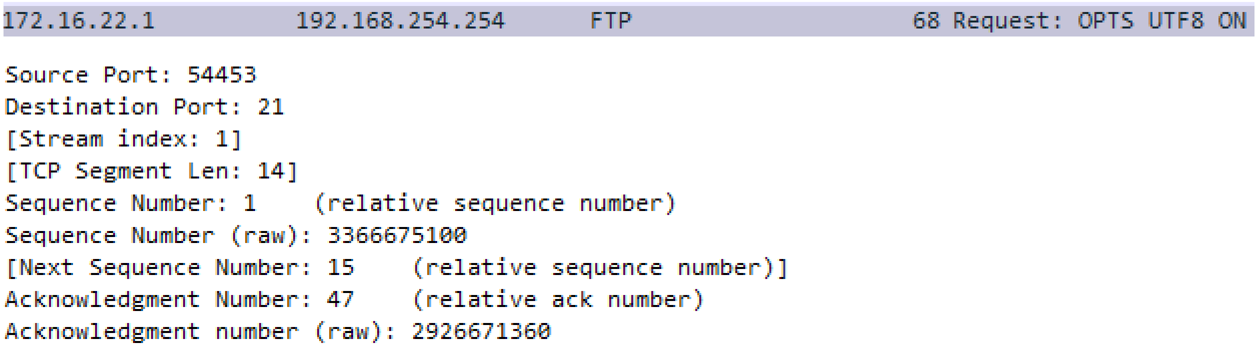
1



```
192.168.254.254        172.16.22.1        FTP        100 Response: 220 Welcome to the eagle-server FTP service.

Source Port: 21
Destination Port: 54453
[Stream index: 1]
[TCP Segment Len: 46]
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 2926671314
[Next Sequence Number: 47    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 3366675100
```

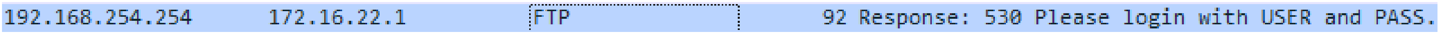
2



```
172.16.22.1          192.168.254.254      FTP              68 Request: OPTS UTF8 ON

Source Port: 54453
Destination Port: 21
[Stream index: 1]
[TCP Segment Len: 14]
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 3366675100
[Next Sequence Number: 15    (relative sequence number)]
Acknowledgment Number: 47    (relative ack number)
Acknowledgment number (raw): 2926671360
```

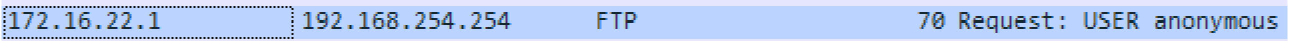
3



```
192.168.254.254      172.16.22.1          FTP           92 Response: 530 Please login with USER and PASS.
```

```
Source Port: 21
Destination Port: 54453
[Stream index: 1]
[TCP Segment Len: 38]
Sequence Number: 47    (relative sequence number)
Sequence Number (raw): 2926671360
[Next Sequence Number: 85    (relative sequence number)]
Acknowledgment Number: 15    (relative ack number)
Acknowledgment number (raw): 3366675114
```

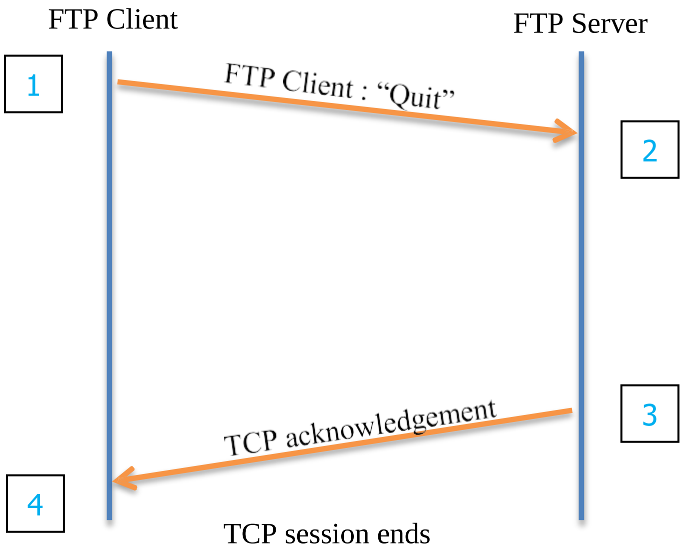
4



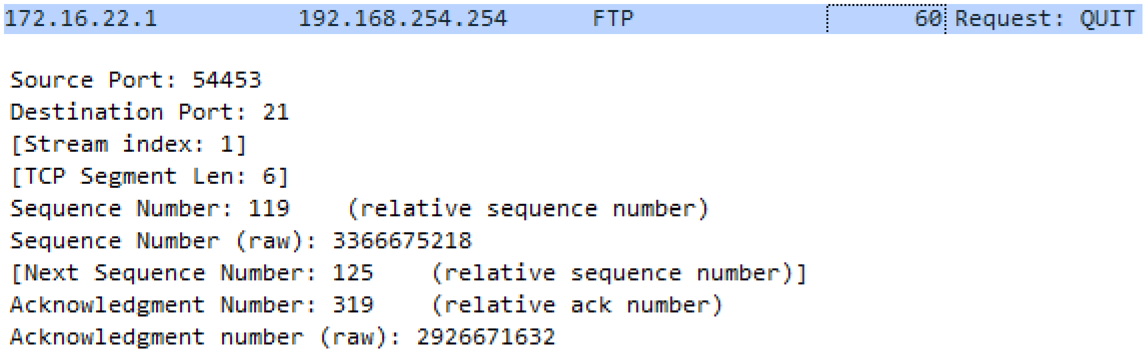
```
172.16.22.1          192.168.254.254      FTP                 70 Request: USER anonymous
```

```
Source Port: 54453
Destination Port: 21
[Stream index: 1]
[TCP Segment Len: 16]
Sequence Number: 15    (relative sequence number)
Sequence Number (raw): 3366675114
[Next Sequence Number: 31    (relative sequence number)]
Acknowledgment Number: 85    (relative ack number)
Acknowledgment number (raw): 2926671398
```

b. When the FTP session has finished, the FTP client sends a command to “quit.” The FTP server acknowledges the FTP termination with a Response: 221 Goodbye. Complete the transaction diagram below based on your Wireshark capture.



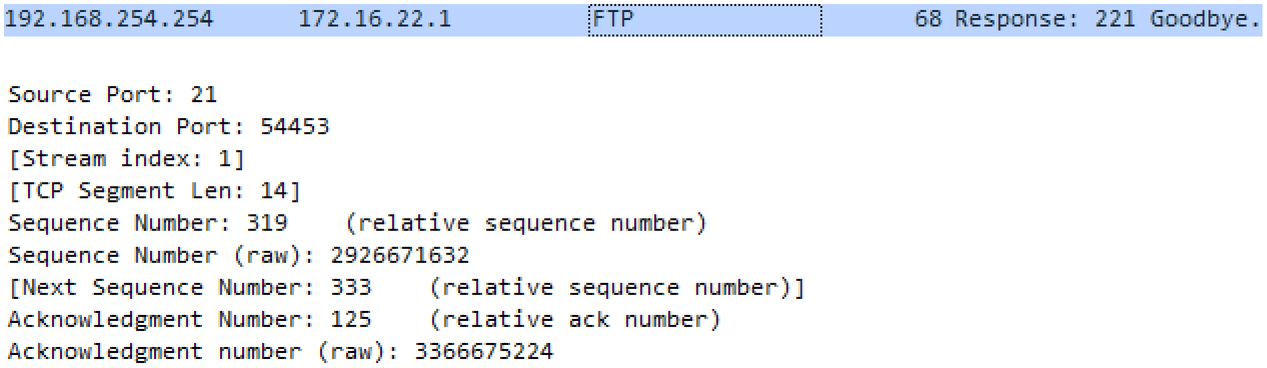
1



```
172.16.22.1          192.168.254.254       FTP              60 Request: QUIT

Source Port: 54453
Destination Port: 21
[Stream index: 1]
[TCP Segment Len: 6]
Sequence Number: 119    (relative sequence number)
Sequence Number (raw): 3366675218
[Next Sequence Number: 125    (relative sequence number)]
Acknowledgment Number: 319    (relative ack number)
Acknowledgment number (raw): 2926671632
```

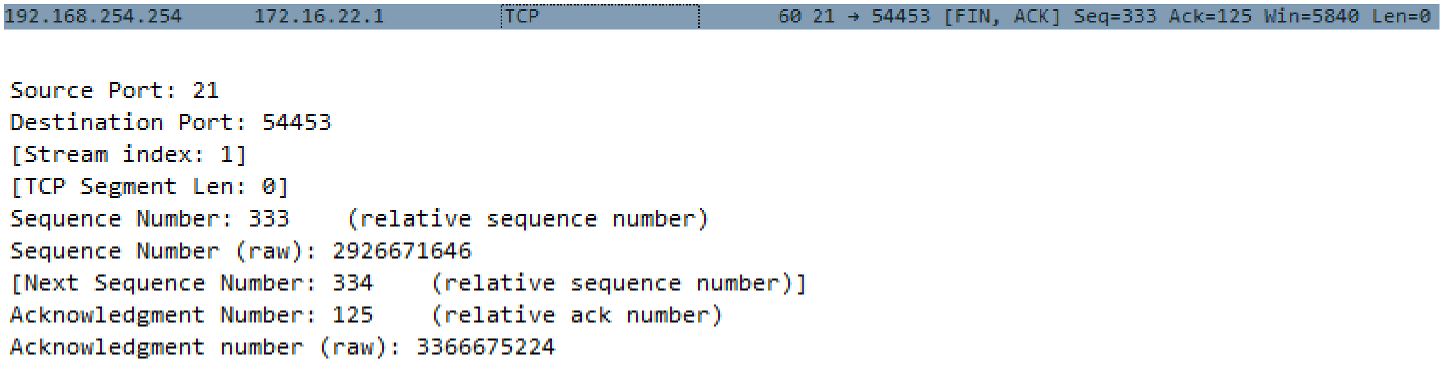
2



```
192.168.254.254      172.16.22.1           FTP              68 Response: 221 Goodbye.

Source Port: 21
Destination Port: 54453
[Stream index: 1]
[TCP Segment Len: 14]
Sequence Number: 319    (relative sequence number)
Sequence Number (raw): 2926671632
[Next Sequence Number: 333    (relative sequence number)]
Acknowledgment Number: 125    (relative ack number)
Acknowledgment number (raw): 3366675224
```

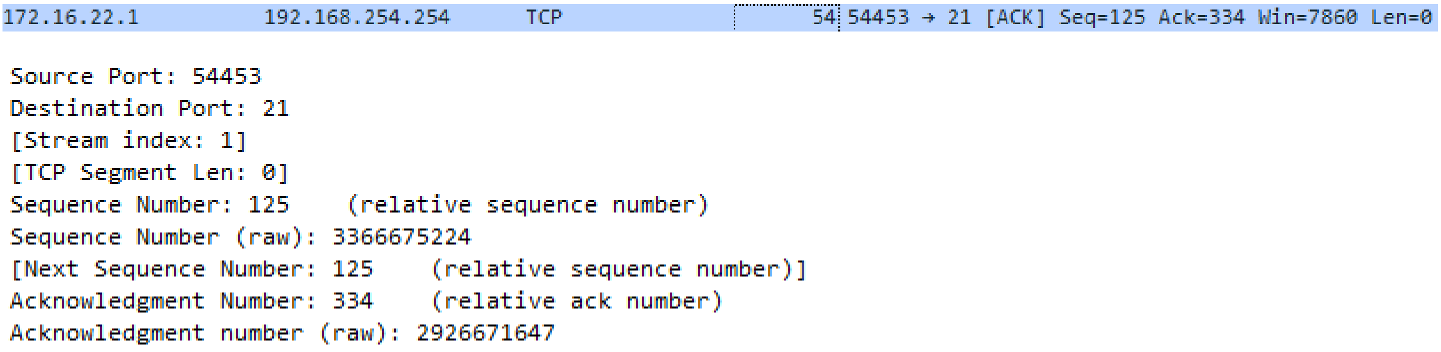
3



```
192.168.254.254      172.16.22.1       TCP            60 21 → 54453 [FIN, ACK] Seq=333 Ack=125 Win=5840 Len=0

Source Port: 21
Destination Port: 54453
[Stream index: 1]
[TCP Segment Len: 0]
Sequence Number: 333    (relative sequence number)
Sequence Number (raw): 2926671646
[Next Sequence Number: 334    (relative sequence number)]
Acknowledgment Number: 125    (relative ack number)
Acknowledgment number (raw): 3366675224
```

4



```
172.16.22.1          192.168.254.254       TCP          54 54453 → 21 [ACK] Seq=125 Ack=334 Win=7860 Len=0

Source Port: 54453
Destination Port: 21
[Stream index: 1]
[TCP Segment Len: 0]
Sequence Number: 125    (relative sequence number)
Sequence Number (raw): 3366675224
[Next Sequence Number: 125    (relative sequence number)]
Acknowledgment Number: 334    (relative ack number)
Acknowledgment number (raw): 2926671647
```