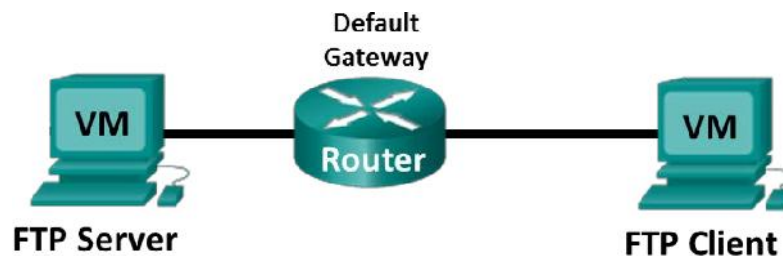


10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures



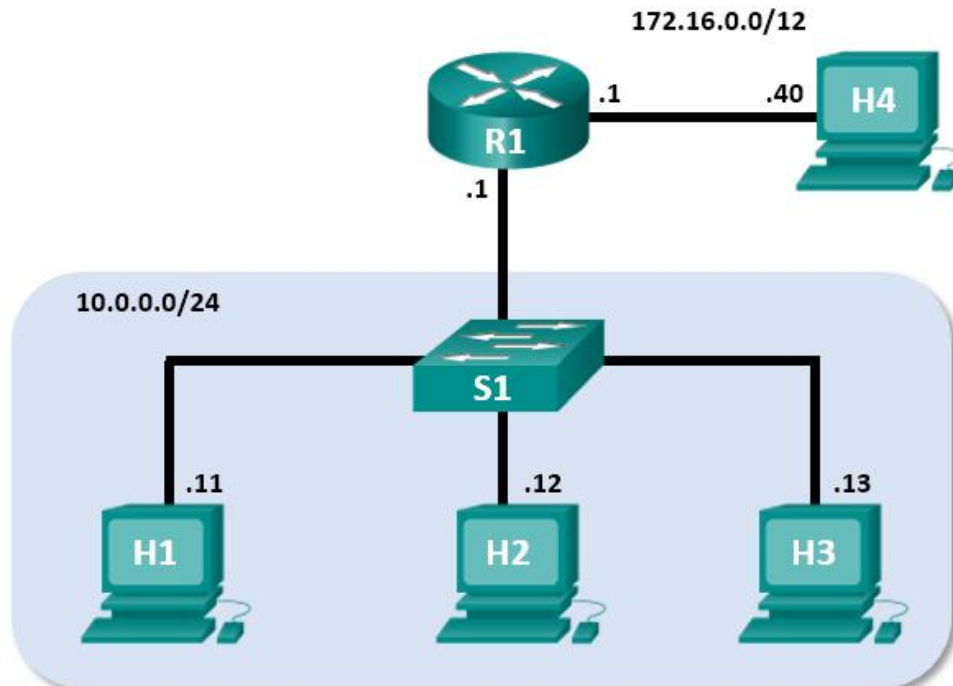
This lab has been updated for use on NETLAB+.
www.netdevgroup.com

Topology - Part 1 (FTP)



Part 1 will highlight a TCP capture of an FTP session.

Mininet Topology - Part 2 (TFTP)



Part 2 will highlight a UDP capture of a TFTP session using the hosts in Mininet.

Objectives

Part 1: Identify TCP Header Fields and Operation Using a Wireshark FTP Session Capture

Part 2: Identify UDP Header Fields and Operation Using a Wireshark TFTP Session Capture

Background / Scenario

Two protocols in the TCP/IP transport layer are TCP (defined in RFC 761) and UDP (defined in RFC 768). Both protocols support upper-layer protocol communication. For example, TCP is used to provide transport layer support for the HyperText Transfer Protocol (HTTP) and FTP protocols, among others. UDP provides transport layer support for the Domain Name System (DNS) and TFTP, among others.

In Part 1 of this lab, you will use the Wireshark open source tool to capture and analyze TCP protocol header fields for FTP file transfers between the host computer and an anonymous FTP server. The terminal command line is used to connect to an anonymous FTP server and download a file. In Part 2 of this lab, you will use Wireshark to capture and analyze UDP header fields for TFTP file transfers between two Mininet host computers.

Instructions

Part 1: Identify TCP Header Fields and Operation Using a Wireshark FTP Session Capture

In Part 1, you use *Wireshark* to capture an *FTP* session and inspect *TCP* header fields.

Step 1: Start a Wireshark capture.

- Start and log into the **Security Onion** VM. Enter the username **analyst** password **cyberops** to login.
- Right-click an empty space on desktop, select **Open Terminal** to open a terminal. Then start **Wireshark** using the following command.

```
[analyst@secOps ~]$ sudo wireshark
```

- Start a Wireshark capture for the **eth1** interface.
- Start a new terminal window to access an external ftp site. Enter **ftp -p 192.168.0.14** at the prompt. Log into the FTP site on Workstation VM with user **anonymous** and no password.

```
[analyst@secOps ~]$ ftp -p 192.168.0.14
Connected to 192.168.0.14.
220 Welcome to Cisco CyberOPS VM FTP service.
Name (192.168.0.14:analyst): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Step 2: Download the Readme file.

- Locate and download the Readme file by entering the **ls** command to list the files.

```
ftp> ls
227 Entering Passive Mode (192,168,0,14,220,105).
150 Here comes the directory listing.
```

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

```
-rw-r--r--  1 0    0          32 Aug  10  16:14 Readme
-rw-r--r--  1 0    0          0 Mar  26   2018 ftp_test
226 Directory send OK.
```

Note: You may receive the following messages:

```
421 Service not available, remote server has closed connection
ftp: No control connection for command
```

```
501 Server cannot access argument
500 command not understood
ftp: bind: Address already in use
```

If this happens, then the FTP server is currently down. You need to go to **Workstation** VM to manually restart the FTP server by entering the **sudo systemctl restart vsftpd** command in a terminal window. Enter **cyberops** as password if prompt.

- b. Enter the command **get Readme** to download the file. When the download is complete, enter the command **quit** to exit. (**Note:** If you are unable to download the file, you can proceed with the rest of the lab.)

```
ftp> get Readme
local: Readme remote: Readme
227 Entering Passive Mode (192,168,0,14,185,36).
150 Opening BINARY mode data connection for Readme (32 bytes).
226 Transfer complete.
32 bytes received in 0.00 secs (315.6566 kB/s)
ftp> quit
221 Goodbye.
```

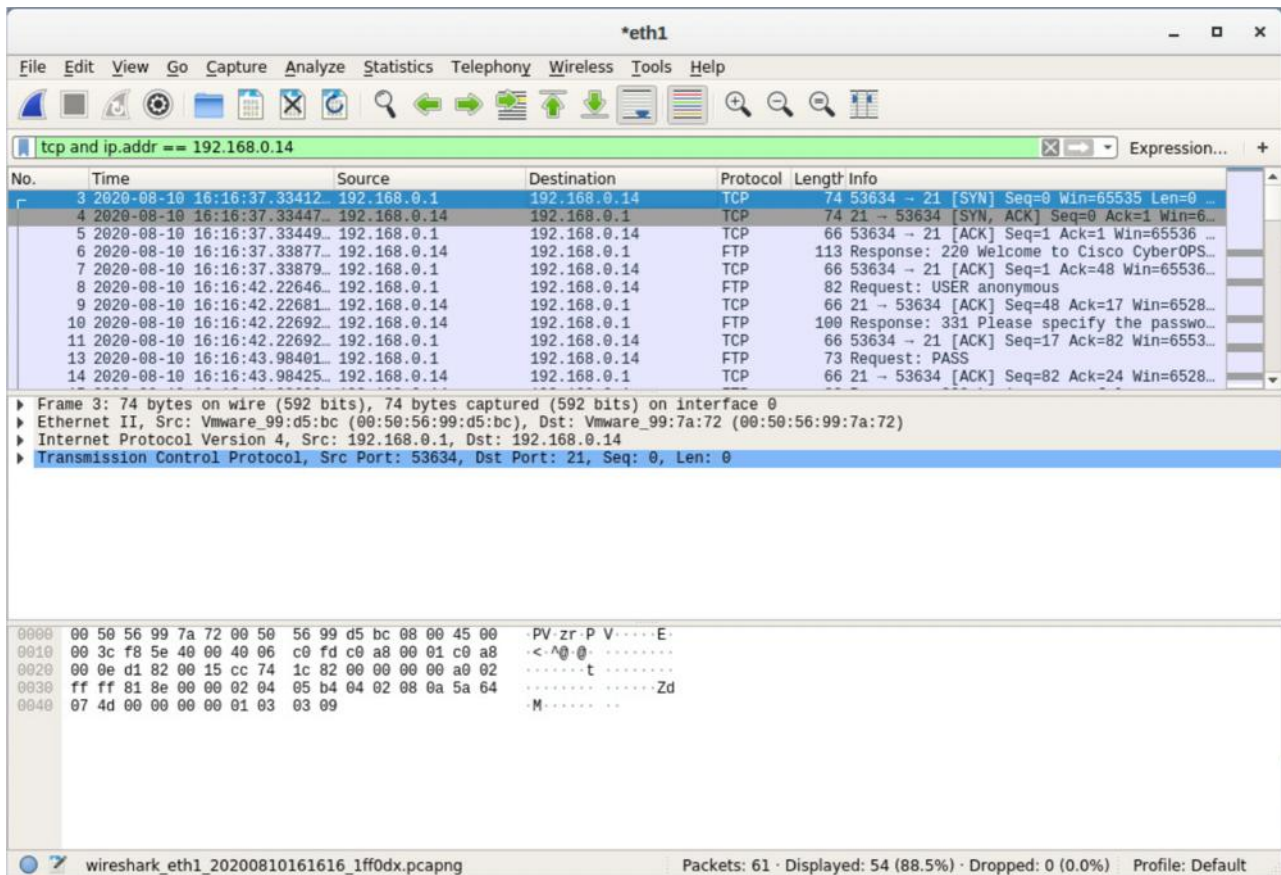
Step 3: Stop the Wireshark capture.

Step 4: View the Wireshark main window.

Wireshark captured many packets during the FTP session to **ftp.cdc.gov**. To limit the amount of data for analysis, apply the filter **tcp and ip.addr == 192.168.0.14** and press **Enter** to apply.

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

Note: The IP address, 192.168.0.14, is the address for **Workstation VM**.



Note: Your Wireshark interface may look slightly different than the above image.

Step 5: Analyze the TCP fields.

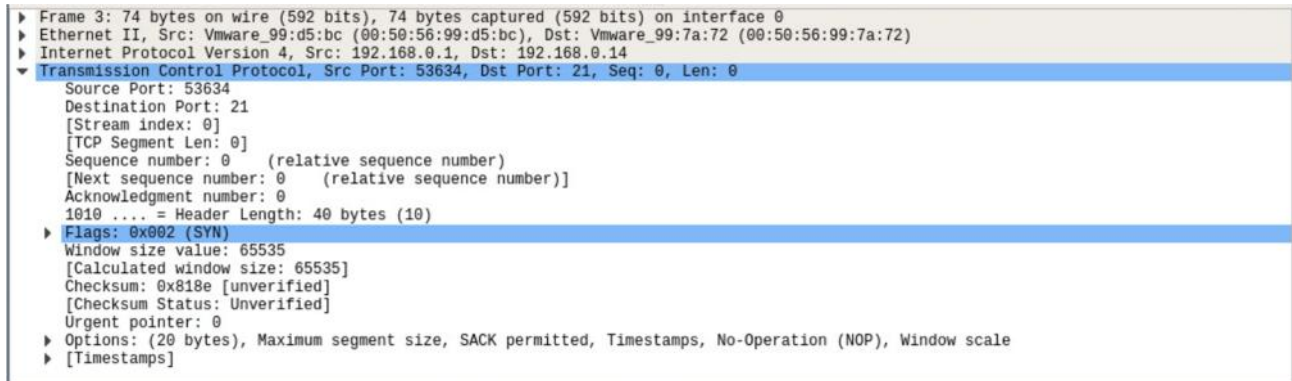
After the TCP filter has been applied, the first three packets (top section) display the sequence of [SYN], [SYN, ACK], and [ACK] which is the TCP three-way handshake.

3	2020-08-10 16:16:37.33412	192.168.0.1	192.168.0.14	TCP	74	53634 → 21 [SYN] Seq=0 Win=65535 Len=0 ...
4	2020-08-10 16:16:37.33447	192.168.0.14	192.168.0.1	TCP	74	21 → 53634 [SYN, ACK] Seq=0 Ack=1 Win=6...
5	2020-08-10 16:16:37.33449	192.168.0.1	192.168.0.14	TCP	66	53634 → 21 [ACK] Seq=1 Ack=1 Win=65536 ...

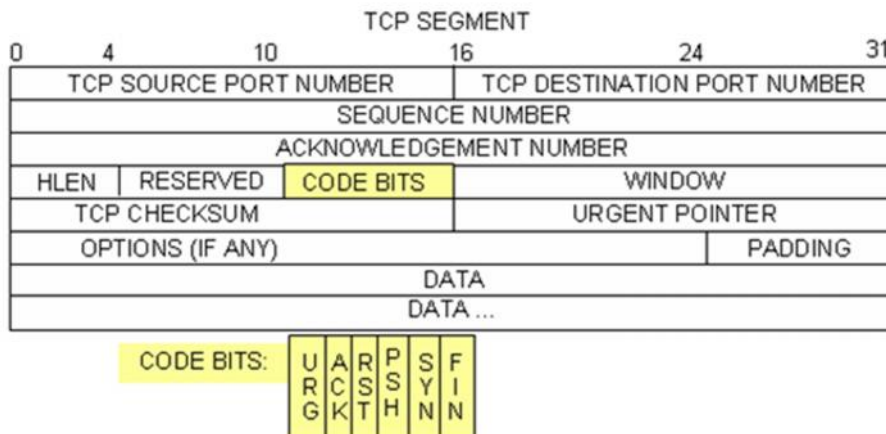
TCP is routinely used during a session to control datagram delivery, verify datagram arrival, and manage window size. For each data exchange between the FTP client and FTP server, a new TCP session is started. At the conclusion of the data transfer, the TCP session is closed. When the FTP session is finished, TCP performs an orderly shutdown and termination.

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

In Wireshark, detailed TCP information is available in the packet details pane (middle section). Highlight the first TCP datagram from the host computer, and expand portions of the TCP datagram, as shown below.



The expanded TCP datagram appears similar to the packet detail pane, as shown below.



The image above is a TCP datagram diagram. An explanation of each field is provided for reference:

- The **TCP source port number** belongs to the TCP session host that opened a connection. The value is normally a random value above 1,023.
- The **TCP destination port number** is used to identify the upper layer protocol or application on the remote site. The values in the range 0–1,023 represent the “well-known ports” and are associated with popular services and applications (as described in RFC 1700), such as Telnet, FTP, and HTTP. The combination of the source IP address, source port, destination IP address, and destination port uniquely identifies the session to the sender and receiver.

Note: In the Wireshark capture above, the destination port is 21, which is FTP. FTP servers listen on port 21 for FTP client connections.

- The **Sequence number** specifies the number of the last octet in a segment.
- The **Acknowledgment number** specifies the next octet expected by the receiver.
- The **Code bits** have a special meaning in session management and in the treatment of segments. Among interesting values are:
 - **ACK** — Acknowledgment of a segment receipt.
 - **SYN** — Synchronize, only set when a new TCP session is negotiated during the TCP three-way handshake.

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

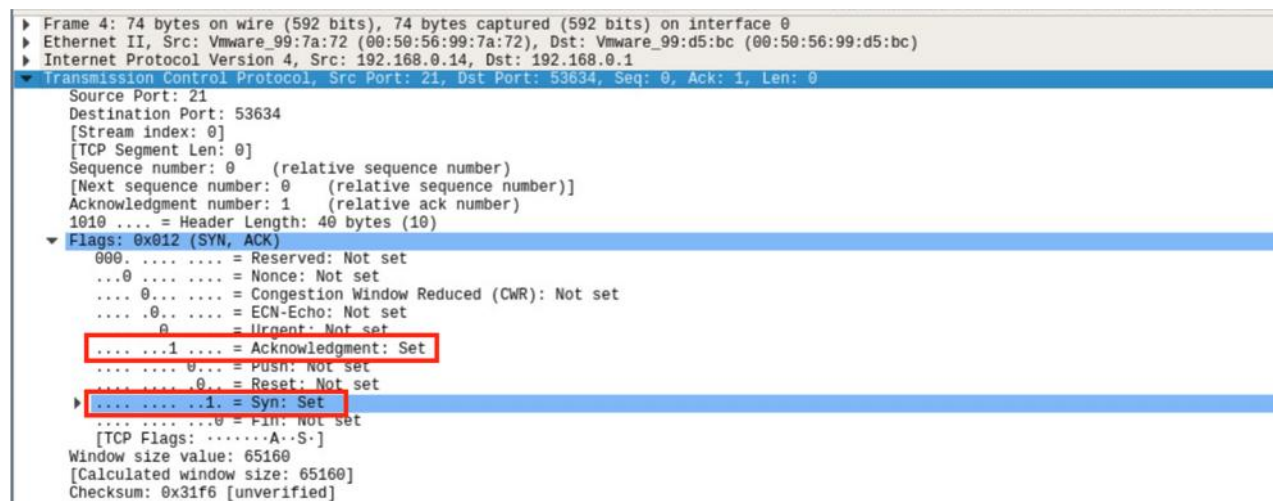
- **FIN** — Finish, the request to close the TCP session.
- The **Window size** is the value of the sliding window. It determines how many octets can be sent before waiting for an acknowledgment.
- The **Urgent pointer** is only used with an Urgent (URG) flag when the sender needs to send urgent data to the receiver.
- The **Options** has only one option currently, and it is defined as the maximum TCP segment size (optional value).

Using the Wireshark capture of the first TCP session startup (SYN bit set to 1), fill in information about the TCP header. Some fields may not apply to this packet.

From the VM to CDC server (only the SYN bit is set to 1):

Description	Wireshark Results
Source IP address	
Destination IP address	
Source port number	
Destination port number	
Sequence number	
Acknowledgment number	
Header length	
Window size	

In the second Wireshark filtered capture, the CDC FTP server acknowledges the request from the VM. Note the values of the SYN and ACK bits.



Fill in the following information regarding the SYN-ACK message.

Description	Wireshark Results
Source IP address	
Destination IP address	

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

Description	Wireshark Results
Source port number	
Destination port number	
Sequence number	
Acknowledgment number	
Header length	
Window size	

In the final stage of the negotiation to establish communications, the VM sends an acknowledgment message to the server. Select the third Wireshark filtered capture. Notice that only the ACK bit is set to 1, and the Sequence number has been incremented to 1.

```
▶ Frame 5: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
▶ Ethernet II, Src: Vmware_99:d5:bc (00:50:56:99:d5:bc), Dst: Vmware_99:7a:72 (00:50:56:99:7a:72)
▶ Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.14
▼ Transmission Control Protocol, Src Port: 53634, Dst Port: 21, Seq: 1, Ack: 1, Len: 0
  Source Port: 53634
  Destination Port: 21
  [Stream index: 0]
  [TCP Segment Len: 0]
  Sequence number: 1 (relative sequence number)
  [Next sequence number: 1 (relative sequence number)]
  Acknowledgment number: 1 (relative ack number)
  1000 .... = Header Length: 32 bytes (8)
  ▼ 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 size value: 128
  [Calculated window size: 65536]
  [Window size scaling factor: 512]
```

Fill in the following information regarding the ACK message.

Description	Wireshark Results
Source IP address	
Destination IP address	
Source port number	
Destination port number	
Sequence number	
Acknowledgment number	
Header length	
Window size	

How many other TCP datagrams contained a SYN bit?

After a TCP session is established, FTP traffic can occur between the PC and FTP server. The FTP client and server communicate with each other, unaware that TCP has control and management over the session.

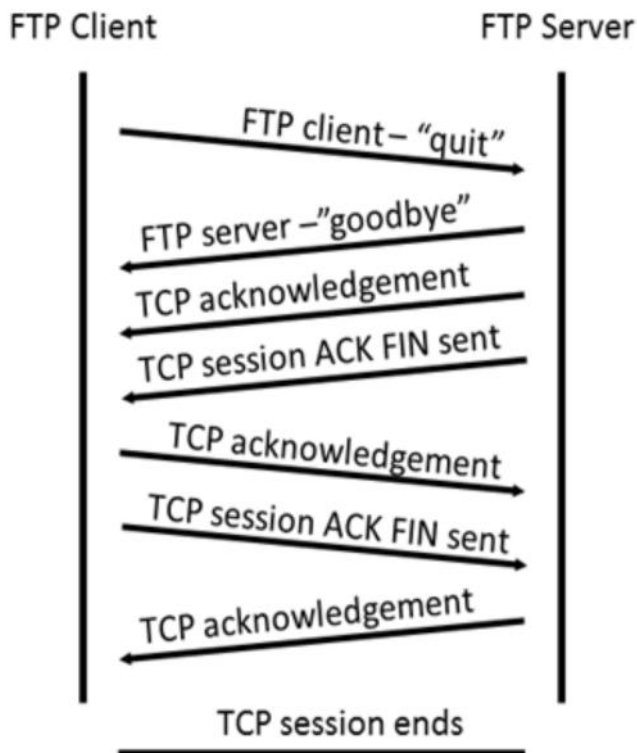
10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

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 the server. This sequence is visible in the Wireshark capture below.

6	2020-08-10 16:16:37.33877...	192.168.0.14	192.168.0.1	FTP	113 Response: 220 Welcome to Cisco CyberOPS...
7	2020-08-10 16:16:37.33879...	192.168.0.1	192.168.0.14	TCP	66 53634 → 21 [ACK] Seq=1 Ack=48 Win=65536...
8	2020-08-10 16:16:42.22646...	192.168.0.1	192.168.0.14	FTP	82 Request: USER anonymous
9	2020-08-10 16:16:42.22681...	192.168.0.14	192.168.0.1	TCP	66 21 → 53634 [ACK] Seq=48 Ack=17 Win=6528...
10	2020-08-10 16:16:42.22692...	192.168.0.14	192.168.0.1	FTP	100 Response: 331 Please specify the passwo...
11	2020-08-10 16:16:42.22692...	192.168.0.1	192.168.0.14	TCP	66 53634 → 21 [ACK] Seq=17 Ack=82 Win=6553...
13	2020-08-10 16:16:43.98401...	192.168.0.1	192.168.0.14	FTP	73 Request: PASS

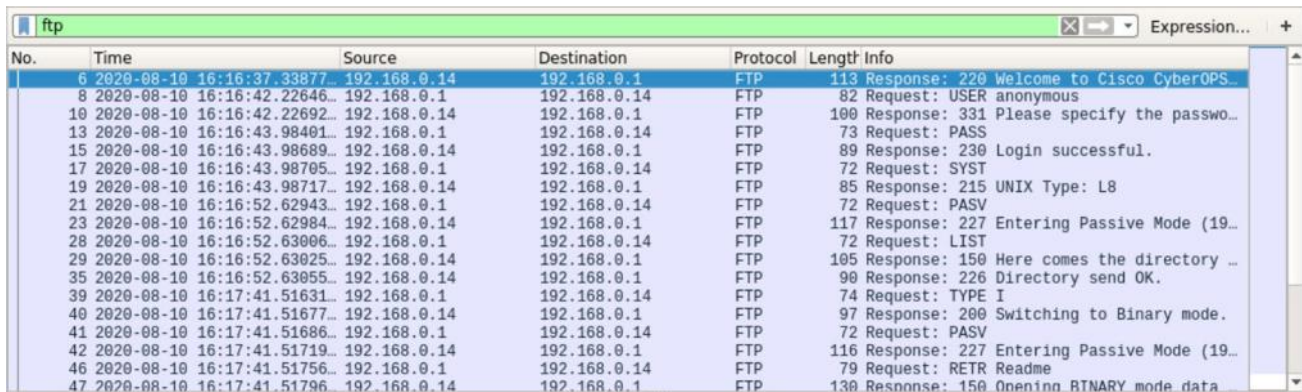
▶	Frame 6: 113 bytes on wire (904 bits), 113 bytes captured (904 bits) on interface 0
▶	Ethernet II, Src: Vmware_99:7a:72 (00:50:56:99:7a:72), Dst: Vmware_99:d5:bc (00:50:56:99:d5:bc)
▶	Internet Protocol Version 4, Src: 192.168.0.14, Dst: 192.168.0.1
▶	Transmission Control Protocol, Src Port: 21, Dst Port: 53634, Seq: 1, Ack: 1, Len: 47
▼	File Transfer Protocol (FTP)
▼	220 Welcome to Cisco CyberOPS VM FTP service.\r\n
	Response code: Service ready for new user (220)
	Response arg: Welcome to Cisco CyberOPS VM FTP service.
	[Current working directory:]

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*. At this time, the FTP server TCP session sends a TCP datagram to the FTP client, announcing the termination of the TCP session. The FTP client TCP session acknowledges receipt of the termination datagram, then sends its own TCP session termination. When the originator of the TCP termination (the FTP server) receives a duplicate termination, an ACK datagram is sent to acknowledge the termination and the TCP session is closed. This sequence is visible in the diagram and capture below.



10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

By applying an **ftp** filter, the entire sequence of the FTP traffic can be examined in Wireshark. Notice the sequence of the events during this FTP session. The username **anonymous** was used to retrieve the Readme file. After the file transfer completed, the user ended the FTP session.

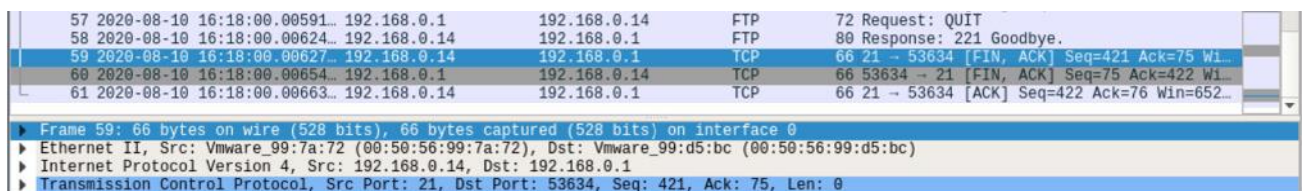


No.	Time	Source	Destination	Protocol	Length	Info
6	2020-08-10 16:16:37.33877...	192.168.0.14	192.168.0.1	FTP	113	Response: 220 Welcome to Cisco CyberOPS...
8	2020-08-10 16:16:42.22646...	192.168.0.1	192.168.0.14	FTP	82	Request: USER anonymous
10	2020-08-10 16:16:42.22692...	192.168.0.14	192.168.0.1	FTP	100	Response: 331 Please specify the passwo...
13	2020-08-10 16:16:43.98401...	192.168.0.1	192.168.0.14	FTP	73	Request: PASS
15	2020-08-10 16:16:43.98689...	192.168.0.14	192.168.0.1	FTP	89	Response: 230 Login successful.
17	2020-08-10 16:16:43.98705...	192.168.0.1	192.168.0.14	FTP	72	Request: SYST
19	2020-08-10 16:16:43.98717...	192.168.0.14	192.168.0.1	FTP	85	Response: 215 UNIX Type: L8
21	2020-08-10 16:16:52.62943...	192.168.0.1	192.168.0.14	FTP	72	Request: PASV
23	2020-08-10 16:16:52.62984...	192.168.0.14	192.168.0.1	FTP	117	Response: 227 Entering Passive Mode (19...
28	2020-08-10 16:16:52.63006...	192.168.0.1	192.168.0.14	FTP	72	Request: LIST
29	2020-08-10 16:16:52.63025...	192.168.0.14	192.168.0.1	FTP	105	Response: 150 Here comes the directory ...
35	2020-08-10 16:16:52.63055...	192.168.0.14	192.168.0.1	FTP	90	Response: 226 Directory send OK.
39	2020-08-10 16:17:41.51631...	192.168.0.1	192.168.0.14	FTP	74	Request: TYPE I
40	2020-08-10 16:17:41.51677...	192.168.0.14	192.168.0.1	FTP	97	Response: 200 Switching to Binary mode.
41	2020-08-10 16:17:41.51686...	192.168.0.1	192.168.0.14	FTP	72	Request: PASV
42	2020-08-10 16:17:41.51719...	192.168.0.14	192.168.0.1	FTP	116	Response: 227 Entering Passive Mode (19...
46	2020-08-10 16:17:41.51756...	192.168.0.1	192.168.0.14	FTP	79	Request: RETR Readme
47	2020-08-10 16:17:41.51796...	192.168.0.14	192.168.0.1	FTP	130	Response: 150 Opening RTNARY mode data

Apply the TCP filter again in Wireshark to examine the termination of the TCP session. Four packets are transmitted for the termination of the TCP session. Because TCP connection is full duplex, each direction must terminate independently. Examine the source and destination addresses.

In this example, the FTP server has no more data to send in the stream. It sends a segment with the FIN flag set in frame 59. The PC sends an ACK to acknowledge the receipt of the FIN to terminate the session from the server to the client in frame 60.

In frame 60, the PC sends a FIN to the FTP server to terminate the TCP session. The FTP server responds with an ACK to acknowledge the FIN from the PC in frame 61. Now the TCP session is terminated between the FTP server and PC.



No.	Time	Source	Destination	Protocol	Length	Info
57	2020-08-10 16:18:00.00591...	192.168.0.1	192.168.0.14	FTP	72	Request: QUIT
58	2020-08-10 16:18:00.00624...	192.168.0.14	192.168.0.1	FTP	80	Response: 221 Goodbye.
59	2020-08-10 16:18:00.00627...	192.168.0.14	192.168.0.1	TCP	66	21 → 53634 [FIN, ACK] Seq=421 Ack=75 Wl...
60	2020-08-10 16:18:00.00654...	192.168.0.1	192.168.0.14	TCP	66	53634 → 21 [FIN, ACK] Seq=75 Ack=422 Wl...
61	2020-08-10 16:18:00.00663...	192.168.0.14	192.168.0.1	TCP	66	21 → 53634 [ACK] Seq=422 Ack=76 Wln=652...

Close everything you opened. Choose not to save if asked.

Part 2: Identify UDP Header Fields and Operation Using a Wireshark TFTP Session Capture

In Part 2, you use Wireshark to capture a TFTP session and inspect the UDP header fields.

Step 1: Start Mininet and tftpd service.

a. Log into the **Workstation** VM. Open a **Terminal** window.

b. Run the following command, enter **cyberops** as the password when prompted.

```
[analyst@secOps ~]$ sudo lab.support.files/scripts/cyberops_topo.py
[sudo] password for analyst:
```

c. Start H1 and H2 at the **mininet>** prompt.

```
*** Starting CLI:
mininet> xterm H1 H2
```

d. In the **H1** terminal window, start the tftpd server using the provided script.

```
[root@secOps analyst]# /home/analyst/lab.support.files/scripts/start_tftpd.sh
[root@secOps analyst]#
```

Step 2: Create a file for tftp transfer.

- a. Create a text file at the **H1** terminal prompt in the /srv/tftp/ folder.

```
[root@secOps analyst]# echo "This file contains my tftp data." > /srv/tftp/my_tftp_data
```

- b. Verify that the file has been created with the desired data in the folder.

```
[root@secOps analyst]# cat /srv/tftp/my_tftp_data
This file contains my tftp data.
```

- c. Because of the security measure for this particular tftp server, the name of the receiving file needs to exist already. On **H2**, create a file named **my_tftp_data**.

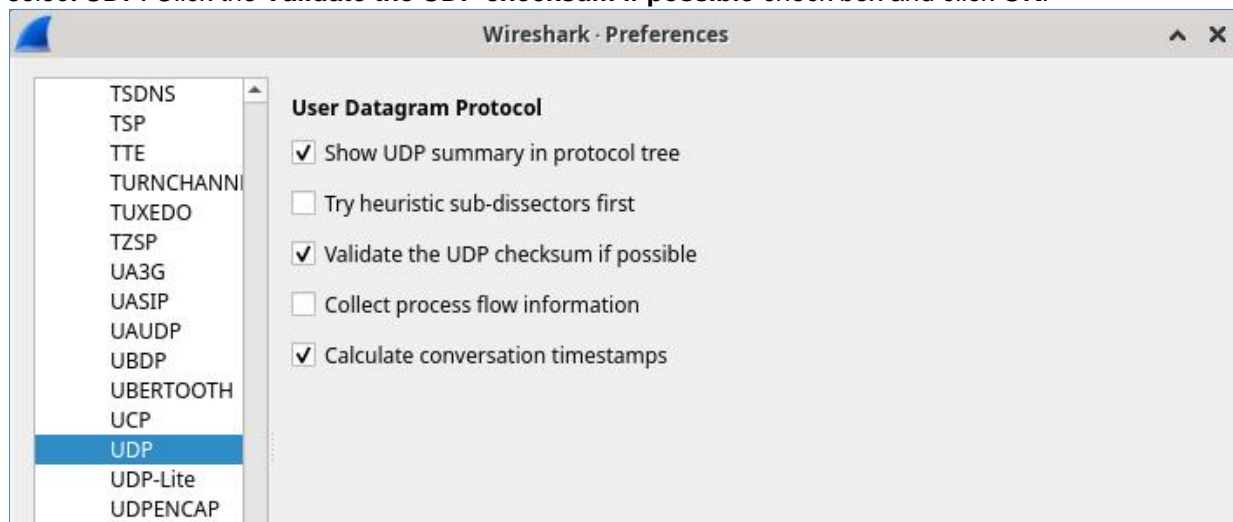
```
[root@secOps analyst]# touch my_tftp_data
```

Step 3: Capture a TFTP session in Wireshark

- a. Start Wireshark in **H1**.

```
[root@secOps analyst]# wireshark &
```

- b. From the **Edit** menu, choose **Preferences** and click the arrow to expand **Protocols**. Scroll down and select **UDP**. Click the **Validate the UDP checksum if possible** check box and click **OK**.



- c. Start a Wireshark capture on the interface **H1-eth0**.

- d. Start a tftp session from **H2** to the tftp server on **H1** and get the file **my_tftp_data**.

```
[root@secOps analyst]# tftp 10.0.0.11 -c get my_tftp_data
```

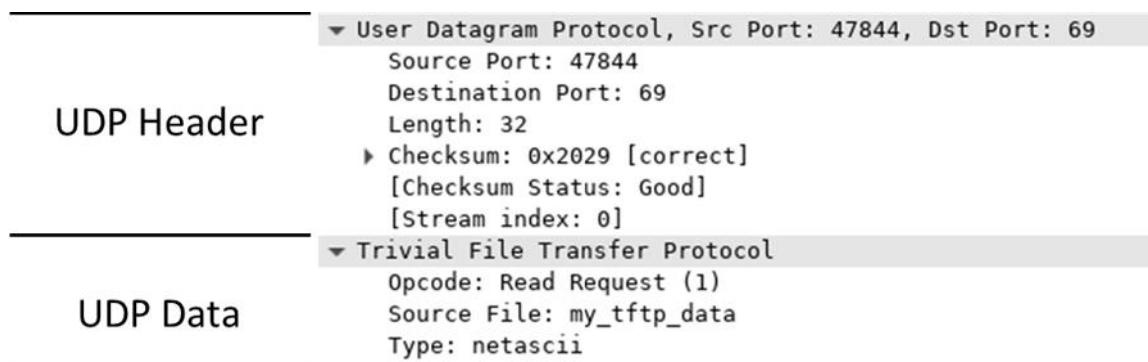
- e. Stop the Wireshark capture. Set the filter to **tftp** and click **Apply**. Use the three TFTP packets to fill in the table and answer the questions in the rest of this lab.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.0.12	10.0.0.11	TFTP	66	Read Request, File: my_tftp_data, Transfer
2	0.001295043	10.0.0.11	10.0.0.12	TFTP	80	Data Packet, Block: 1 (last)
3	0.001735272	10.0.0.12	10.0.0.11	TFTP	46	Acknowledgement, Block: 1

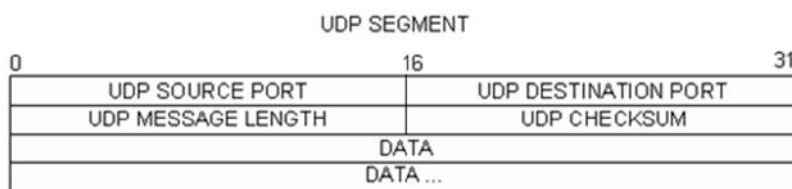
Detailed UDP information is available in the Wireshark packet details pane. Highlight the first UDP datagram from the host computer and move the mouse pointer to the packet details pane. It may be

10.4.3 Lab - Using Wireshark to Examine TCP and UDP Captures

necessary to adjust the packet details pane and expand the UDP record by clicking the protocol expand box. The expanded UDP datagram should look similar to the diagram below.



The figure below is a UDP datagram diagram. Header information is sparse, compared to the TCP datagram. Similar to TCP, each UDP datagram is identified by the UDP source port and UDP destination port.



Using the Wireshark capture of the first UDP datagram, fill in information about the UDP header. The checksum value is a hexadecimal (base 16) value, denoted by the preceding 0x code:

Description	Wireshark Results
Source IP address	
Destination IP address	
Source port number	
Destination port number	
UDP message length	
UDP checksum	

How does UDP verify datagram integrity?

Using the Wireshark capture of the second UDP datagram, the first frame returned from the tftp server, fill in the information about the UDP header:

Description	Wireshark Results
Source IP address	
Destination IP address	

Description	Wireshark Results
Source port number	
Destination port number	
UDP message length	
UDP checksum	

Notice that the return UDP datagram has a different UDP source port, but this source port is used for the remainder of the TFTP transfer. Because there is no reliable connection, only the original source port used to begin the TFTP session is used to maintain the TFTP transfer.

Also, notice that the UDP Checksum is incorrect. This is most likely caused by UDP checksum offload. You can learn more about why this happens by searching for “UDP checksum offload”.

Step 4: Clean up

In this step, you will shut down and clean up Mininet.

- In the terminal that started Mininet, enter **quit** at the prompt.

```
mininet> quit
```

- At the prompt, enter **sudo mn -c** to clean up the processes started by Mininet.

```
[analyst@secOps ~]$ sudo mn -c
```

Reflection Question

This lab provided the opportunity to analyze TCP and UDP protocol operations from captured FTP and TFTP sessions. How does TCP manage communication differently than UDP?
